UNIVERSITÄT
SIEGEN

Modern Optimization Techniques in Computer Vision

— From variational models to machine learning security —

DISSERTATION

zur Erlangung des Grades eines Doktors der Naturwissenschaften

vorgelegt von

Jonas Alexander Geiping, M.Sc.

geb. am 25.03.1992 in Berlin, Tempelhof

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Hölderlinstraße 3,
57076 Siegen

Januar 2021

Betreuer und erster Gutachter
Prof. Michael Möller
Universität Siegen, Computer Vision Chair

Externer Gutachter
Prof. Tom Goldstein
Universität of Maryland, College Park, Perotto Chair of Machine Learning

Tag der mündlichen Prüfung
9. April 2021

Zusammenfassung

Diese Arbeit präsentiert Forschungsergebnisse in mehreren Problemstellungen aus dem Feld des maschinellen Sehens und diskutiert generalisierte Optimierungsstrategien für solche mit einem konzeptuellen Fokus auf kompositionellen Optimierungsstrategien wie Bi-Level Optimierung. Die optimale Graph-basierte Diskretisierung von Variationsproblemen in Rahmen von Minimalen Partitionen, die theoretische Analyse von kompositioneller Optimierung durch nichtkonvexe Majorisierer, die Aufgabe des Lernens von Energiemodellen durch nichtkonvexe Majorisierer, und die Anwendungen dieser Bilevel-Optimierung im Rahmen der Sicherheitsanalyse von maschinellem Lernen beim Datenschutz in föderierten Lernverfahren und potentiellen Verfälschungen von Bilddaten für Bildklassifikation, sind Themen dieser kumulativen Arbeit.

Abstract

This thesis presents research into multiple optimization topics in computer vision with a conceptual focus on composite optimization problems such as bilevel optimization. The optimal graph-based discretization of variational problems in minimal partitions, the theoretical analysis of nonconvex composite optimization by nonconvex majorizers, the bilevel problem of learning energy models by nonconvex majorizers, and the machine learning security applications of bilevel optimization in privacy analysis of federated learning and dataset poisoning of general image classification are featured in this cumulative work.

Acknowledgements

Danke an alle, die mich auf diesem Weg unterstützt und begleitet haben. Thank you all, for the shared memories of grand ideas, of tight deadlines and of all those days that just came and went:

Michael Möller, who must have advised several hundred students in secret, to give the great advice he gave for these years; for his brightness, serenity, trust and time. From the first two rooms, through Venice, GRK presentations, and the creation of a fledgling group, then to catch up on the other side of the planet: Danke für alles, und das ist schon eine Menge.

Tom Goldstein, for his support and for his friendly invitation to visit UMD; for his optimism and his wealth of ideas and his time. Let's have another boat for lunch some time after all of this pandemic stuff.

Fjedor Gaede, für das Proofreading dieser Dissertation, aber für allem für die vielen ideenreichen Stunden mit Papier und Kaffee, für die wir zum Glück immer wieder Zeit finden konnten, trotz der Entfernung.

The DFG Research Training Group 1564 of the University of Siegen, und stellvertretend Andreas Kolb, for their warm welcome and the many events - I was glad to be a part of such a vibrant group when I started alone.

Der GRK Club, Markus Kluge, Tak Ming Wong, Andreas Görlitz, danke für die regelmäßigen Dosen Normalität, getarnt als Mensa, Dinosaurierrunde, Gartenhausbesuch oder auch nur Kaffee und Spekulatius. From the before time, not to forget Gianni Franchi, Nadine Hoffmann and Davoud Shahlaei.

Die neugefundene Computer Vision, danke an Hartmut Bauermeister, Hannah Dröge, Vaishnavi Gandikota und Marcel Seelbach, für die vielen spannenden Treffen und Gespräche. Danke an Willi Gräfrath und Sarah Wagener, für die Hilfe bei den wichtigen Dingen, die ich sonst vergessen hätte.

The people from UMD, for their immediate welcome and the many hours both in and outside the university we crammed into those few months; The Norbert Wiener Center, for graciously letting me hang out in their lab unannounced, the Computer Science Department and administration. In short: Micah Goldblum, Liam Fowl, Avi Schwarzschild, also for our many food adventures; Ronny Huang and Ali Shafahi, also for their kindness; and everyone else within the wider circle around Tom's group for having me over for a bit.

Meine Familie, Greta, Frederik, Philipp, Lukas und Daniel, meine Eltern Karin und Norbert. Dank eurer pausenlosen Unterstützung, war ich dann doch irgendwann plötzlich fertig.

Die Leute aus Herbern, danke für eure Freundschaft und eure Geduld, nicht nur mit dem RB91.

Also, I would like to thank all of my co-authors, even if mentioned before: I've greatly enjoyed working with all of you and it is one of the major perks of our field to be able to work with so many interesting, insightful and curious people. I'd like to thank Hendrik Dirks, Daniel Cremers, Michael Moeller, Andreas Görlitz, Andreas Kolb, Micah Goldblum, Avi Schwarzschild, Tom Goldstein, W. Ronny Huang, Liam Fowl, Gavin Taylor, Ping-Yeh Chiang, Renkun Ni, Steven Reich, Ali Shafahi, Fjedor Gaede, Hartmut Bauermeister, Hannah Dröge, Wojciech Czaja, Eitan Borgnia, Valeriia Cherepanova, Amin Ghiasi and Arjun Gupta.

So, I decided to simulate a universe...So, if you see a mote of dust vanish from your vision...I must have misplaced a rock...sometime in the last few billions and billions of millenia...

[203]

Contents

CHAPTER 1

Introduction

Mathematical optimization is a central framework underpinning a multitude of applications in computer vision. Fundamentally, optimization can be understood as a principled strategy to convert a measure of success to an algorithm that computes successes: Any application, task or problem for which the set of possible solutions can be defined in conjunction with a function that measures the quality of solutions directly poses an optimization problem - How can we find the best solution from the set of possible solutions? The key component in this construction is the function that measures success; even if we were to disregard optimization as a framework, a formalized measure of success is necessary to scientifically understand, rank and falsify solutions proposed by novel algorithms. As soon as such a measure is present, it can be optimized, generally by considering approximations to this measure which enjoy properties beneficial to optimization theory, but whose difference to the original measure can be exactly quantified. In the field of computer vision, this understanding allows us to approach applications that are seemingly dissimilar from a practical point of view, with shared tools and optimization strategies.

This cumulative dissertation contains four recently published works Geiping et al., Geiping & Moeller, Geiping & Moeller, Geiping et al. [2020, 2018, 2019, 2020] and one work accepted for publication Geiping et al. [2021] from the intersection of machine learning, variational methods and optimization. A key theme shared in these studies is the construction of understanding for novel challenges from a shared perspective of mathematical optimization, and further, the application of developed concepts to practical issues in fields such as machine learning security and variational methods. This work represents central parts of the corpus of work published as PhD student at the University of Siegen, alongside Geiping et al., Görlitz et al., Chiang et al., Huang et al., Goldblum et al. [2018, 2019, 2020, 2020, 2020] and Borgnia et al. [2020].

The optimization problems investigated in this thesis can be categorized under a variety of denominations, depending on sub-field and method of derivation, as a variational problem or variational model, maximum-a-posteriori estimate, energy model, energy-based model, among others. We formalize the optimization problem via

$$x_{\text{opt}} \in \arg\min_{x \in X} E(x, y), \tag{1.1}$$

for some set X of possible solutions x and an objective $E : X \times Y \to \mathbb{R}$ that measures success based on some data $y \in Y$. To give a concrete example, in a fundamental computer vision scenario, we might want to remove noise artifacts from camera images. Hence the input data $y \in Y$ represents a measured image contaminated by noise, X represents the space of all possible images and E represents a measure for success that depends on both the input data y and measures how "realistic" we consider a denoised candidate image $x \in X$. The output x_{opt} then represents an optimally denoised image.

From a perspective of *variational problems*, this optimization problem arises as the Lagrangian formulation of a partial differential equation describing the transformation from noisy to denoised image, for example via mean curvature motion [245]. More generally, the objective arises from some variational model, often, but not necessarily based on physical descriptions, such as for example minimal surfaces. From this interpretation, the objective E is often referred to as *energy*, so that the overall problem is named an *energy model* - but this naming convention does not require a physical energy. We consider optimization strategies for a variational model based on minimal partitions in Chapter 2 and examples for variational models for video super-resolution and scene flow were studied in Geiping et al. [2018] and Görlitz et al. [2019].

Statistical estimation instead considers the fundamental object to be a probability distribution $p(x, y)$ over all natural images similar to the input data y. Returning to the previous example, the denoised image is then the most likely image from this distribution, and can be found by *maximum likelihood estimation* of p. The maximum likelihood estimate (MLE) is equivalent to minimizing the negative log-likehood, so that we again recover the optimization problem via $E(x, y) = -\log(p(x, y))$. For systems where the probability distribution can be expressed as Boltzmann distribution, this connection is immediate, as then p is directly defined by $p(x, y) \propto e^{-E(x,y)}$ [121]. Energy models arising from this field are also described as energy-based models [168].

The relation of x and y can be made precise by considering the main task as estimating the *posterior distribution* $p(x|y)$, i.e. a single probability distribution p which is conditional on the input data y [28]. By Bayes' theorem the maximum-a-posteriori estimate (MAP) is equivalent to $p(y|x)\frac{p(x)}{p(y)}$, for which minimizing the negative logarithm can be simplified to

$$x_{\text{opt}} \in \arg\min_{x \in X} -\log(p(y|x)) - \log p(x). \tag{1.2}$$

This structured optimization problem naturally consists of two terms, the first, named data likelihood or data fidelity, measures how closely a candidate solution x resembles the data y, while the second term, the regularizer, introduces prior information that measures how much x resembles a natural image.

The more structure is known for an optimization problem, the better it can be solved and a wide range of optimization algorithms are designed to target specific structures, such as the additive model $E(x, y) = G(x, y) + R(x)$ as arising from the MAP estimate above. Chapter 3 features a theoretical discussion and analysis of problems with the structure

$$x_{\text{opt}} \in \arg\min_{x \in X} G(\rho(x), y) + R(x), \tag{1.3}$$

also known as *composite* optimization problems, due to the additionally present composition of $G \circ \rho$ in the first term for some nonlinear mapping $\rho : X \to Z$ and $G : Z \times Y \to \mathbb{R}$. This structure arises naturally, for example, when considering the reconstruction and denoising of nonlinear measurements, such as in Time-of-Flight Imaging.

The *training* of machine learning models on the other hand, as a special case of function appproximations, induces an optimization problem parametrizing an unknown function $n(y, \theta) : Y \times \Theta \to X$ via parameters $\theta \in \Theta$ and optimizing these parameters for given data points $\{y_i, x_i\}_{i=1}^N$ from a space of observations $Y \times X$ and for a given loss function $\mathcal{L} : X \times X \to \mathbb{R}$ by

$$\theta_{\text{opt}} \in \arg\min_{\theta \in \Theta} \sum_{i=1}^N \mathcal{L}(n(y_i, \theta), x_i). \tag{1.4}$$

This is also a composite optimization problem, optimizing $\mathcal{L} \circ n$. For the example of denoising, n would be a function optimized to map from a noisy input y to a denoised output x, based on exemplary pairs (y_i, x_i) of noisy and noisy-free images. Most important in this scenario is the context switch of the variable that is optimized. Previously this variable directly represented the solution to the problem, i.e. the denoised image, but here it represents the parameters of a function n which itself returns the solution to the problem. The semantic meaning of these variables changes, but the problem is mathematically structured in the same way. Empirical investigations about the theory of machine learning optimization were further studied in Goldblum et al. [2020].

For both the energy models of (1.1) and special case of the machine learning models of (1.4), a crucial optimization structure featured in multiple parts of this thesis, are *bilevel optimization problems*. These are composite optimization problems, where the inner function ρ in (1.3) itself is given implicitly as the solution to another optimization problem:

$$z_{\text{opt}} \in \arg\min_{z \in Z} G(\rho(z), y) + R(z) \quad \text{s.t. } \rho(z) = \arg\min_{x \in X} F(x, z). \tag{1.5}$$

This structure is essential when reasoning about optimization problems from a meta-perspective - allowing for the optimization of components of an optimization problem. Because optimization problems are used ubiquitously, the task of optimizing about optimization problems appears frequently. It may encode applications such as control over the optimization task, data-specific modifications, or the introduction of side-effects into the optimization problem.

In Chapter 4 we investigate energy models such as (1.1), but are interested in parametrizing these energy models and training them based on observational data as in a machine learning model (1.4). In this manner, we are faced with the bilevel problem of optimizing properties of an optimization problem. For our example of denoising, we consider learning an optimal regularizer R as one of the applications in Chapter 4:

$$\theta_{\text{opt}} \in \arg\min_{\theta \in \Theta} \sum_{i=1}^{N} \mathcal{L}(n(y_i, \theta), x_i) \quad \text{s.t. } n(y, \theta) = \arg\min_{x \in X} G(x, y) + R(x, \theta). \tag{1.6}$$

In comparison to (1.4), the process of denoising is not represented directly by an explicitly given function $n(\theta, y)$, but implicitly by an energy optimization as in (1.1) whose parameters are trained to be optimal. From the statistical perspective, this corresponds to learning to approximate the probability distributions that represent the prior knowledge about the problem.

Chapter 5 and Chapter 6 contain applications in security for machine learning models. Both chapters investigate the vulnerability of the training process of machine learning models, (1.4), to malicious side-effects. In Chapter 5 we discuss that private data can be uncovered in models trained by federated learning, a specific collaborative optimization algorithm. Chapter 6 and Huang et al. [2020] feature data poisoning. The data used to optimize a machine learning model is optimized to achieve some malicious effect. If we denote the measure of success of this malicious effect by $\mathcal{L}_{\text{adv}}(\theta)$, then the attack can be formalized as the bilevel problem of

$$x_{\text{opt}} \in \arg\min_{x \in X} \mathcal{L}_{\text{adv}}(\theta(x)) \quad \text{s.t. } \theta(x) \in \arg\min_{\theta \in \Theta} \sum_{i=1}^{N} \mathcal{L}(n(y_i, \theta), x_i). \tag{1.7}$$

In both chapters, the formalization of an attack as bilevel optimization problem with respect to the training optimization uncovers security risks. Adversarial goals can be introduced as side-effects when the original optimization problem is manipulated in this manner. These applications aid in analyzing possible security risks of modern machine learning systems. Some preliminary information about defensive strategies is provided in Borgnia et al. [2020].

Formally, this work thus presents four recently published, internationally peer-reviewed publications Geiping et al., Geiping & Moeller, Geiping & Moeller, Geiping et al. [2020, 2018, 2019, 2020] as well as one accepted for publication Geiping et al. [2021], in each chapter in the original text of their respective final versions, with updated formatting and minor corrections. Every chapter is preceded by a discussion section, containing a brief overview over each project, conceptualizing the research done in the context of this thesis and delineating the contributions of the main author and many helpful collaborators who graciously contributed to this research.

In this manner the dissertation follows the statutory provisions for a cumulative dissertation laid out in the *Promotionsordnung der Naturwissenschaftlich-Technischen Fakultät* of the University of Siegen. Several chapters feature a second appendix which contains additional unpublished material that may be of interest to specialized readers.

List of Publications

[1] Ping-Yeh Chiang, Jonas Geiping, Micah Goldblum, Tom Goldstein, Renkun Ni, Steven Reich, and Ali Shafahi. Witchcraft: Efficient PGD Attacks with Random Step Size. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3747–3751, May 2020. doi: 10.1109/ICASSP40776.2020.9052930.

[2] Jonas Geiping and Michael Moeller. Composite Optimization by Nonconvex Majorization-Minimization. *SIAM Journal on Imaging Sciences*, pp. 2494–2528, January 2018. doi: 10.1137/18M1171989.

[3] Jonas Geiping and Michael Moeller. Parametric Majorization for Data-Driven Energy Minimization Methods. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 10262–10273, 2019. URL http://openaccess.thecvf.com/content_ICCV_2019/html/Geiping_Parametric_Majorization_for_Data-Driven_Energy_Minimization_Methods_ICCV_2019_paper.html.

[4] Jonas Geiping, Hendrik Dirks, Daniel Cremers, and Michael Moeller. Multiframe Motion Coupling for Video Super Resolution. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, pp. 123–138. Springer International Publishing, 2018. ISBN 978-3-319-78199-0. doi: 10.1007/978-3-319-78199-0_9.

[5] Jonas Geiping*, Hartmut Bauermeister*, Hannah Dröge*, and Michael Moeller. Inverting Gradients - How easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc//paper_files/paper/2020/hash/c4ede56bbd98819ae6112b20ac6bf145-Abstract.html.

[6] Jonas Geiping, Fjedor Gaede, Hartmut Bauermeister, and Michael Moeller. Fast Convex Relaxations using Graph Discretizations. In *31st British Machine Vision Conference (BMVC 2020, Oral Presentation)*, Virtual, September 2020. URL https://www.bmvc2020-conference.com/conference/papers/paper_0694.html.

[7] Jonas Geiping*, Liam H. Fowl*, W. Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller‡, and Tom Goldstein‡. Witches' Brew: Industrial Scale Data Poisoning via Gradient Matching. In *Ninth International Conference on Learning Representations (ICLR 2021) to be presented in*, May 2021. URL https://openreview.net/forum?id=01olnfLIbD.

[8] Micah Goldblum*, Jonas Geiping*, Avi Schwarzschild, Michael Moeller, and Tom Goldstein. Truth or backpropaganda? An empirical investigation of deep learning theory. In *Eighth International Conference on Learning Representations (ICLR 2020, Oral Presentation)*, April 2020. URL https://iclr.cc/virtual_2020/poster_HyxyIgHFvr.html.

[9] Andreas Görlitz, Jonas Geiping, and Andreas Kolb. Piecewise Rigid Scene Flow with Implicit Motion Segmentation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1758–1765, November 2019. doi: 10.1109/IROS40897.2019.8968018.

[10] W. Ronny Huang*, Jonas Geiping*, Liam Fowl, Gavin Taylor, and Tom Goldstein. MetaPoison: Practical General-purpose Clean-label Data Poisoning. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc//paper_files/paper/2020/hash/8ce6fc704072e351679ac97d4a985574-Abstract.html.

[11] Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, Tom Goldstein, and Arjun Gupta. Strong Data Augmentation Sanitizes Poisoning and Backdoor Attacks Without an Accuracy Tradeoff. Accepted for publication at *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. URL http://arxiv.org/abs/2011.09527.

Shared authorship denoted by *, ‡.

CHAPTER 2

Fast Convex Relaxations using Graph Discretizations

Contextualization

This chapter reprints the publication "Fast Convex Relaxations using Graph Discretizations", published as oral presentation at the British Machine Vision conference 2020 with co-authors Fjedor Gaede, Hartmut Bauermeister and Michael Moeller (Geiping et al. [2020] [103]). In the wider context of this thesis, this chapter is primarily focused on improving the practical efficiency of a particular variational problem as introduced in (1.1), namely the minimal partitions problem by redeveloping a good spatial discretization of the continuous problem that is relaxed to a convex problem by functional lifting.

This works thus falls into a line of work of the author Geiping et al., Görlitz et al. [2018, 2019] that investigates model-based approaches and their applications in computer vision. Crucially, these model-based, handcrafted techniques have been successful in many applications, especially with limited data, but currently tend to be outperformed by learning-based approaches especially in complex computer vision applications such as semantic segmentation or image classification. Variational problems, which as mentioned are an example of energy-based models, are well-suited to handcrafted modelling, due to their immediate formulation as maximum-a-posteriori estimates, see (1.2), yet this is also what makes it challenging to learn parts of the model: As discussed, this amounts to a bilevel optimization of parameters of the variational model as in (1.5), a problem which we will revisit later.

This work is practically an application of graph methods for variational models developed in [286] to the setting of continuous minimal partition problems, which are then solved via continuous functional lifting schemes. Fjedor Gaede as domain expert as such provided valuable insight into mathematical descriptions of graph-based, discrete variational problems and introduced, described and implemented the L^0-Cut-Pursuit algorithm which is utilized for graph construction in the paper and proved an earlier version of Proposition 2.1. Hartmut Bauermeister contributed valuable knowledge of continuous lifting schemes, checked mathematical formalism and conducted as well as described the experiments on stereo matching. Michael Moeller added important context and oversight regarding the topics of variational problems and continuous lifting, proposed and coded the application shown in Figure 2.4 and improved the writing and presentation of this work considerably. Jonas Geiping proposed this project after many discussions in collaboration with Fjedor Gaede, reviewed the literature, posed and derived the central mathematical models in subsection 2.3.2, implemented the approach for segmentation and stereo matching, conducted remaining experiments such as the quantitative and qualitative analysis featured in Figure 2.5, Figure 2.7, Table 2.1 and the appendix, and contributed chiefly to the writing of this work.

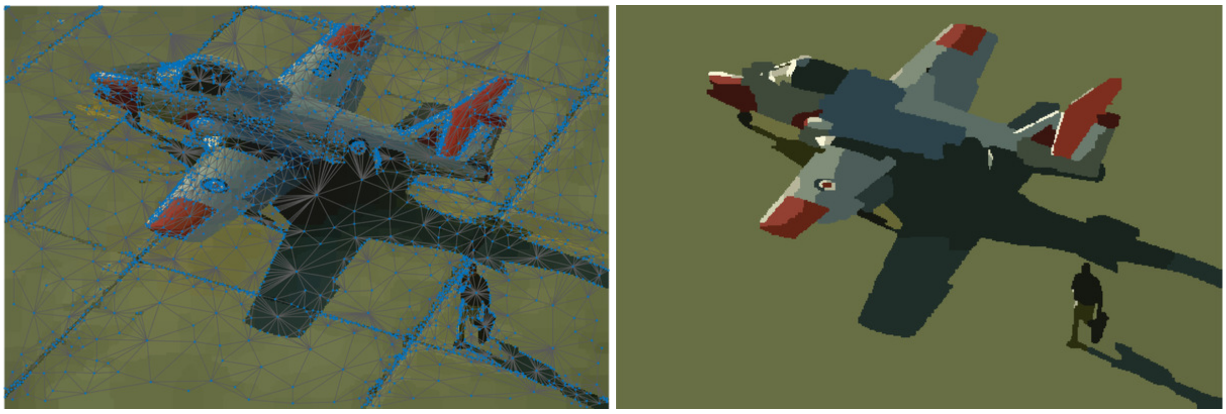


Figure 2.1: Graph discretization on the left and segmentation by convex relaxation with 32 labels, computed on the graph structure on the right. The segmentation is 3.03% less optimal compared to relaxation on a full Cartesian grid, but requires only around 4.8% of computation time, the original problem has 4.9 mio. variables, and the reduced problem only 120k.

Abstract

Matching and partitioning problems are fundamentals of computer vision applications with examples in multilabel segmentation, stereo estimation and optical-flow computation. These tasks can be posed as non-convex energy minimization problems and solved near-globally optimal by recent convex lifting approaches. Yet, applying these techniques comes with a significant computational effort, reducing their feasibility in practical applications. We discuss spatial discretization of continuous partitioning problems into a graph structure, generalizing discretization onto a Cartesian grid. This setup allows us to faithfully work on super-pixel graphs constructed by SLIC or Cut-Pursuit, massively decreasing the computational effort for lifted partitioning problems compared to a Cartesian grid, while optimal energy values remain similar: The global matching is still solved near-globally optimal. We discuss this methodology in detail and show examples in multi-label segmentation by minimal partitions and stereo estimation, where we demonstrate that the proposed graph discretization can reduce runtime as well as memory consumption of convex relaxations of matching problems by up to a factor of 10.

2.1 Introduction

Matching problems and the closely inter-related minimal partitioning problems are low-level computer vision tasks that build a backbone for a variety of applications such as multi-label segmentation [40], stereo estimation [132, 233] and optical flow estimation [123, 43]. However, posing these problems as energy minimization problems leads to non-convex objectives that are difficult to solve. In the last years, it has been demonstrated that functional lifting techniques are very well suited for solving these non-convex minimization problems via convex relaxations in a higher-dimensional space. Unfortunately, despite the precise solutions these methods provide, they incur significant costs in memory and computational effort, due to the high dimensionality of the lifted problem.

Consider the continuous minimal partitions problem, also referred to as piecewise-constant Mumford-Shah problem [202], which builds the basis of the aforementioned computer vision applications,

$$\min_{\{P_k\}_{k=1}^L} \sum_{k=1}^{L} \int_{P_k} -f_k(x) + \frac{1}{2}\operatorname{Per}(\Omega, P_k). \tag{2.1}$$

Given a set of L potential functions f_k we are looking for a partitioning of the set Ω into L non-overlapping partitions $\{P_k\}_{k=1}^L$, i.e. $P_k \cap P_l = \emptyset$ if $k \neq l$ and $\bigcup_{k=1}^L P_k = \Omega$ [53]. Without the regularizing perimeter term, the solution is given by $\arg\max_k f_k(x)$ for every $x \in \Omega$, but under inclusion of the second term in (2.1), the perimeter of each partition $\mathrm{Per}(\Omega, P_k)$ is penalized, leading to spatially coherent solutions, where every point is "matched" to a partition, but the global surface energy stays minimal.

Minimal partitions problems are abundant in imaging. In the discrete setting, they directly relate to the Potts model [230] and MRFs [41, 40]. This variability hinges on the choice of potential functions f_k. For multi-label segmentation, for example, we can consider each f_k to give the prior likelihood that the point x should be assigned label k, with the likelihood being computed by model-based approaches [77] or returned as the output of a neural network [64]. On the other hand, considering every partition with label k to encode a displacement of k pixels between two stereo images, the same framework can be used to solve stereo matching problems [302]. Further immediate examples include optical flow[123], scene flow and multiview reconstruction [151].

To be able to solve (2.1), we minimize over a set of partitions. This discrete matching problem at the heart of the minimal partitions problem is in general NP-hard. In practice, one of the most powerful approaches is the solution of a suitable convex relaxation of the original minimal partitions problem. To do so, the minimization over partitions is first replaced by minimization over their characteristic functions $u_k : \Omega \to \{0, 1\}$, satisfying $\sum_{k=1}^L u_k(x) = 1 \ \ \forall x \in \Omega$,

$$\min_{\{u_k\}_{k=1}^L} \sum_{k=1}^L \int_\Omega -f_k(x)u_k(x) + \int_\Omega |Du_k|, \tag{2.2}$$

where we have equivalently replaced the perimeter of a set by the total variation of its characteristic function, see [53]. In a next step, the functions u_k are relaxed to take values in the full interval $[0, 1]$, either directly [304, 169], or by jointly deriving tighter convex reformulations of the regularizer [53]. Relaxation approaches often lead to near-optimal solutions with high fidelity [229, 271], yet the computational effort amounts to solving a non-smooth, non-strongly convex optimization problem over all functions u_k, each of which is usually discretized to be as large as the given potential functions. Especially when k is large, memory costs quickly become impractical for computer vision applications.

In this work we hence consider strategies to remediate the computational costs of functional lifting techniques, without majorly impeding their global matching capabilities. As illustrated in Figure 2.1, we propose to first discretize the problem on a precomputed graph structure instead of a Cartesian grid, and then solve the convex relaxation on the graph. This strategy leads to significant computational advantages while sacrificing almost no accuracy in terms of the energy of the final solution.

2.2 Related Work

The minimal partitions problem is a prime example of energy minimization methods in computer vision [202, 245, 45], which have found widespread use. In the context of convex relaxations of these partition problems, there have been works in as diverse applications such as stereo estimation [229, 228, 295, 233, 234], optical flow [271, 269], segmentation [303, 304, 169, 53],and optimization on manifolds [170], with algorithmic improvements such as [267].

Previous work discusses the optimal discretization of the continuous label dimension [197, 165, 196], reducing the computational effort of functional lifting in a variety of applications. In our work, we discuss an orthogonal direction of research, as we are discussing compact discretizations of the image space.

The choice of efficient discretization of the input image data is directly related to superpixel approaches, e.g. [2, 290]. Their general idea is to generically reduce the computational complexity of any (pixel-based) numerical algorithm, by locally grouping pixels of similar color to larger superpixels. The most prominent algorithm in current practice is SLIC (Simple Iterative Linear Clustering) [2]. Ideally, the superpixel setup should also be chosen by an appropriate minimization procedure that adheres object edges. However, edge adherence is often costly. An interesting exception is the Cut-Pursuit algorithm [159, 160], which solves total variation minimization and related problems in a fast sequence of binary graph cuts, making it competitive as a discretization step and leading to boundaries that better adhere with minimal partitions. This approach has been successfully applied in practice in such works as [161, 112] and we will contrapose a superpixel structure generated by Cut Pursuit with one generated by SLIC in our main comparison to a Cartesian grid.

2.3 Graph Discretizations for Convex Relaxations

2.3.1 Preliminaries

Let us first introduce our general notation. For the discrete setup we consider an undirected graph structure is defined by its vertices V, edges connecting vertices $E \subset V \times V$ and weights of these edges $w \in \mathbb{R}^{|E|}$. We refer to [91] for details.

The continuous minimal partition problem requires the definition of the total variation of a function $u = (u_1, \dots, u_L) \in L^1(\Omega, \mathbb{R}^L)$ as

$$TV(u) = \sup \left\{ - \int_\Omega \sum_{k=1}^{L} u_k(x) \operatorname{div} \mathbf{p}_k(x)\, dx, \;\; \mathbf{p} \in C_c^1(\Omega, \mathbb{R}^{d\times L}), \;\; \sum_{k=1}^{L} |\mathbf{p}_k(x)|^2 \leq 1 \right\},$$

Note that the above definition reduces to $TV(u) = \sum_{k=1}^{L} \int_\Omega |\nabla u_k(x)|\, dx$ for smooth u. We define u to be an element of the space of bounded variation $SBV(\Omega, \mathbb{R}^L)$ if $TV(u)$ is finite. We can then identify this value with the mass of the distributional derivative Du, i.e. $\int_\Omega |Du| = TV(u)$. The bounded Radon measure Du can be decomposed [12, Thm. 10.4.1] into

$$Du = \nabla u\, \mathcal{L}^d + Cu + (u^+ - u^-) \otimes \nu_u \mathcal{H}^{d-1} \llcorner J_u, \tag{2.3}$$

where $\mathcal{L}^d$ is the Lebesgue measure, J_u is the jump set of u, where $u^+ \neq u^-$, i.e. the values at the boundary differ, ν_u the normal of the boundary and Cu a remainder Cantor part. In the following we will consider functions $u \in SBV(\Omega, \mathbb{R}^L)$, which is the space of functions for which $Cu = 0$ [6, 196]. We further define the perimeter of a measurable set $P \subset \Omega$, $\operatorname{Per}(\Omega, P)$, in turn by the total variation of its characteristic function $\chi_P : \Omega \to \mathbb{R}$ [53], the boundary of a set as $\partial S = \bar{S} \setminus \operatorname{int}(S)$, and the length of the boundary $\Gamma_{k,l} = \partial P_k \cap \partial P_l$ between two sets P_k, P_l via

$$|\Gamma_{k,l}| = \mathcal{H}^{d-1}(\Gamma_{k,l}), \tag{2.4}$$

where, again, $\mathcal{H}^{d-1}$ denotes the $d-1$-dimensional Hausdorff measure. These definitions allow us to examine the continuous boundary of shapes. Refer to Figure 2.2, where these continuous objects are marked in red.

2.3.2 Graph Discretization

We are interested in solving the continuous minimal partition problems (2.2) numerically. To do so we need to translate the problem into the discrete setting. To take a step from the continuous definitions to a discrete problem, we make use of the fact that we expect solutions u^* to the minimal partitioning problem to be piecewise constant with a finite number of pieces. A good discretization to a finite setting that mimics this piecewise constant structure exactly. *We hence represent the discretization by a graph of candidate constant sets, the nodes of which represent each separate constant piece and where neighboring pieces are connected by edges in the graph.* After solving the matching problem on this discrete graph, the final solution u^* can be reassembled by assigning to each constant piece its matched value according to the respective value of the node that represents it. This setup is sketched in Figure 2.2.

Note that this is a generalization of a classical discretization to a Cartesian grid. Placing a continuous function on a pixel grid corresponds to claiming that the function is piecewise-constant on every image pixel - hence the boundaries of the solution u^* to the minimal partitions problem will be a subset of the boundaries imposed by the image pixels.

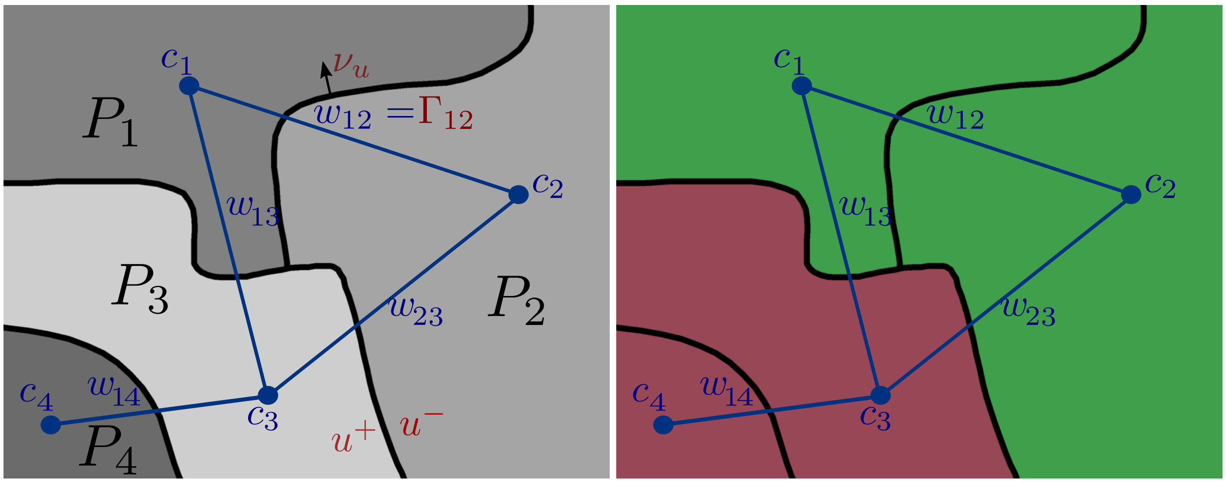


Figure 2.2: Sketch of the discretization process for a graph-based discretization. *Left:* The underlying continuous function $u \in SBV_\Pi(\Omega, \mathbb{R}^L)$ is pictured in black, with piecewise-constant partitions $\Pi = \{P_1, P_2, P_3, P_4\}$ shown in gray as well as the discrete graph structure in blue. *Right:* A minimal partitions problem with two potentials is solved on this graph structure. Pictured is the solution u^* which now corresponds to a piecewise constant solution (in red and green).

In slight generalization of the minimal partitions problem in (2.2) we now discuss the type of continuous functionals $F : SBV(\Omega, \mathbb{R}^L) \to \mathbb{R}$, that we want to represent discretely. Due to the discontinuities of SBV, we define different components of F on the continuous parts and the jump parts J_u of (2.3):

$$F(u) = \int_{\Omega \setminus J_u} \Phi(x, u(x), \nabla u(x))\, dx + \int_{J_u} \kappa\left(|u^+ - u^-|\right) |\nu_u| d\mathcal{H}^{d-1}, \tag{2.5}$$

where ∇, u^+, u^- refer to the decomposition detailed in (2.3). $\Phi : \Omega \times \mathbb{R}^L \times \mathbb{R}^{L\times d} \to \mathbb{R}$ is a function defined away from the jump set of u, while $\kappa : \mathbb{R} \to \mathbb{R}$ is a concave function measuring the jump penalty with $\kappa(0) = 0$ [53, 196]. We can consider the first term to be a generalized data term, and the second as a (jump)-regularizer.

To connect the continuous formulation of (2.5) to a discrete setting we define the discretization as a finite set of candidate sets $\Pi = \{P_i \subset \Omega \mid P_i \cap P_j = \emptyset,\ \forall j \neq i\}$ with $M = |\Pi|$ partitions. The continuous function $u \in SBV(\Omega, \mathbb{R}^L)$ is assumed to be constant on every partition, so that we can denote its value on partition $P_i \in \Pi$ by a vector $c_i \in \mathbb{R}^L$. Thus, $u(x) = c_i$ for every $x \in P_i \subset \Omega$. The partition Π can be represented by a set of nodes $V = \{1, \ldots, M\}$ where each node corresponds to a segment $P_i \in \Pi$. Furthermore we can describe every boundary between sets P_i and P_j as Γ_{ij} and by that define an edge set $E \subset V \times V$ as $E = \{(i,j) \in V \times V \mid |\Gamma_{ij}| > 0, i \neq j\}$. Note, that the perimeter of some partition $P_i \in \Pi$ is given by $\text{Per}_{P_i} = \sum_{(i,j)\in E} |\Gamma_{ij}|$.

Let us assume that our desired solution u^*, which minimizes (2.5), is piecewise constant. More formally, given some partition Π let us write $u \in SBV_\Pi(\Omega, \mathbb{R}^L)$ to denote continuous functions in SBV which are piecewise constant on the regions in Π, and assume $u^* \in SBV_\Pi(\Omega, \mathbb{R}^L)$. This implies that the jump set J_u is a subset of $\cup_{(i,j)\in E} \Gamma_{ij}$ and that $\Omega \setminus J_u$ is a subset $\cup_{i \in V} P_i$, or, in other words, the discrete partitioning by Π is able to represent the continuous structure of u^*.

Under the above assumption we can restrict the minimization of F over all functions $u \in SBV(\Omega, \mathbb{R}^L)$ to those in $SBV_\Pi(\Omega, \mathbb{R}^L)$ which allows to simplify (2.5) to a problem in which merely the values c_i inside the piecewise constant regions are the unknowns. Let us discuss the three main components of (2.5) separately.

Data Term:
Considering F for any $u \in SBV_\Pi(\Omega, \mathbb{R}^L)$ allows us to rewrite the first term of (2.5) as

(2.6) $$K(u) = \int_{\Omega \setminus J_u} \Phi(x, u(x), \nabla u(x))\, dx = \sum_{i=1}^{M} \int_{P_i} \Phi(x, c_i, 0)\, dx =: K_\Pi(c)$$

which is the discrete representation $K_\Pi(c) : \mathbb{R}^{M \times L} \to \mathbb{R}$ of this term that mere depends on the values c_i. For linear data terms such as in (2.2), i.e. $\Phi(x, u(x), \nabla u(x)) = \sum_{k=1}^{L} f_k(x) u_k(x) = f_k(x)(c_k)_i$ for $x \in P_i$, this is further simplified to

(2.7) $$K_\Pi(c) = \sum_{k=1}^{L} \sum_{i=1}^{M} (c_i)_k \int_{P_i} f_k(x)\, dx = \sum_{i=1}^{M} \langle c_i, \tilde{f}_k \rangle$$

with $\tilde{f}_k = \left(\int_{P_i} f_k(x) dx \right)_{k=1}^{L} \in \mathbb{R}^L$.

Regularization Term:
For the jump regularization, we can write $R(u)$ for any $u \in SBV_\Pi(\Omega, \mathbb{R}^L)$ as

(2.8) $$\begin{aligned} R(u) &= \int_{J_u} \kappa\left(|u^+ - u^-|\right)\, d\mathcal{H}^{d-1} = \sum_{(i,j) \in E} \int_{\Gamma_{ij}} \kappa\left(|c_i - c_j|\right)\, d\mathcal{H}^{d-1} \\ &= \sum_{(i,j) \in E} \kappa\left(|c_i - c_j|\right) \int_{\Gamma_{ij}} d\mathcal{H}^{d-1} = \sum_{(i,j) \in E} w_{ij}\; \kappa\left(|c_i - c_j|\right) =: R_\Pi(c) \end{aligned}$$

identifying the weights $w_{ij} = \int_{\Gamma_{ij}} d\mathcal{H}^{d-1} = |\Gamma_{ij}|$. With this weighting we can define the weighted finite graph $G = (V, E, w)$ as the discrete graph structure with which any continuous function $u \in SBV_\Pi(\Omega, \mathbb{R}^L)$ can be represented. Note that if κ is the identity, then $R_\Pi(c)$ is equivalent to graph total variation of c (cf. [106, 42]).

Constraint Set:
We are further carrying a constraint set when minimizing the minimal partitions problem. However, both constraints are pointwise and therefore straight forward to relate to constraints on c_i, i.e., the constraint set directly translates to

$$C_\Pi = \left\{ c \,\middle|\, (c_i)_k \in [0, 1],\ \sum_{k=1}^{L} (c_i)_k = 1, \forall i \right\}.$$

Interestingly, the above restriction from the minimization of F over $SBV(\Omega, \mathbb{R}^L)$ to its minimization over $SBV_\Pi(\Omega, \mathbb{R}^L)$ (which translates into the minimization of F_Π over $c \in C_\Pi$) remains valid as long as the jump set of the true solution is a subset of the jumps in the partition Π, independent of what exactly the "super" jump-set of Π is. Let us formalize this result:

Proposition 2.1. *Assume a discretization Π and its assorted partitions P_i to be given. Let u^* be a minimizer to the continuous problem (2.2) for given potentials f_k. If the jump-set J_{u^*} of u^* is a subset of the jump set of Π given as the boundaries $\cup_{(i,j) \in E} \Gamma_{ij}$, then*

$$\min_{u \in C} F(u) = \min_{c \in C_\Pi} F_\Pi(c),$$

for the discrete energy $F_\Pi = K_\Pi + R_\Pi$, i.e. the continuous minimum $F(u^*)$ is equal to the minimum $F_\Pi(c^*)$ of the discrete energy of F_Π under the constraints C_Π.

Proof. See appendix. ■

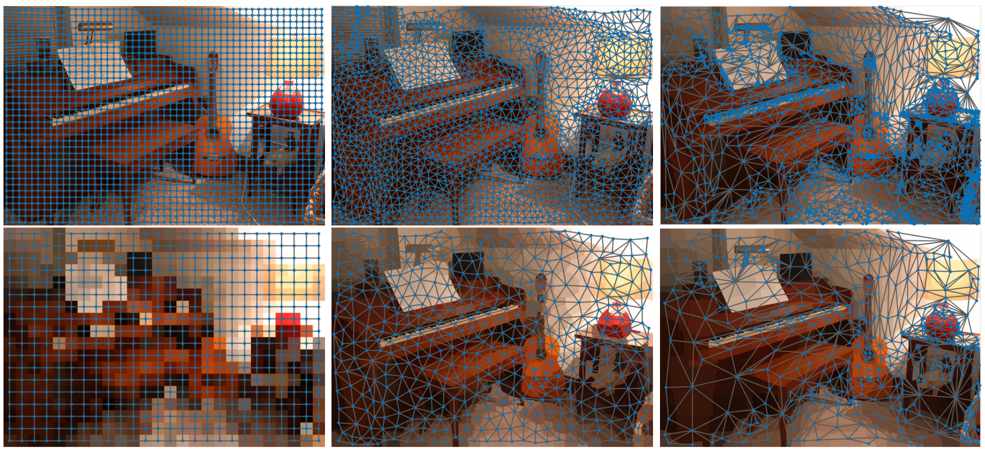


Figure 2.3: From left to right: Grid Sampling, SLIC Superpixels and L^0 Cut-Pursuit . Images from the Middlebury dataset [253]. The top row shows a fine discretization into the same number of nodes for every method, whereas the lower row shows a coarse discretization with the same number of nodes for every method.

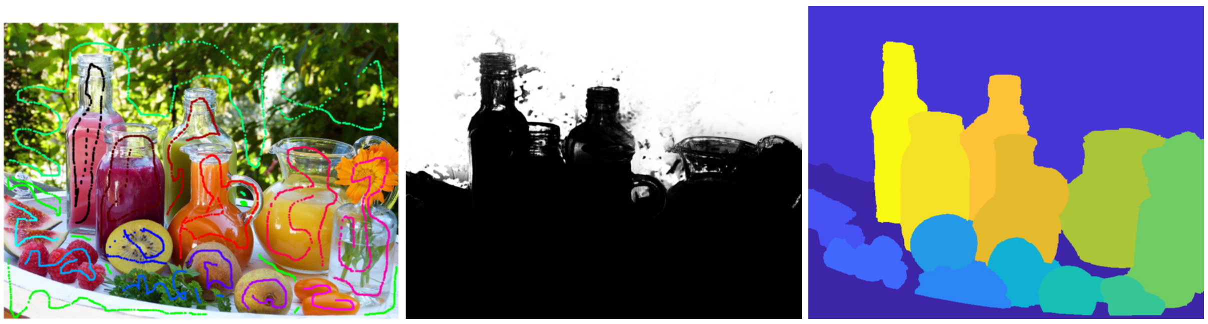


Figure 2.4: Illustrating the use of model-based segmentation methods: The user scribbles different objects to be segmented (left) from which a unary data term f is generated, e.g. by approaches like [207] or a pointwise neural network. As the unaries are insufficient for a good segmentation (illustrated by the fact that a thresholding of the unaries shown in the middle does not segment the background well), the proposed framework offers an efficient approach to obtain accurate segmentations as shown on the right.

Proposition 2.1 shows that if the jump set of u^* is contained in Pi, then the exact optimum u^* of the continuous problem can actually by found by computing a discrete solution c^* of the function F_Π numerically on a finite graph Practically however, we now need to find some partition Π that approximates (or ideally overestimates) the true jump set J_{u^*}, but consists of a limited number of segments. On a Cartesian grid, the equivalent operation is to subsample the image, result in the "superpixels" seen in Figure 2.3 on the left, which are not well aligned with edges in the images. However approaches such as SLIC (middle) or Cut-Pursuit (right, in the variant of [286]) are more adept at finding a superset of candidate partitions.

2.4 Numerical Evaluation

This section focuses on evaluating the proposed approach. We discuss examples in segmentation and stereo estimation. For segmentation we show a practical example, where the approach is used to align the output of a pixelwise neural network. We then follow up with a detailed comparison of graphs generated by SLIC, Cut Pursuit and subsampling.

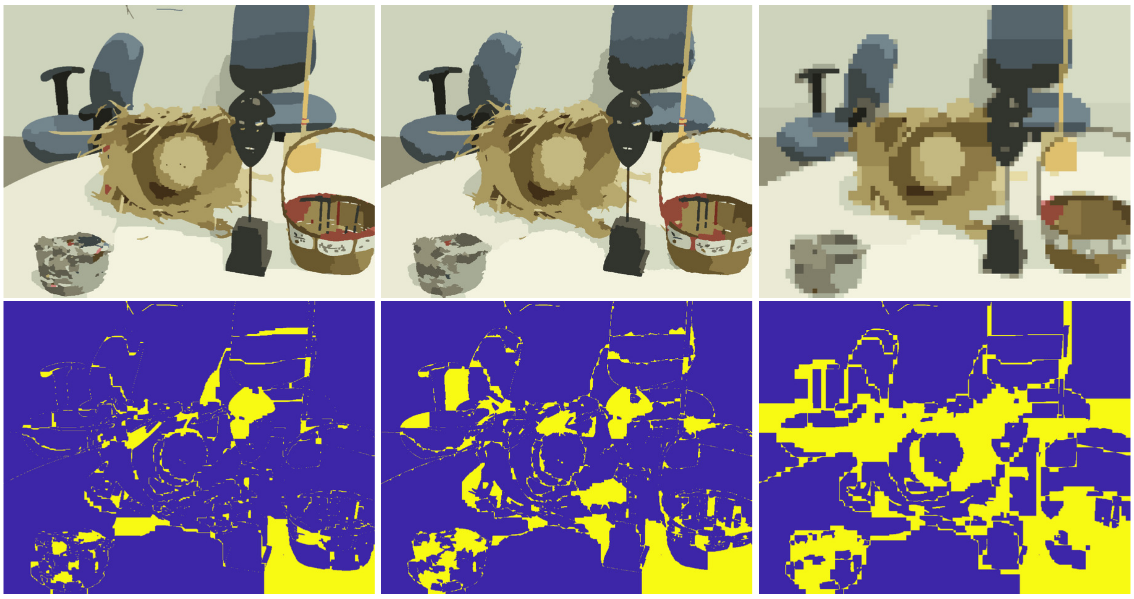


Figure 2.5: Qualitative comparison of matching quality for the example of image segmentation from a given set of pixel features. From left to right:L^0-CP graph, the SLIC graph, and a rectangular grid (right), all with the same number of vertices. The graphs are constructed as described in subsection 2.3.2. The top images show the final minimal partition result. The bottom images show the errors compared to the minimal partition computed on the full pixel grid, where yellow marks regions that are matched differently compared to the ground truth matching of the full image grid.

2.4.1 Segmentation

Multi-label segmentation is a central application of minimal partition problems, having been discussed in the continuous setting in works such as [57, 227] and widely studied in discrete methods such as [41, 152]. To apply multilabel segmentation we use the model described in (2.1), which can be recovered from (2.5) by setting $\kappa = \mathrm{Id}$ and choosing the L^1 norm for $|\cdot|$, leading to an anisotropic penalty of the jumps. We solve the discrete matching on the graph by a preconditioned primal-dual algorithm as discussed in [286]. Relaxed solutions are matched to corresponding partitions with maximal argument.

Use Case:
As one application scenario, imagine a user wants to segment an image by marking the objects to be segmented with scribbles, see Figure 2.4 on the left. Once the scribbling is complete we train a tiny pixelwise fully connected network on classifying the scribbled pixels correctly. The output of this network provides us with pixelwise features as shown in Figure 2.4 in the middle. Globally matching these features with 15 labels on the full grid requires 6 GB of memory, whereas the proposed approach reduces the memory requirements to 0.2 GB due to the graph construction and needs only 13% of computation time in total to yield the segmentation shown in Figure 2.4 on the right, which is precise enough to conduct various image manipulations such as inserting or removing some of the bottles or fruits.

Quantitative Analysis:
To analyze a wide range of images with canonical potentials, we turn to cartooning, i.e. multi-label segmentation with a fixed set of target colors chosen by a k-means selection. Figure 2.5 visualizes the result of the minimal partition problem for our graph discretization via L^0-CP, a graph constructed via SLIC, and a subsampling of the pixel grid, all with the same number of vertices. Checking the error maps on the bottom row of Figure 2.5 we see that both superpixel methods lead to solutions that closely match the solution at the finest level, while the subsampling is comparatively error-prone. The L^0-CP constructed discretization outperforms the SLIC-based discretization, due to its closer adherence to image edges.

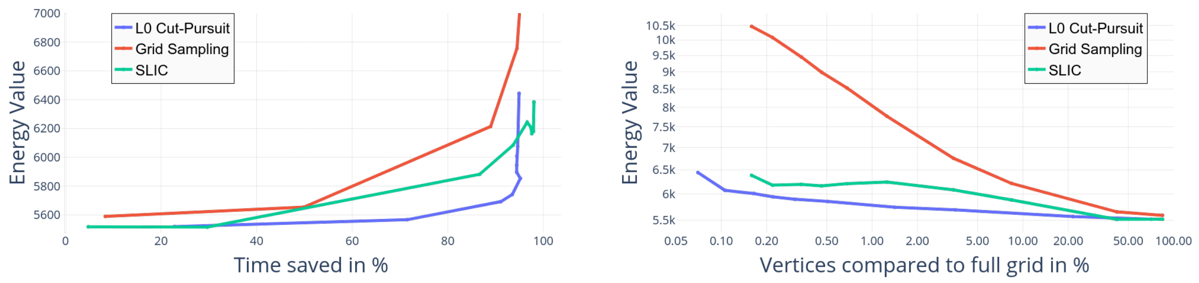


Figure 2.6: Computing the minimal partition on a well chosen graph discretization is much more efficient than computing it on the full grid. The y-axis in both plots denotes the energy value of the ground truth solution, which is computed on the full grid. Left: Time saved vs ground truth plotted vs the matching energy of the minimizer. Right: The number of nodes compared to the partition energy of the minimizer.

Ex.	Methods	Red. Rate	Time Saved	Mem.	Energy Offset
1	L^0-CP	$\mathbf{31\%}$	$\mathbf{79\%}$	$\mathbf{15MB}$	$\mathbf{0.73\%}$
	SLIC	41%	41%	16MB	1.18%
	sampling	74%	22%	32MB	5.77%
2	L^0-CP	$\mathbf{3.6\%}$	$\mathbf{87\%}$	$\mathbf{12MB}$	$\mathbf{3.5\%}$
	SLIC	8.42%	$\mathbf{87\%}$	32MB	6.29%
	sampling	8.42%	$\mathbf{87\%}$	27MB	13.33%
3	L^0-CP	$\mathbf{6.2\%}$	$\mathbf{83\%}$	$\mathbf{463MB}$	$\mathbf{2.1\%}$
	SLIC	14%	79%	1144MB	4.43%
	sampling	14%	82%	2976MB	3.79%

Table 2.1: Different scores for three examples. Shown are the ratio of time saved and the ratio of energy mismatch. The baseline method (a full image grid) uses 301, 684 and 6113 MB for each experiment.

Evaluating the gained efficiency in terms of time and in times of vertices in Figure 2.6 shows that this behavior leads to stable improvements over a wide range of graph discretization steps, energy values can be matched very closely using the superpixel-based graph discretization. In Table 2.1 we compare the three methods for three different examples. We find significant savings in runtime and memory, while staying close to the original energy, observing that graph-based discretization leads to a significant improvement in accuracy, compared to computing the segmentation on a downsampled grid, and further that using the L^0-CP superpixels leads to the most efficient final result, even though the computation of these superpixels itself takes more time than SLIC (the time/memory to compute superpixels and construct the graph is factored into all measurements we consider).

2.4.2 Stereo Matching

For the task of estimating disparities in stereo images the problem setting is different from that of segmentation tasks. While we want to reconstruct discrete labels for segmentation the estimated disparities between images live in a continuous range and therefore need dedicated treatment. The binarized vector structure of discrete segmentation labels ideally has to be translated to a continuously metricized label space. Assuming piecewise constant disparities in natural images it is still possible to transfer the graph reduction ideas to stereo estimation as proposed in [228] and the sublabel accurate setting of [197]. As the data term for stereo matching can be expressed as a cost-vector defined for each pixel, we need to find a sensitive scalar data function where superpixels can be computed to construct the graph. A natural

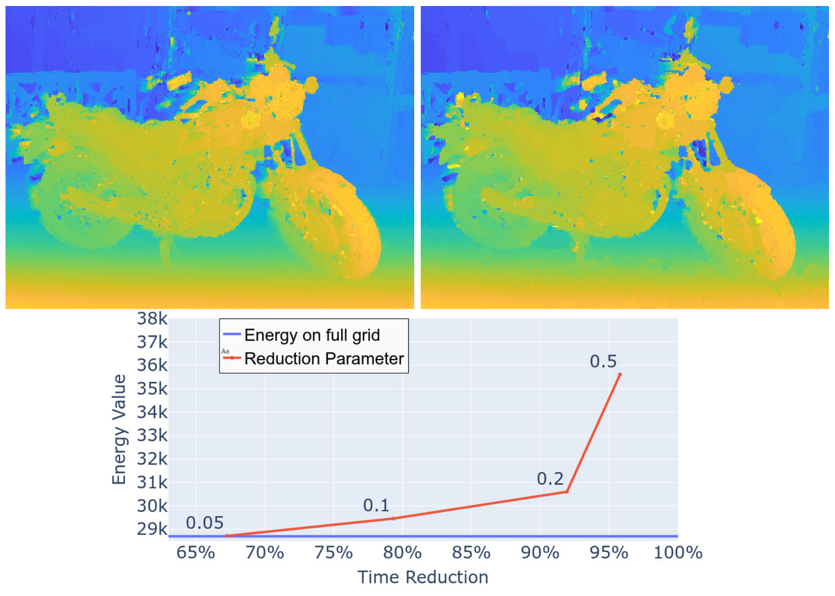


Figure 2.7: Results on stereo matching baselines. *Top:* Comparison of full (left) and proposed reduced (right) matching. Both methods use sublabels [197] between 32 labels. The proposed method uses Cut-Pursuit (with parameter $\alpha_c = 0.1$, a higher parameter corresponds to fewer vertices in the reduced graph) to find a reduced graph, amounting to a time reduction by a factor of 4.8, although the matching quality is near indistinguishable. *Bottom:* Time reduction and energy levels for different parameters α_c, showing the granular relationship between graph reduction and difference in energy value of the matching algorithm.

choice for such a function is the pixelwise minimizing argument of the data term. This is motivated by the intuition that constant regions of pointwise minimizing disparities likely induce constant regions of the original data term. The features f_k are either given directly as absolute pixelwise disparities, or as output of a stereo network and are then matched globally to combinations of candidate disparities in a given range. Figure 2.7 (top) gives a visual impression of the approximation behavior of the graph reduction on an exemplary stereo image. Figure 2.7 (bottom) visualizes the time vs. the achieved energy values of our method compared to the full stereo matching problem. Note the scale of the x-axis. We can easily reduce the necessary time and memory costs by using the graph-based discretization. Despite of the significant speedup for stereo matching the proposed method still is capable of producing visually pleasing results, as the matching is still computed with respect to all variables, just with an optimally chosen discretization.

2.5 Conclusions

In this work we presented strategies for the efficient realization of convex relaxations by directly moving from geometric properties of minimal partitions solutions to a graph-based discretization. We prove that such a graph-based discretization can be constructed in adherence to the global partitioning problem and implementing it on superpixel graphs yields accurate and efficient solutions in practice. We further find that using a superpixel approach that is more faithful to minimal surface energies, as the L^0-Cut Pursuit algorithm leads to more accurate solutions compared to SLIC and can be well worth the additional effort. We believe that the proposed methodology can facilitate the use of convex relaxation methods in practical applications, especially if input data is of high-resolution, where memory and computation constraints made these approaches previously infeasible.

Acknowledgements:
This research was supported by the German Research Foundation (DFG) under grant MO 2962/2-1.

The next sections contain the original appendix of publication [103].

2.A Proof of Proposition 1

We intend to show Proposition 1 by first showing a lemma for functionals of the form

$$F(u) = \int_{\Omega\setminus J_u} \Phi(x, u(x), \nabla u(x))\, dx + \int_{J_u} \kappa\left(|u^+ - u^-|\right) |\nu_u| d\mathcal{H}^{d-1}, \tag{2.9}$$

for $u \in SBV(\Omega, \mathbb{R}^L)$ under constraints C given as

$$C = \big\{u \bigm| u_k(x) \in [0,1],\ \sum_{k=1}^{L} u_k(x) = 1\big\}.$$

Lemma 2.2. *Assume a discretization Π and its assorted partitions P_i to be given. Let u^* be a minimizer to the continuous problem (2.9). If the jump-set J_{u^*} of u^* is a subset of the jump set of Π given as the boundaries $\cup_{(i,j)\in E}\Gamma_{ij}$, then*

$$\min_{u\in C} F(u) = \min_{c\in C_\Pi} F_\Pi(c),$$

for the discrete energy $F_\Pi = K_\Pi + R_\Pi$, i.e. the continuous minimum $F(u^*)$ is equal to the minimum $F_\Pi(c^*)$ of the discrete energy of F_Π under the constraints C_Π.

Proof. As defined in Section 3.2 of the main paper, we consider the space $SBV_\Pi(\Omega, \mathbb{R}^L)$ of functions in $SBV(\Omega, \mathbb{R}^L)$ that are piecewise-constant on partition Π. From the assumption that J_{u^*} is a subset of the jump set of Π, given by $\bigcup_{(i,j)\in E} \Gamma_{ij}$, we deduce $u^* \in SBV_\Pi(\Omega, \mathbb{R}^L)$. For a partition Π define

$$\Xi_\Pi = \{u \in C \mid u \in SBV_\Pi(\Omega, \mathbb{R}^L)\},$$

for

$$C = \big\{u \bigm| u_k(x) \in [0,1],\ \sum_{k=1}^{L} u_k(x) = 1\big\}.$$

Then $\Pi' \leq \Pi$, where "$\leq$" refers to the partial order on partitions meaning Π' is a finer partition than Π. This implies $\Xi_\Pi \subseteq \Xi_{\Pi'}$. For $u \in C_\Pi$ and the according c from Section 3.2 we already have shown $F_\Pi(c) = F(u)$.

Hence, $u^* \in \Xi_\Pi$ allows us to write

$$\begin{aligned} \min_{c\in C_\Pi} F_\Pi\,(c) &= \min_{\tilde u\in \Xi_\Pi} F(\tilde u) \\ &\leq F\,(u^*) = \min_{u\in C} F(u). \end{aligned}$$

Equality now follows due to $\Xi_\Pi \subseteq C$ from

$$\min_{u\in C} F(u) \leq \min_{\tilde u\in\Xi_\Pi} F(\tilde u) = \min_{c\in C_\Pi} F_\Pi\,(c)\,.$$ ■

Now we can find Proposition 2.1 as a simply corollary. The minimal partitions problem

$$\min_{\{u_k\}_{k=1}^L} \sum_{k=1}^{L} \int_\Omega -f_k(x) u_k(x) + \int_\Omega |Du_k|, \tag{2.10}$$

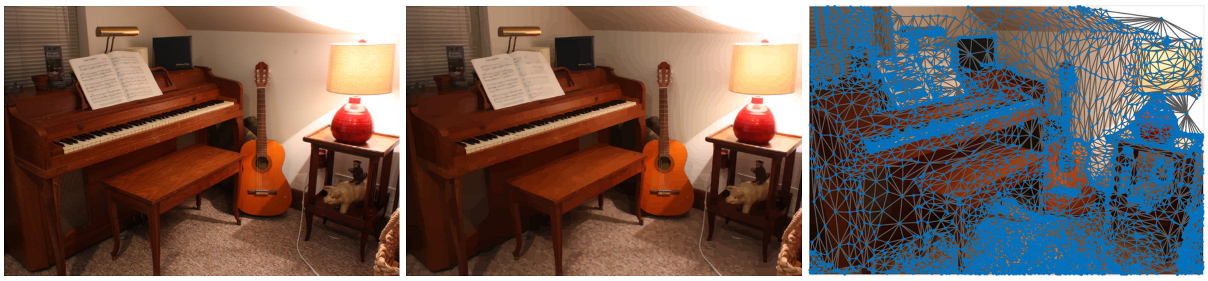


Figure 2.8: Reduction of the raw image (left) to 6031 nodes in the graph structure (Right). In the middle we reproject the graph structure onto the original image, visualizing the high fidelity of the representation, even as the number of nodes reduces to 0.45% of the full grid.

is a special case of (2.9) by choosing the data term via

$$K_\Pi(c) = \sum_{k=1}^{L}\sum_{i=1}^{M} (c_i)_k \int_{P_i} f_k(x)\,dx = \sum_{i=1}^{M} \langle c_i, \tilde{f}_k\rangle. \tag{2.11}$$

and setting $\kappa = \mathrm{Id}$ for the regularization term.

Proposition 2.1. *Assume a discretization Π and its assorted partitions P_i to be given. Let u^* be a minimizer to the continuous problem (2.2) for given potentials f_k. If the jump-set J_{u^*} of u^* is a subset of the jump set of Π given as the boundaries $\cup_{(i,j)\in E}\Gamma_{ij}$, then*

$$\min_{u\in C} F(u) = \min_{c\in C_\Pi} F_\Pi(c),$$

for the discrete energy $F_\Pi = K_\Pi + R_\Pi$, i.e. the continuous minimum $F(u^*)$ is equal to the minimum $F_\Pi(c^*)$ of the discrete energy of F_Π under the constraints C_Π.

Proof. Apply Lemma 2.2 to (2.2). ■

2.B Algorithmic Details

For implementation reference we replicate some parts of the L^0 Cut-Pursuit [159] variant of [286] in the continuous setting.

To obtain a good trade-off between having as few segments as possible but still constructing a partition whose jump set is a super set of the jump set of a minimizer u^*, we exploit a modification of the Cut-Pursuit (CP) algorithm of [159] discussed in [286]. In [159] Landrieu and Obozinski develop an approach to solve total variation problems [245, 45] with an alternating method solving graph cuts and reduced problems on the smaller graphs generated by these cuts. This is an efficient method with superior performance compared to more classical optimization methods as primal-dual [54] or Douglas-Rachford [90] algorithms minimizing the total variation problem. The Cut-Pursuit algorithm can be further extended to a variant minimizing the L^0 norm of the graph gradient. This strategy is able to quickly return approximate solutions to partitioning problems for a relatively high number of partitions compared to the number of potentials L we consider. The final number of partitions depends on regularization parameters and is data-dependent. In [286] Tenbrinck et. al. modify the Cut-Pursuit for L^0 by simplifying the algorithm to alternating between a graph cut and solving the data term separately on each generated partitions. We will denote this method as L^0-Cut-Pursuit (L^0-CP). We discuss this algorithm in a continuous setting with an L^2 data fidelity term, resulting in the following alternating algorithm: For a given function $u^k \in SBV(\Omega,\mathbb{R})$ and some given data $g \in L^1(\Omega,\mathbb{R})$, one step of the algorithm consists of the two alternating optimization steps. The first one is

$$B^{k+1} = \operatorname*{argmin}_{B\subset\Omega} \int_\Omega (u^k(x) - g(x))1_B(x)\,dx + \alpha_c \int_\Omega |D1_B|, \tag{2.12}$$

where 1_B denotes a characteristic function on B. This set minimization in (2.12) is binary and thus globally solvable. Then compute Π^{k+1} as the connected components of B^{k+1} and, in a second step, find the mean over every partition,

$$c_i^{k+1} = \frac{1}{|P_i|} \int_{P_i} g(x)\, dx. \tag{2.13}$$

From the values c_i^{k+1} of the partitions $P_i \in \Pi^{k+1}$, we can compute the continuous solution via

$$u^{k+1} = \sum_{P_i \in \Pi^{k+1}} c_i^{k+1}\, 1_{P_i}. \tag{2.14}$$

Note that such an algorithm operates on $SBV(\Omega, \mathbb{R})$ rather than $SBV(\Omega, \mathbb{R}^L)$ in order to be much more efficient, particularly for large L. In comparison to a naive subsampling on a grid (left) and the SLIC

Algorithm 2.1: L^0-Pursuit from [286]

```
Data: Image data $g$
$c^0 \leftarrow \mathrm{mean}(g)$
$\Pi^0 \leftarrow \{\Omega\}$
while $\Pi^k \neq \Pi^{k+1}$ do
    $B^{k+1} \leftarrow$ Solve (2.12) for given $u^k$
    $\Pi^{k+1} \leftarrow$ connected components of $B^{k+1}$
    $c^{k+1} \leftarrow$ Solve (2.13) for given $\Pi^{k+1}$
    $u^{k+1} \leftarrow$ Compute as in (2.14)
    $k \leftarrow k+1$
end
$\Pi \leftarrow \Pi^{k+1}$
Result: Discretization $\Pi$
```

superpixels from [2], the L^0-CP generates less uniformly-sized regions, allowing to combine large constant regions into a single node in a graph and thus being well suited for an efficient coarsification with accurate edges. Algorithm 2.1 shows the steps that this algorithm follows for further clarification.

2.B.1 Implementation

On a discrete image grid generated by sensor data, (2.12) becomes a binary partitioning on a discrete graph, which can be solved efficiently by a *maxflow* algorithm, e.g. Boykov-Kolmogorov [41]. In the end variants, such as L^1-Cut-Pursuit [159] or using a real-time Mumford-Shah such as [270] would also be possible candidates to find a candidate partition, yet we did not find these variants to yield either sufficient speed or sufficient accuracy around edges to be applicable - for algorithms that do not explicitly track the partitioning, the partition also has to be computed from the final result in an additional post-processing step.

When applying the Cut-Pursuit algorithm, we first need to consider which data will be used for g, the input to the Cut-Pursuit algorithm. A straightforward approach is to set

$$g(x) = \underset{k}{\operatorname{argmin}}\ f_k(x) \tag{2.15}$$

for the given label potentials, but especially for segmentation, using the color or grayscale image data directly is also reasonable under the assumption that piecewise constant objects in the RGB image correspond belong to separate labels, as done for algorithms such as SLIC.

We have chosen the `search-trees` implementation of [40] to solve the discrete version of the binary partition problem stated in (2.12). This can be done by reformulating the energy into a flow-graph structure with two additional terminal nodes *sink* and *source*. How to assign the right capacities to the edges can be taken from [159] or [286]. A significant bottleneck of this straightforward *maxflow* implementation is that the computation is difficult to parallelize. Thus, the computational time can increase drastically for very large images or other input data. For real-time applications with access to parallelization via GPUs or CPUs with sufficient cores we would recommend porting the entire pipeline into a single framework and using a primal dual algorithm with diagonal preconditioned stepsizes as in [226] not only for the minimal partitions problem but also the binary cuts. Especially running both subroutines on the GPU is potentially highly beneficial for large images. On the other hand, solving the binary cut with a primal-dual algorithm only approximates the solution in finite time and convergence criteria have to be chosen carefully to guarantee accurate results. In contrast *maxflow* termination criteria are straightforward, which is why we focus on *maxflow* in this work, aside from its applicability to weaker hardware with low specifications.

2.C Experimental Setup

We implement the graph-structured optimization of the convexified partition problem via a primal-dual algorithm [54] with diagonal preconditioning [226]. The preconditioning allows us to reconcile the step sizes of the algorithm with the varying sizes of the graph partitions. We use the implementation of this algorithm from https://github.com/tum-vision/prost, which conducts GPU computations with a Matlab wrapper. The L^0-CP implementation is written in Matlab using just the internal *maxflow* implementation. For the usecase study we use colors and coordinates as features to be classified via a 2-hidden-layer fully network with batchnorm and ReLU activations with 6 and 12 hidden neurons. Due to the tiny architecture and the few scribbles, the training of the network takes 18 seconds on a Laptop CPU (without fine-tuning the hyperparameters), and inferring pixelwise unaries on the entire 1440×1920 pixel image takes less than a second.

For the comparison of the inset table (Table 1) we consider three example images, 1. "cedar.bmp"[296], 2. "fish.jpg"[188], 3. "bin.png"[253], computed for a multi-label segmentation with $L = 16$ labels. *Reduction Rate* is the ratio between the number of segments to the full number of nodes. *Time save* describes the ratio of time that was saved by the graph discretization and *Energy offset* the ratio of energy mismatch. Note that we always denote the measured the time as the sum of the time used for the reduction method and the computational time to solve the label problem.

2.D Superpixel-Sublabel Stereo Lifting

When using a sublabel-based stereo estimation, then attention has to be paid to the treatment of the data term, as it does not bear the linear structure with to respect to the label coefficients anymore and hence interferes with the direct application of the proposed graph reduction. A closer look on the stereo problem formulation of [197] however reveals that the data term in between two neighboring labels is formed by calculating the convex envelope over the finitely sampled disparity costs. To be more precise, the stereo matching cost can be regarded as a in general non-linear data term $f_x\colon \mathbb{R}^L \to \mathbb{R}$ for each $x \in \Omega$. It directly operates on the lifted variable function $u\colon \Omega \to \mathbb{R}^L$. The data term is then relaxed to f_x^{**} for each x. This eventually amounts to a piecewise linear data term for each interval. Summing the data terms over superpixels, however, even yields a tighter convex approximation as

$$\sum_{i=1}^{M} \left(\int_{P_i} f_x dx \right)^{**} \geq \int_{\Omega} f_x^{**} dx, \tag{2.16}$$

where the left data term is the one effectively used when applying the method from [197] to superpixels as discussed.

2.E Further plots

Figure 2.8 shows the fidelity of the L^0-Cut Pursuit representation of an RGB image. The reduction 0.45% in comparison to the full grid is hardly noticable without zooming in. Figure 2.9 shows a variant of Figure 6 in the paper. We visualize PSNR / SSIM / DICE values for the cartooning problem. These are computed

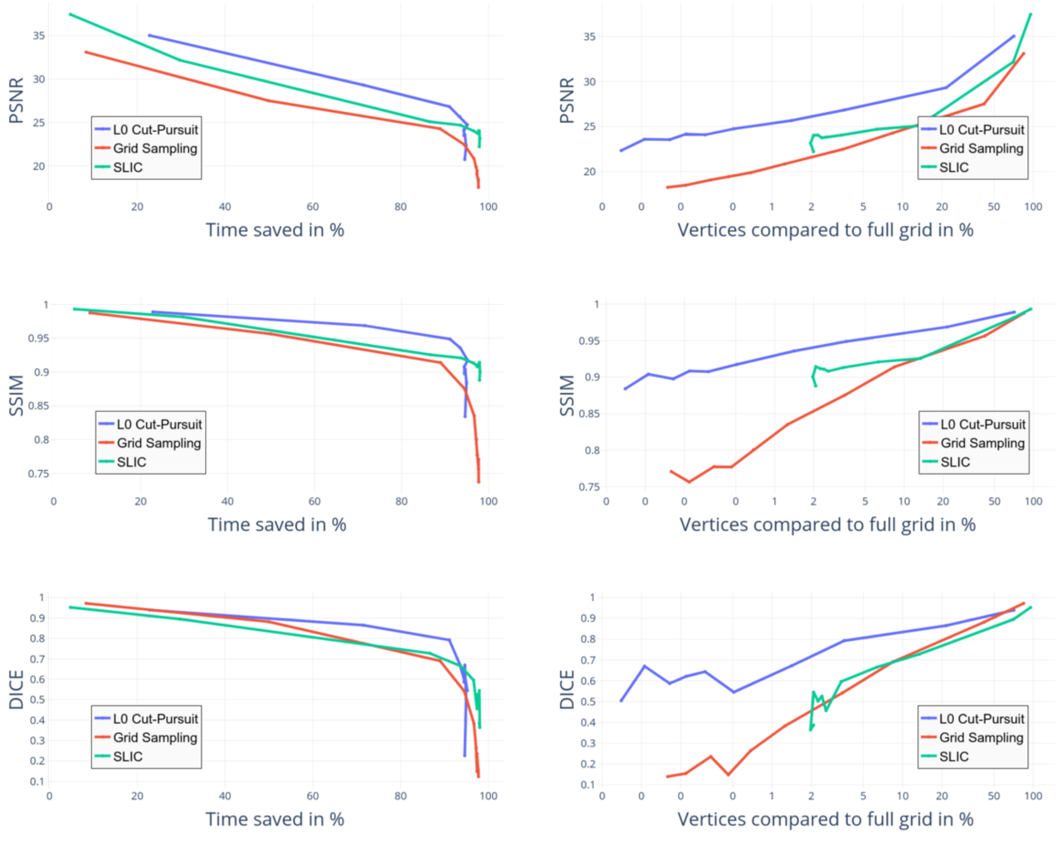


Figure 2.9: Computing the minimal partition on the graph is much more efficient than grid subsampling or SLIC superpixels. Left: Time saved vs 100% on the full grid plotted vs PSNR/SSIM/DICE score of the segmentation vs the full grid segmentation. Right: The number of nodes compared also compared to the PSNR/SSIM/DICE score of the segmentation.

by reassembling the output image from the piecewise constant segmentation and comparing it to the input image.

CHAPTER 3

Composite Optimization by Nonconvex Majorization-Minimization

Contextualization

This chapter reprints the publication "Composite Optimization by Nonconvex Majorization-Minimization", published as a journal publication with the SIAM Imaging journal with co-author Michael Moeller (Geiping & Moeller [2018] [104]). This chapter will focus on composite optimization as introduced in (1.3), centered on an application based on variational methods. The lifting approaches discussed in Chapter 2 can only be applied to a special class of non-convex optimization problems - free discontinuities - containing integrals over nonlinear functions acting on a point x and their boundary, $E(x) = \int_\Omega \nu(u, x(u), \nabla x(u)))\ du.$

However, applications such as Time-of-Flight reconstruction [150] lead to imaging conditions that require the inversion of an imaging operator, e.g. a blur operator, which acts on a neighborhood of points and thus breaks the above mentioned definition of a free discontinuity problem. Conceptually, this work can be understood as an iterative approximation to the task of optimizing nonconvex problems composed of both an outer operator and an inner free discontinuity problem. We construct a minimization-majorization strategy that builds a majorizer in each step which can in turn be solved by optimizing a subproblem that can be lifted. As this approach of a composite majorization-minimization algorithm with nonconvex majorizer was not yet contained in the literature, we analyze this setting in general and utilize insights from modern optimization theory to derive convergence properties for the case of arbitrary mappings of L-smooth adaptable functions $G : \mathbb{R}^m \to \mathbb{R}$, continuous functions $\rho : \mathbb{R}^n \to \mathbb{R}^m$ and $R : \mathbb{R}^n \to \mathbb{R}$ composited to

$$E(x) = G(\rho(x)) + R(x),$$

see also (1.3). Later chapters will contain a variety of bilevel optimization problems, which are a special case of composite optimization problems considered here. For bilevel problem, the complexity of ρ will however require specialized tools that we will develop in those chapters.

This project was conceived after long discussions about possible strategies to incorporate operators into lifting frameworks. Michael Moeller suggested the initial experiment of a nonlinear Jacobi iteration as iterative approach, which is now contained as motivating example in section 3.2, further provided support and crucial critical discussions for many proofs and definitions in this work, and contributed to the quality of writing and mathematical style of this work. Jonas Geiping introduced the generalization of this problem to L-smooth adaptable functions, defined the necessary conditions and assumptions, proved the statements on descent, slope bound and global convergence, provided the implementation details, visualizations and implemented the algorithm in Matlab for the synthetic examples and the application in Time-of-Flight Imaging. Additional unpublished appendices containing a continuous reformulation, and further details regarding the grid search employed in some examples, are provided in this thesis.

This work was further improved by anonymous SIAM Imaging reviewers who graciously helped to iron out consistency issues regarding properties of the reference function h and its domain.

Abstract

The minimization of a nonconvex composite function can model a variety of imaging tasks. A popular class of algorithms for solving such problems are majorization-minimization techniques which iteratively approximate the composite nonconvex function by a majorizing function that is easy to minimize. Most techniques, e.g. gradient descent, utilize convex majorizers in order to guarantee that the majorizer is easy to minimize. In our work we consider a natural class of nonconvex majorizers for these functions, and show that these majorizers are still sufficient for a globally convergent optimization scheme. Numerical results illustrate that by applying this scheme, one can often obtain superior local optima compared to previous majorization-minimization methods, when the nonconvex majorizers are solved to global optimality. Finally, we illustrate the behavior of our algorithm for depth super-resolution from raw time-of-flight data.

3.1 Introduction

Many imaging tasks that can be regarded as the minimization of some objective function, also called energy, can be solved by nonlinear optimization. Unfortunately, many energies arising from the faithful modeling of the data formation process and a state-of-the-art regularization term are inherently nonconvex, coupled, and high dimensional. Since determining the global minimizer of such a cost function is rarely feasible, one frequently turns to (gradient-based) methods that only find a, possibly sub-optimal, critical point of the energy landscape [205].

Interestingly, some high-dimensional nonconvex optimization problems do admit a global solution within reasonable time. Besides problems for which the solution can be determined analytically, the aforementioned class includes *separable* problems on a bounded domain, i.e. problems for which the minimization of an energy E with respect to some variable $u \in \mathbb{R}^n$ decomposes into the minimization of separate low-dimensional energies, e.g. $E(u) = \sum_{i=1}^n E_i(u_i)$. Even more remarkably, there are several types of non-separable nonconvex optimization problems which can be reformulated as convex problems, e.g. via convex relaxation techniques [56] or via functional lifting [228], and still yield a globally optimal solution to the original nonconvex problem. Unfortunately, the aforementioned techniques rely on a special structure of the objective. Even seemingly minor perturbations of the required structure make it impossible to exploit these techniques, and lead practitioners to consider local (gradient-based) methods again.

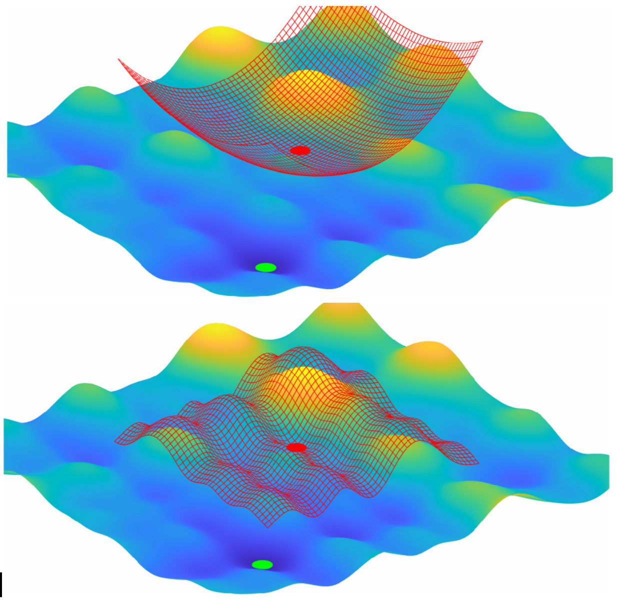
Interestingly, many of such local methods admit an interpretation in the framework of *majorization-minimization* techniques: In each iteration, the energy E is approximated by a simpler function E_{u^k} which satisfies

$$\begin{aligned} E_{u^k}(u) &\geq E(u), \\ E_{u^k}(u^k) &= E(u^k), \end{aligned}$$

for u^k being the current iterate. By defining the next iterate to be the minimizer of the approximation E_{u^k},

$$u^{k+1} = \arg\min_u E_{u^k}(u),$$

one automatically obtains monotonically decreasing objective values.



[]

[]

Figure 3.1: Nonconvex versus convex majorization. (a) shows an energy of type (3.2) with a convex majorizer. (b) shows the same energy, but with a solvable nonconvex majorizer. The initial point is marked in red, the global minimum of the energy in green. We can see that the shown nonconvex majorizer can better represent the given function.

Common gradient-based methods use simple convex approximation functions E_{u^k}, e.g. quadratic functions,

$$E_{u^k}(u) = E(u^k) + \langle \nabla E(u^k), u - u^k \rangle + \frac{1}{2\tau}||u - u^k||^2, \tag{3.1}$$

in the case of gradient descent. While this leads to easy-to-solve subproblems, such approximation functions E_{u^k} are only a crude approximation of the original energy and almost all information about the shape of the original energy landscape is lost.

In this work we propose a novel majorization-minimization technique with nonconvex functions E_{u^k} with the idea to

1. approximate the original energy landscape much more faithfully, and
2. still be able to minimize E_{u^k} globally by considering functions E_{u^k} that are either separable or can be minimized via relaxation techniques.

As illustrated in a simple two-dimensional example in Figure 3.1, one can expect a more faithful approximation of the original energy to yield 'better' local minima: While the para-bolic approximation of section 3.1 yields a nearby local minimum, the separable nonconvex majorizer in section 3.1 allows to skip several local minima. In this example, the minimizer of the nonconvex majorizer is in a close vicinity to the global minimizer after just a single step of the algorithm.

While our motivation comes from the (somewhat heuristic) idea of finding 'better' local minima, our convergence analysis does *not* depend on the subproblems being solved to global optimality. For the remainder of the paper we consider the minimization of composite energies of the form

$$E(u) = G(\rho(u)) + R(u), \tag{3.2}$$

for suitable functions $G : \mathbb{R}^m \to \mathbb{R}$, $\rho : \mathbb{R}^n \to \mathbb{R}^m$ and $R : \mathbb{R}^n \to \mathbb{R}$, via the iterative minimization of

$$E_{u^k}(u) = G(\rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k) \rangle + R(u) + \frac{1}{2\tau} ||\rho(u) - \rho(u^k)||^2. \tag{3.3}$$

The model function E_{u^k} is a naturally global, but nonconvex, majorizer of E for suitable τ as we will see later. A typical example for 'simple' functions $\rho : \mathbb{R}^n \to \mathbb{R}^n$ and $R : \mathbb{R}^n \to \mathbb{R}$ is given when both functions are separable, i.e. $\rho(u) = (\rho_1(u_1), \dots, \rho_n(u_n))$ and $R(u) = \sum_{i=1}^n r_i(u_i)$. In this case, the nonconvex majorizer (3.3) is then also separable and can be solved in each dimension separately.

We continue summarizing some of the related work for nonconvex and composite optimization problems and illustrate how the proposed majorization-minimization technique (3.3) differs from the methods that have been considered in the literature so far.

3.1.1 Related Work

The current field of nonlinear optimization is quite wide. In the following overview of related work we focus on results, that like our method do not require convexity of the objective function and we limit ourselves to generalizations of first-order methods. The general framework of majorization-minimization methods has been reviewed widely in the literature of the recent decades, see for example, [127, 186, 273, 298].

The first option for tackling the minimization of (3.2) is to ignore the composite structure of $G \circ \rho$, naturally leading to schemes like the aforementioned gradient descent (GD) (3.1) or the closely related forward backward splitting (FBS) [61, 25, 206]. As we will see in more detail below, the proposed scheme recovers such algorithms in the special case of ρ being the identity. The convergence[1] of a general class of nonconvex first-order descent methods, including GD and FBS, was shown e.g. in [11]. It is important to note that such a convergence is nontrivial for arbitrary nonconvex functions and requires, for example, some algebraic notion of 'tameness' [131], that is nevertheless usually present in practice.

The most limiting assumption in these first-order methods is the Lipschitz continuity of the gradient of $F = G \circ \rho$, the first part of the objective function. This class of problems was recently extended in [18, 37, 27] to L-smooth adaptable functions , these functions are not necessarily convex or L-smooth, only a Legendre function h must exist, so that $Lh - F$ is convex for some $L > 0$. The previously mentioned methods can be extended to a descent 'relative' to these Legendre functions. Defining the Bregman distance of h as $D_h(u, v) = h(u) - h(v) - \langle \nabla h, u - v \rangle$, [37]'s majorizer can be written as

$$E_{u^k}(u) = \langle \nabla F(u^k), u - u^k \rangle + R(u) + \frac{1}{\tau} D_h(u, u^k). \tag{3.4}$$

They show that the sequence of iterates generated by this type of majorizer converges for appropriate τ and conditions to h, F and R, which include the KŁ-property [33] (which follows from the mentioned notion of 'tameness') and the assumption that $\operatorname{dom} h = \mathbb{R}^n$.

We can relate [37] to the earlier approach of [71]. Here, the functions h are restricted to induced norms, however they are allowed to change during the sequence of iterations, $h^k = \frac{1}{2} || \cdot ||^2_{A^k}$ where each A^k is a symmetric positive definite matrix. These matrices are chosen so that (3.4) is a majorizer of E at u^k, which is in turn guaranteed if $D_{h^k - F}(u, u^k) \geq 0$. This is a weaker assumption than $h - F$ convex, which is equivalent to $D_{h-f}(u, v) \geq 0$, but limited by the use of induced norms. [71] also shows global convergence under the KŁ-property.

Recent works have also proposed general frameworks for iteratively replacing the original minimization problem with simple approximation functions E_{u^k} beyond majorization-minimization. [88] analyzes approximation functions E_{u^k}, satisfying $|E_{u^k}(u) - E(u)| \leq \omega(||u - u^k||)$ for a proper growth function ω. A minimization scheme of these approximation functions exhibits subsequential convergence to critical points, even if the subproblem evaluations are inexact. These approximation functions need not necessarily be convex, but the distance of their subsequent evaluations must tend to zero. A slightly different

[1]In the context of first order methods, we consider 'convergence' as implying that the sequence of iterates converges to a stationary point of the objective function.

generalization is discussed in [214], where approximation functions constructed by $E_{u^k} = \bar{E}_{u^k} + D_h(u, u^k)$ with $|\bar{E}_{u^k}(u) - E(u)| \leq \omega(||u - u^k||)$ are examined. Here ω is a growth function and D_h a Bregman distance generated by a Legendre function, generalizing the previously discussed (3.4). Subsequence convergence can again be shown here, under relatively weak conditions. However the approximation function E_{u^k} is taken to be convex in [214] to, among other properties, guarantee the success of a backtracking scheme and reach an implementable algorithm.

A review of Majorization-Minimization methods that still allow for a sequence of iterates to converge globally under the KŁ-property can be found in [34]. There, majorizers E_v are required, most prominently, to be m-strongly convex and to fulfill the abstract descent inequality $\operatorname{dist}(0, \partial E_v(u)) \leq c||v - u||$. This condition however, will be difficult to fulfill in our setting due to the presence of ρ, and we will thus seek convergence under different conditions.

Coming to related work in composite optimization we find that there are two ways to handle problems of type (3.2): Either we linearize the outer function G in each approximation, or the inner function ρ. Linearizing the inner function ρ leads to methods that are reminiscent of classical Levenberg-Marquardt algorithms for nonlinear least-squares problems. The approximation function can be written as

$$E_{u^k}(u) = G\left(\rho(u^k) + J_\rho(u^k)(u - u^k)\right) + R(u) + \frac{1}{2\tau}||u - u^k||^2, \tag{3.5}$$

where J_ρ denotes the Jacobian of ρ. A classical application for this composition are systems of nonlinear equations. Due to the inner linearization, it is in general not required that G is smooth. Subsequence convergence follows as a result of [171, 88] or [214]. Global convergence for convex G and $R = 0$ is shown under the KŁ-property in [222]. Further literature can be found under the terms 'prox-linear' or 'prox-descent', e.g. [171, 89]. Linearizing the outer function leads to algorithms related to iterative re-weighting procedures:

$$E_{u^k}(u) = G(\rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k) \rangle + R(u) + \frac{1}{2\tau}||u - u^k||^2. \tag{3.6}$$

Subsequence convergence follows from the general result of [88] under the assumption that the distance of subsequent iterates tends to zero. Further analysis, related to special cases in iterative re-weighting can be found in [213] or under more general assumptions, but including the convexity of E_{u^k} in [214]. The connection to iterative reweighting is immediate for concave G, as then τ can be taken arbitrarily large and the proximal term vanishes. This formulation is closely related to our work and differs from ours in the way we measure the distance to the previous iterate. We later discuss the implications of this difference.

As a first visualization, Figure 3.2 and Figure 3.3 show these majorization functions in two dimensions. For a nonconvex function of type (3.2) in subsection 3.1.1, a gradient descent majorizer is shown in subsection 3.1.1 and a forward-backward splitting in subsection 3.1.1. We see that for both majorizers their respective minimizers, marked in green, are located in a close neighborhood to the current iterate, marked in red. Both algorithms will likely converge to a nearby local minimum of the original energy subsection 3.1.1.

The presented related majorizers for composite optimization are shown in Figure 3.3. Subsection 3.1.1 and Subsection 3.1.1 show both linearization variants, namely (3.5) and (3.6). These generally produce more faithful representations of the original energy (subsection 3.1.1), but both minimizers are still far away from the global minimum. Finally, Subsection 3.1.1 shows our majorizer (3.3). Note that the minimizer of this majorizer can not only be computed efficiently due to its separability, but also allows for a global view of the function and its minimizer almost coincides with the global minimum although the initial point is quite far from it.

Finally, a recent preprint [36] proposes to solve composite minimization problems with a different approach, namely a nonlinear splitting variant, reformulating the problem to

$$\min_{u,v \in \mathbb{R}^n} G(v) + R(u) \quad \text{s.t. } \rho(u) = v, \tag{3.7}$$

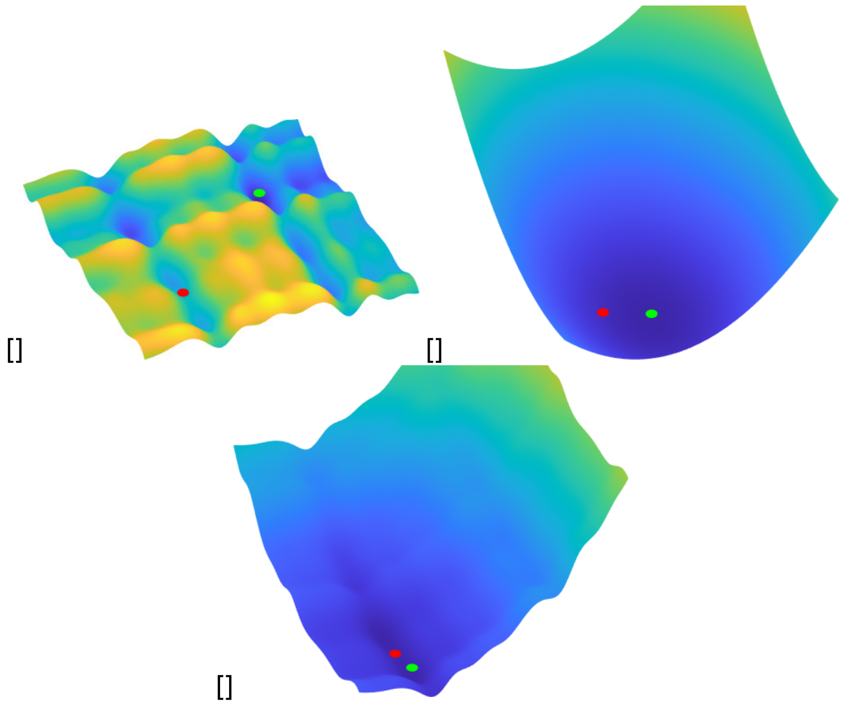


Figure 3.2: Visualization of related work. (a) shows the original function of type (3.2), (b) shows a gradient descent majorizer (3.1), (c) shows a forward-backward splitting majorizer (3.4). The point u^k is equal in each figure and shown in red and the minimizer u^{k+1} in green.

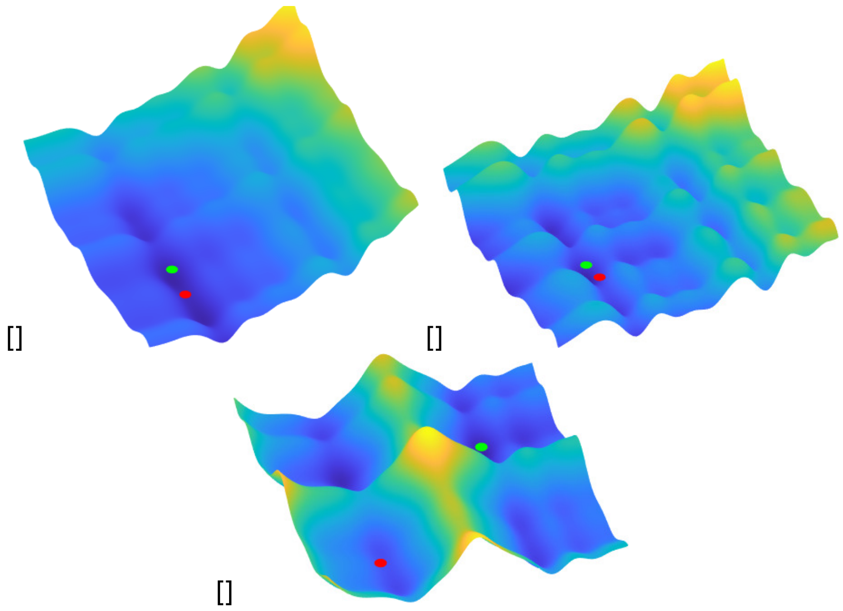


Figure 3.3: Visualization of related work. (a) shows a prox-linear type inner linearization (3.5), (b) shows an outer linearization (3.6), (c) shows finally shows the proposed (separable) majorizer (3.3). The point u^k is equal in each figure and shown in red and the minimizer u^{k+1} in green.

and introducing an augmented Lagrangian formulation

$$\min_{u,v\in\mathbb{R}^n} G(v) + R(u) + \langle w, \rho(u) - v\rangle + \frac{\tau}{2}||\rho(u) - v||^2. \tag{3.8}$$

with an additional variable $w \in \mathbb{R}^n$ that mimics the dual variable of the convex setting. This is a quite interesting result, as it shows that the complementarity of forward-backward splitting and augmented Lagrangian methods can be extended into the composite setting. Whereas our method is a generalization of forward-backward splittings, their work generalizes ADMM [99]. Critically both ours and their approach rely on the efficient solution of a nonlinear programming task as intermediate step in the algorithm. For us, this is the nonconvex majorizer (3.3), the corresponding problem in [36, Eq. (6.3)] is the minimization of (3.8) for u:

$$u^{k+1} = \arg\min_u G(v^{k+1}) + R(u) + \langle w^k, \rho(u) - v^{k+1}\rangle + \frac{\tau^k}{2}||\rho(u) - v^{k+1}||^2 + \frac{\mu}{2}||u - u^k||^2. \tag{3.9}$$

Both subproblems are in general equally difficult as they are connected for $\mu = 0$, identifying $v^{k+1} = \rho(u^k)$ and $w^k = \nabla G(\rho(u^k))$.

Although formulated in less generality in terms of the involved functions but in more generality in terms of the number of nested functions, the update equation of the related work [98, Eq. (11)] for solving problem (3.2) can be written as

$$u^{k+1} = \arg\min_u R(u) + \frac{1}{2}\|\rho(u) - \rho(u^k)\|^2 + \sigma\langle\nabla G(\rho(u^k)), \rho(u) - \rho(u^k)\rangle + \frac{1}{2\tau}\|u - u^k\|^2,$$

for an affine linear ρ. This is similar to the proposed algorithm but also contains the additional proximity term for $u - u^k$. The analysis we provide in this work could make it interesting to revisit [98] in the two-layer case.

Solving the subproblems via lifting. While the convergence analysis of our approach will make rather weak assumptions on the global quality of the solution used in each of the subproblems (3.3), we found our method to be particularly effective and successful if the (nonconvex) subproblems are solved to global optimality. This raises the question what types of functions allow to determine globally optimal solutions to such subproblems.

A rather simple case occurs if the involved functions are separable or separable into blocks of few variables. In these situations we can apply exhaustive search and branch-and bound algorithms to each block separately [143, 115].

More interesting for imaging tasks is the class of functions where the subproblems can be solved by *functional lifting*. It was shown in [228, 53] that free discontinuity-type energies, in particular,

$$E(u) = \int_\Omega \nu(x, u(x), \nabla u(x))\, dx, \quad u \in W^{1,1}(\mathbb{R}^n, \mathbb{R}) \tag{3.10}$$

with $\nu : \mathbb{R} \times \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ being continuous in its second argument, and convex and continuous in its third argument, can be optimized globally by *lifting* the problem into a higher dimensional space where it admits a convex representation. Recent works, e.g. [197, 196], discuss how to discretize the continuous formulation accurately and return to the finite-dimensional setting of this work.

We therefore expect to be able to solve all nonconvex majorizers E_{u^k} that are a discretization of (3.10) to (near)-global optimality, allowing us to consider highly non-trivial majorizing functions. Similar relaxation methods exist in the discrete community via graph cuts for Markov random fields, see [152, 41, 133] and the references therein.

3.1.2 Organization of this work

This work introduces an optimization algorithm for the sum of a function and a composite function, which iteratively minimizes a nonconvex majorizing function (3.3). The algorithm is detailed and discussed in section 3.2 and basic properties are discussed in the first part of section 3.3. The second part of section 3.3 then extends these basic properties to a global convergence under the KŁ-property and uniqueness of R-minimizing solutions. Several generalizations and implementation details follow in section 3.4. Finally, section 3.5 shows some promising numerical results on synthetic examples where the proposed algorithm is able to find better minima than competing first-order methods, while being much more efficient than methods from global optimization applied to the discussed problem class (3.2). We then close section 3.5 with an application to depth super resolution from noisy time-of-flight data.

3.2 The General Principle

Before we begin the formal introduction of the necessary context and provide convergence and basic properties in their full generality it is instructive to reduce the problem formulation to a very simple test case.

Let us consider the standard Jacobi-iteration:

$$u^{k+1} = D^{-1}(f - (A-D)u^k) \tag{3.11}$$

which solves the linear equation $Au = f$ for symmetric $A \in \mathbb{R}^{n\times n}$ whose diagonal is D. We can interpret this scheme as successively minimizing the function

$$E_{u^k}(u) = \langle u, \frac{1}{2}Du + (A-D)u^k - f\rangle - \langle u^k, \frac{1}{2}(A-D)u^k\rangle, \tag{3.12}$$

which is a majorizer to $E(u) = \frac{1}{2}\langle u, Au - f\rangle$, if $D - A$ is positive definite. Now we would like to solve the nonlinear equation system $A\rho(u) = f$ for some function $\rho : \mathbb{R}^m \to \mathbb{R}^n$. And we do the same as before and apply our previous majorizer to $\rho(u)$:

$$u^{k+1} = \arg\min_u \langle \rho(u), \frac{1}{2}D\rho(u) + (A-D)\rho(u^k) - f\rangle - \langle \rho(u^k), \frac{1}{2}(A-D)\rho(u^k)\rangle. \tag{3.13}$$

If ρ is separable, then these problems can still be solved efficiently in each dimension, thereby iteratively solving $A\rho(u) = f$. As we will see in more detail in Example 3.1, (3.13) is a particular instance of the algorithm we propose and study in this paper, yielding nonconvex majorizers that are still easy to minimize. While this illustrates the main idea of our algorithm, the situation becomes even more interesting if an additional regularization R makes a substitution like $z = \rho(u)$ impossible.

3.2.1 The Algorithm

Now we are ready to formulate the algorithm in full generality.

We consider the task of minimizing functions $E : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\} =: \overline{\mathbb{R}}$ and define the domain of E by $\operatorname{dom} E =: \{u \in \mathbb{R}^n \mid E(u) < \infty\}$. We denote the closure of this domain by $\overline{\operatorname{dom} E}$. A function is proper if $\operatorname{dom} E \neq \emptyset$. We call a function lower semi-continuous if we have $\liminf_{u\to\bar{u}} E(u) \geq E(\bar{u})$ for all $\bar{u} \in \operatorname{dom} E$. The distance of a vector $u \in \mathbb{R}^n$ to a subset S of $\mathbb{R}^n$ is defined via $\operatorname{dist}(u, S) = \inf_{x\in S} ||u - x||$. We denote the pre-image of a mapping ρ on a set S by $\rho^{-1}(S)$. A proper function is essentially smooth if its convex subdifferential ∂h is locally bounded and single-valued on its domain [20] or equivalently if $\operatorname{dom} \partial h = \operatorname{int} \operatorname{dom} h \neq \emptyset$ [240, Thm 26.1].

We consider the optimization problem

$$\min_{u\in\Delta} E(u) = G(\rho(u)) + R(u), \tag{3.14}$$

minimizing the composite and additive model E over a closed set defined via $\Delta = \rho^{-1}(C)$ for a closed convex set $C \subset \mathbb{R}^m$ with $\operatorname{int} C \neq \emptyset$. We employ a convex function h that mirrors the geometry of the problem and mimics the behavior of G. We make the following assumptions on these functions:

Basic Assumptions:

- $h : \mathbb{R}^m \to \overline{\mathbb{R}}$ is a proper, lower semi-continuous, convex function that is essentially smooth with $\overline{\operatorname{dom} h} = C$,
- $G : \mathbb{R}^m \to \overline{\mathbb{R}}$ is a proper, lower semi-continuous function with $\operatorname{dom} h \subset \operatorname{dom} G$, which is differentiable on $\operatorname{int} \operatorname{dom} h$
- $R : \mathbb{R}^n \to \overline{\mathbb{R}}$ is a proper, lower semi-continuous function and $\operatorname{dom} R \cap \rho^{-1}(\operatorname{int} \operatorname{dom} h) \neq \emptyset$.
- $\rho : \mathbb{R}^n \to \mathbb{R}^m$ is a continuous function.

Under these assumptions, E is a proper, lower semi-continuous objective function. We define the Bregman distance of two vectors $u \in \mathbb{R}^m$ and $v \in \operatorname{int} \operatorname{dom} h \subset \mathbb{R}^m$ relative to the chosen function h by

$$D_h(u, v) = h(u) - h(v) - \langle \nabla h(v), u - v \rangle.$$

and we set $D_h(u, v) = \infty$ if $v \notin \operatorname{int} \operatorname{dom} h$. We choose a step size $\tau > 0$ to be discussed later, a starting vector $u^0 \in \rho^{-1}(\operatorname{int} \operatorname{dom} h)$, and then apply the following iterative scheme:

Main Algorithm:

(3.15) $$u^{k+1} \in \operatorname*{arg\,min}_{u \in \mathbb{R}^n} \frac{1}{\tau} D_h(\rho(u), \rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k) \rangle + G(\rho(u^k)) + R(u)$$

Dicussions of well-definedness and convergence will also follow later in section 3.3. The use of a Bregman distance is an immediate generalization of the usual squared norms, e.g via $h(u) = \frac{1}{2}||u||_2^2$, which allows us a greater level of generality, as we will discuss later in section 3.4.

Example 3.1. *Returning to the Jacobi example from before, we now see in particular that setting* $G(v) = \frac{1}{2}\langle v, Av - f \rangle$, $R(u) = 0$ *and* $h(u) = \frac{1}{2}||u||_D^2$ *exactly recovers the nonlinear Jacobi updates in* (3.13).

In practice this algorithm is applicable even if the subproblems (3.15) can only be solved up to a local optimum. However it is especially interesting if (3.15) can actually be solved globally. In our applications we mainly consider three interesting cases for this, although our theoretical analysis in later chapters is not necessarily limited to those.

First, if ρ and R are Lipschitz and separable, in the sense that $\rho : \mathbb{R}^n \to \mathbb{R}^n$ can be written as $\rho(u) = (\rho_1(u_1), \dots, \rho_n(u_n))$ and $R : \mathbb{R}^n \to \overline{\mathbb{R}}$ can be written as $R(u) = \sum_{i=1}^n r_i(u_i)$, then (3.15) decomposes into one-dimensional subproblems for each u_i. We use separable $h(u) = \sum_{i=1}^m h_i(u_i)$, so that $D_{h_i}(u_i, v_i) = h_i(u_i) - h_i(v_i) - h_i'(v_i)(u_i - v_i)$ and find that the majorizer decouples so that

(3.16) $$u_i^{k+1} \in \operatorname*{arg\,min}_{u_i} \frac{1}{\tau} D_{h_i}(\rho_i(u_i), \rho_i(u_i^k)) + \frac{\partial G(\rho(u^k))}{\partial u_i}(\rho_i(u_i) - \rho_i(u_i^k)) + r_i(u_i).$$

These univariate nonconvex problems can be solved very efficiently and in parallel by uniform grid searches or more elaborate exhaustive branch-and-bound strategies, due to the Lipschitz properties R and ρ whenever R has a bounded domain.

A particularly interesting and practically relevant case are energies of the form

(3.17) $$E(u) = \sum_{i=1}^m F_i\left(\sum_{j=1}^n \rho_{ij}(u_j)\right) + \sum_{i=1}^n r_i(u_i),$$

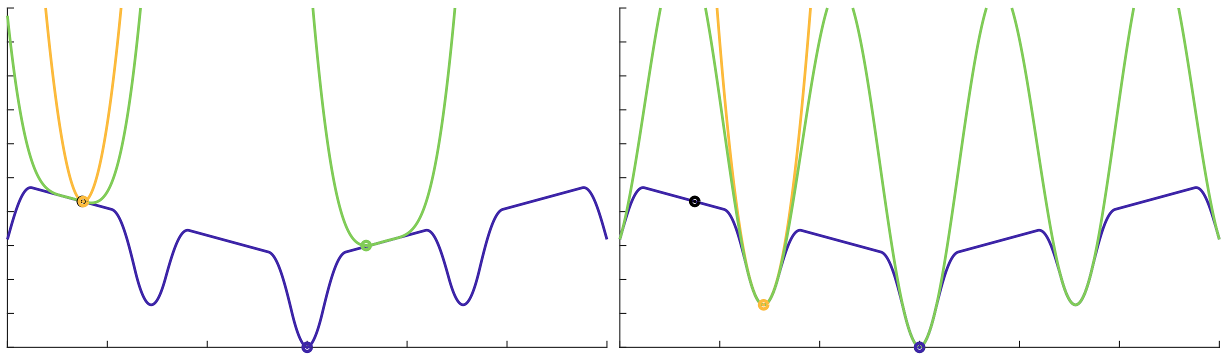


Figure 3.4: Proximity relative to ρ can be crucial during minimization. Initialization marked in black, global minimum in dark blue. Our majorizer, i.e. (3.15) and its minimizer are marked in green. A majorizer that measures proximity relative to u, i.e. (3.6), shown in yellow. The left figure shows a single step, the right figure shows the algorithm output and final majorizers.

where we have $\rho_{ij} : \mathbb{R} \to \mathbb{R}$ and $r_i : \mathbb{R} \to \overline{\mathbb{R}}$, $F_i : \mathbb{R} \to \overline{\mathbb{R}}$ and we again assume a bounded domain. These models appear naturally in several nonlinear regression tasks. But again, the problem can be decomposed into one-dimensional subproblems and we apply our algorithm, as the subproblems decouple if we set $G(v) = \sum_{i=1}^m F_i(\sum_{j=1}^n v_{ij})$ and $\rho = (\rho_{11}, \dots, \rho_{mn})$.

Remarkably, both of the above examples still yield (near)-globally solvable subproblems, if the separable regularization is replaced by a suitable penalty on the gradient of the unknown. While such subproblems are nonconvex and non-separable they can still be solved efficiently with the lifting techniques discussed in the context of equation (3.10). We detail these types of problems in subsection 3.4.3.

3.2.2 Special Cases

We note several cases, where the method reduces to simpler approaches: First, if ρ is the identity, then we immediately recover a non-composite problem, the setting of [37]. If ρ is invertible, then we can minimize over z with the regularizer $R(\rho^{-1}(z))$ and again recover a non-composite problem. Further, if G is separable as well, then it would be easier to take the whole problem directly as a nonconvex majorizer, which would converge in a single iteration. If the regularizer R is zero, then the algorithm works fine, yet we would like to highlight that it is possibly easier to solve the minimization over $G(v)$ first under the constraint of $v \in \rho^{-1}(\operatorname{int}\operatorname{dom} h)$ (for separable ρ, h this would be an especially easy constraint), and then optimize $D_h(\rho(u), v^*)$.

3.2.3 Proximity relative to the inner function

Unlike standard schemes, (3.15) measures the proximity between $\rho(u)$ and $\rho(u^k)$, instead of u and u^k as in previous works on composite optimization [89, 88, 39, 214]. However our choice, motivated by the nonlinear Jacobi example previously mentioned, is advantageous, whenever the subproblems can still be solved efficiently.

The main advantage is the leverage we gain. By updating relative to ρ, we are able to directly apply smoothness properties and subsequent descent lemmas for G, easily finding a global majorizer in each step. Furthermore, our step size can now be chosen analytically independent of ρ and all new iterates are feasible in the sense that $u^{k+1} \in$ dom ρ and $\rho(u^{k+1}) \in$ dom h.

To make a more intuitive argument, we also note that penalizing the direct proximity between u and u^k of course limits the updates u^{k+1} to a neighborhood of u^k. If we are able to solve subproblems to global optimality, then limiting our updates in a local area seems unnecessary. If we penalize the proximity in ρ, then we only stay in a local area relative to G, which is necessary, as we linearized G. But otherwise we allow for arbitrarily large updates as long as $\rho(u)$ is similar to $\rho(u^k)$, which is of no issue, as we solve our subproblems globally. By this approach we hope to find interesting stationary points globally and not just locally in a neighborhood around the starting vector.

Figure 3.4 visualizes this behavior in 1D. Given G, a smooth version of $\min(u^2, \lambda)$, $\rho(u) = \sin(u)$ and $R(u) = \alpha|u|$ we majorize around the black mark using $h(u) = u^2$. We see that the proximity relative to ρ is critical for reaching the global minimizer.

It is quite instructive to compute both update steps for a linear composition example, i.e. $E(u) = F(Au)$ for $A \in \mathbb{R}^{m\times n}$. One can check that the updates relative to $\rho_{ij}(u_j) := a_{ij}u_j$ in (3.17) then correspond to a gradient descent with diagonal preconditioning, whereas the update in u directly would correspond to standard gradient descent.

3.3 Algorithm Discussion and Convergence

In the following section we will analyze convergence properties of the proposed algorithm. We will specify the assumptions we make on G, discuss well-posedness of subproblems and give a descent lemma. We will then make further assumptions on tameness of the function and uniqueness of R-minimizing solutions to prove global convergence.

During this discussion we will move toward the exact structure of the main algorithms (3.15) in three steps, the first two being variants, where we first only assume that the subproblems (3.15) are solved 'sufficiently' and then only assume that the subproblems are solved 'sufficiently' to a stationary point. We do this to highlight precisely when global solutions to the subproblems are necessary and what advantages this confers; knowing that for some problems, solving the nonconvex subproblem to global optimality, might be too difficult.

3.3.1 Basic Properties

To find fixed step sizes for the algorithm we need to assume some bound on the change in the gradient of the function. An appropriate generalization of Lipschitz continuity that gives a bound "relative" to the chosen function h [18, 37], defines the following property:

Definition 3.2 (L-smooth adaptable). *A proper, lower semi-continuous function $G : \mathbb{R}^m \to \overline{\mathbb{R}}$ is called L-smooth adaptable relative to a convex function h if there exist $L > 0$ so that $Lh - G$ is convex on $\operatorname{int}\operatorname{dom} h$.*

As a consequence of the L-smooth adaptability property we have the following descent inequality:

Lemma 3.3 (Descent Lemma, [18, Lemma 1]). *If the proper lower semi-continuous function G is L-smooth adaptable relative to an essentially smooth convex function h so that $\operatorname{dom} h \subset \operatorname{dom} G$ and G is differentiable on $\operatorname{int}\operatorname{dom} h$, then*

$$G(z) - G(w) - \langle \nabla G(w), z - w\rangle \leq LD_h(z, w) \quad \forall z, w \in \operatorname{int}\operatorname{dom} h.$$

Proof. $D_f(z, w) \geq 0\, \forall z, w \in \mathbb{R}^n$ if and only if f is convex. Hence $D_{Lh-G}(z, w) \geq 0$ which yields $D_G(z, w) \leq LD_h(z, w)$, due to the additivity of the Bregman distance. ■

We define the subproblem energy in the following by

$$E_{u^k}(u) = \frac{1}{\tau}D_h(\rho(u), \rho(u^k)) + G(\rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k)\rangle + R(u). \tag{3.18}$$

for some u^k so that $\rho(u^k) \in \operatorname{int}\operatorname{dom} h$. For the first part of this section, we now make the following assumptions:

Assumptions A:

- Our basic assumptions from subsection 3.2.1 hold,

- E is bounded from below,
- E is coercive,
- $Lh - G$ is convex on $\operatorname{int}\operatorname{dom} h$,
- Every iteration is solved sufficiently, so that $E_{u^k}(u^{k+1}) \leq E_{u^k}(u^k) = E(u^k)$ and $u^{k+1} \in \rho^{-1}(\operatorname{int}\operatorname{dom} h)$.

Under these assumptions we will discuss the validity of (3.18) as a majorizer for E and the well-posedness of the minimization of $E_{u^k}(u)$. Note that in practice, we often gain coerciveness by considering functions E with a bounded domain. The most important assumption here is the smoothness assumption on G given by its L-smooth adaptability. The fifth assumption is very general and holds, for example, already when each sub-problem is solved only by finite sampling. We also need the technical assumption that $\rho(u^{k+1}) \in \operatorname{int}\operatorname{dom} h$, which holds e.g. if $\operatorname{dom} h = \mathbb{R}^m$ or if R is convex and ρ is continuously differentiable. It can also be guaranteed through a set of constraint qualifications arising from [241, 10.6,10.9], yet we omit further discussion of this issue.

Lemma 3.4 (Majorization Property). *Under the assumptions A, given some $\tau \leq \frac{1}{L}$ and $\rho(u^k) \in \operatorname{int}\operatorname{dom} h$, $E_{u^k}(u)$ is a majorizer of E, i.e. it fulfills the properties*

- $E_{u^k}(u) \geq E(u) \quad \forall u \in \mathbb{R}^n$
- $E_{u^k}(u^k) = E(u^k)$.

Proof. A quick computation shows that $E_{u^k}(u^k)$ is equal to $E(u^k)$, as the Bregman distance $D_h(x, y)$ is zero if $x = y$ and $x, y \in \operatorname{int}\operatorname{dom} h$. Now, using Lemma 3.3 for G and inserting arbitrary $\rho(u) \in \operatorname{int}\operatorname{dom} h$ and $\rho(u^k) \in \operatorname{int}\operatorname{dom} h$ gives

$$G(\rho(u)) \leq G(\rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k) \rangle + L D_h(\rho(u), \rho(u^k)).$$

On the other hand, we have, due to $\frac{1}{\tau} \geq L$,

$$E_{u^k}(u) \geq L D_h(\rho(u), \rho(u^k)) + G(\rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k) \rangle + R(u).$$

Combining both inequalities gives the desired result for any $u \in \mathbb{R}^n$ s.t. $\rho(u) \in \operatorname{int}\operatorname{dom} h$. If $\rho(u) \notin \operatorname{int}\operatorname{dom} h$, then $E_{u^k}(u) = \infty$, so that the inequality is trivially fulfilled. ■

Now let us consider the set of minimizers of E_{u^k} (3.18):

$$M_\tau(u^k) = \left\{ \bar{u} \in \Delta \mid \bar{u} \in \arg\min_u E_{u^k}(u) \right\} \tag{3.19}$$

Lemma 3.5. *Under the assumptions A, the set $M_\tau(u^k)$ is non-empty and compact for any $u^k \in$ dom E if $\tau \leq \frac{1}{L}$.*

Proof. We already know that E is coercive. However, as E_{u^k} is a majorizer for $\tau \leq \frac{1}{L}$, $E_{u^k}(u) \geq E(u)$, it is itself coercive. Furthermore E and hence E_{u^k} is bounded from below. As a result $E_{u^k}(u)$ is lower semi-continuous and proper with bounded level sets. [241, Theorem 1.9] now guarantees that $\inf E_{u^k}$ is finite and that the set $M_\tau(u^k)$ is non-empty and compact. ■

3.3.2 Descent Properties

As usual for majorization-minimization algorithms, we gain a monotone decrease in the objective function:

Lemma 3.6 (Descent Lemma). *If the assumptions A hold, then the energy E is monotonically decreasing for all iterates u^k if $\tau < \frac{1}{L}$ is chosen as step size. The descent rate is*

$$E(u^{k+1}) - E(u^k) \leq -\frac{1-\tau L}{\tau} D_h(\rho(u^{k+1}), \rho(u^k))). \tag{3.20}$$

Proof. Using Lemma 3.3 for G we insert $\rho(u^{k+1})$ and $\rho(u^k)$ so that

$$G(\rho(u^{k+1})) - G(\rho(u^k)) \leq \langle \nabla G(\rho(u^k)), \rho(u^{k+1}) - \rho(u^k)\rangle + L D_h(\rho(u^{k+1}), \rho(u^k)).$$

Furthermore, because every iteration is solved sufficiently, we know that

$$\frac{1}{\tau} D_h(\rho(u^{k+1}), \rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u^{k+1}) - \rho(u^k)\rangle + R(u^{k+1}) + G(\rho(u^k)) \leq G(\rho(u^k)) + R(u^k).$$

Adding both inequalities yields

$$G(\rho(u^{k+1})) - G(\rho(u^k)) + R(u^{k+1}) - R(u^k) \leq (L - \frac{1}{\tau}) D_h(\rho(u^{k+1}), \rho(u^k)),$$

which is the desired result. Due to the convexity of h, the right-hand side is always non-negative if $\tau < \frac{1}{L}$. ■

From this we gain convergence in function values, subsequence convergence and some notion of convergence speed. For clarity of presentation we define the "outer" sequence $z^k := \rho(u^k)$.

Corollary 3.7. *Under the assumptions A, we see that for a step size of $\tau < \frac{1}{L}$,*

- *the sequence of function values $(E(u^k))_{k=1}^\infty$ converges to a limit E^*,*
- $\lim_{k\to\infty} D_h(z^{k+1}, z^k) = 0$,
- *there exist converging subsequences $(u^{k_l})_{l=0}^\infty$ and $(z^{k_l})_{l=0}^\infty$,*
- *the sequence $\min_{1\leq k\leq N} D_h(z^{k+1}, z^k)$ converges to 0 with order $\mathcal{O}(\frac{1}{N})$.*

Proof. Summing both sides of the descent inequality in (3.20) for $k = 1, \ldots, N$ and simplifying the expression gives

$$\alpha \sum_{k=1}^N D_h(z^{k+1}, z^k) \leq E(u^1) - E(u^N) \leq C$$

for $\alpha = \frac{1-\tau L}{\tau} > 0$. $(E(u^k))_{k=1}^\infty$ is a monotone decreasing sequence, that is bounded as $-\infty < \inf E \leq E(u^k) \leq E(u^1)$ and thus converging. We gain the existence of converging subsequences u^{k_l} due to these bounds and the lower semi-continuity and coercivity of E. The continuity of ρ allows us to extend this to the existence of converging subsequences z^{k_l}.

Concerning the convergence rate, define a minimal proximity over all iterates

$$\mu_N =: \min_{1\leq k\leq N} D_h(z^{k+1}, z^k),$$

so that

$$\sum_{k=1}^N D_h(z^{k+1}, z^k) \leq \frac{E(u^1) - E(u^N)}{\alpha}$$

implies

$$\mu_N \leq \frac{E(u^1) - E(u^N)}{\alpha N} \leq \frac{C}{N}.$$

■

For later use we define the set of all accumulation points of the sequence u^k, generated by our algorithm for a given starting vector u^0 as

(3.21) $$\text{accum}(u^0) = \{u \in \mathbb{R}^n \mid \lim_{l \to \infty} u^{k_l} = u \text{ for a subsequence } u^{k_l} \text{ of } u^k\}.$$

This set is non-empty as the sequence is bounded and closed as a set of limit points.

3.3.3 Convergence Properties

In this section we want to prove further statements of convergence. Up to now, we only gave assumptions on G and on the relative minimization of the subproblems. For arbitrary ρ and R we can thus not expect a global convergence of the sequence of iterates, u^k, mostly because the accuracy up to which each subproblem is solved has not been specified yet. However, even if the subproblems are solved exactly, we need to assume some algebraic properties of E.

We will first give an appropriate optimality condition for our subproblems and specify the algebraic notion of 'tameness' discussed previously. Under these assumptions we will show a global convergence of the sequence $z^k = \rho(u^k)$ and convergence to critical points. This is a natural convergence result as the distance of successive iterates is only measured relative to $D_h(\rho(u), \rho(u^k))$. It can be a conscious modeling choice to allow several equivalent critical points u^* to be found by a single run of the algorithm. However we will also see that the choice of regularizer R directly controls a global convergence in u^k, if the subproblems are solved to global optimality.

From now on, we consider a limiting subgradient:

Definition 3.8 (Subgradients [241, 8.3]). *A function $E : \mathbb{R}^n \to \overline{\mathbb{R}}$ has the subgradient $v \in \mathbb{R}^n$ at a point $\bar{u} \in$ dom E, if*

$$\liminf_{\substack{u \to \bar{u}, \\ u \neq \bar{u}}} \frac{E(u) - E(\bar{u}) + \langle v, u - \bar{u} \rangle}{||u - \bar{u}||} \geq 0$$

and we write $v \in \hat{\partial} E(\bar{u})$. v is further an element of the limiting subgradient $\partial E(\bar{u})$ at $\bar{u}$ if sequences exist so that $u^i \to \bar{u}$, while $E(u^i) \to E(\bar{u})$ and $v^i \to v$ for $v^i \in \hat{\partial} E(u^i)$.

Note that in our case there exists some $\bar{u}$ for every $u^k \in \rho^{-1}(\text{int dom}\, h)$ so that $\partial E_{u^k}(\bar{u})$ is non-empty due to Lemma 3.5 and Fermat's rule [241, 10.1]. Rockafellar's optimality condition for limiting subgradients over a set [241, 8.15] is a necessary condition for local minima. For our needs we consider the following version:

Lemma 3.9 (Optimality Condition). *If assumptions A hold and ρ is continuously differentiable, then a local minimum of (3.14) at $\bar{u} \in \rho^{-1}(\text{int dom}\, h)$ implies that*

(3.22) $$-J_\rho(\bar{u})^* \nabla G(\rho(\bar{u})) \in \partial R(\bar{u}),$$

Proof. This follows from [241, 8.15] and [241, 8.8], as the constraint $u \in \rho^{-1}(\text{dom}\, h)$ is not active for $\bar{u} \in \rho^{-1}(\text{int dom}\, h)$. ■

We further call the set of all points $\bar{u}$ that fulfill this condition crit$E$.

However considering just the subdifferential of arbitrary functions leaves too many pathological cases for successful analysis of global convergence properties [87]. We thus follow recent literature on nonconvex optimization and consider functions that further satisfy the Kurdyka-Łojasiewicz property:

Definition 3.10 (Nonsmooth Kurdyka-Łojasiewicz property [32]). *For a proper and lower semi-continuous function $E : \mathbb{R}^n \to \overline{\mathbb{R}}$, we define its local Kurdyka-Łojasiewicz property (KŁ) at a point $\bar{u} \in \operatorname{dom} E$ by the attribution that there exist $\eta > 0$, $\varphi \in C^0[0,\eta) \cap C^1(0,\eta)$ with $\varphi(0) = 0$, φ concave, $\varphi' > 0$ and a neighborhood $U(\bar{u})$, so that*

$$\varphi'\left(E(u) - E(\bar{u})\right) \operatorname{dist}\left(0, \partial E(u)\right) \geq 1,$$

for all $u \in U(\bar{u})$ with $E(\bar{u}) < E(u) < E(\bar{u}) + \eta$.

If E is for example semi-algebraic, then it satisfies the KŁ-property at any $\bar{u} \in \operatorname{dom} E$. A proper semi-algebraic function $E : \mathbb{R}^n \to \mathbb{R}$ has a finite number of critical points [87]. We note that any function definable in an o-minimal structure satisfies the KŁ-property [33]. Further, the property can be uniformized to yield

Lemma 3.11 ([35, Lemma 6]). *Let Ω be a compact set and consider a proper, lower semi-continuous function $E : \mathbb{R}^n \to \overline{\mathbb{R}}$. If E is constant on Ω and satisfies the KŁ-property at every point in Ω, then there exist $\varepsilon > 0$, $\eta > 0$, $\varphi \in C^0[0,\eta) \cap C^1(0,\eta)$ with $\varphi(0) = 0$, φ concave, $\varphi' > 0$ such that for all $\bar{u}$ in Ω the uniformized KŁ-property,*

$$\varphi'\left(E(u) - E(\bar{u})\right) \operatorname{dist}\left(0, \partial E(u)\right) \geq 1,$$

holds for all $u \in \mathbb{R}^n$ with $dist(u, \Omega) < \varepsilon$ and $E(\bar{u}) < E(u) < E(\bar{u}) + \eta$.

Now we are ready to collect our set of assumptions.

Assumptions B:

- The function E is continuous on its domain and satisfies the KŁ-property at every point in the set accum(u^0),
- h is m-strongly convex on $\operatorname{int}\operatorname{dom} h$,
- $\rho : \mathbb{R}^n \to \mathbb{R}^m$ is continuously differentiable,
- $\frac{1}{\tau}\nabla h - \nabla G$ is locally Lipschitz continuous on $\operatorname{int}\operatorname{dom} h$,
- Given the set $Z = (z^k)_{k=1}^{\infty}$, we require that $\bar{Z} \subset \operatorname{int}\operatorname{dom} h$
- every iteration satisfies $0 \in \partial E_{u^k}(u^{k+1})$ and uses a step size $\tau < \frac{1}{L}$.

These extended assumptions now allow us to prove the following statements:

1. The sequence of $z^k = \rho(u^k)$ converges globally to a value z^*.
2. All accumulation points of subsequences $(u^{k_l})_{i=1}^{\infty}$ are stationary points of E.
3. There is a correspondence between the limit point z^* and the accumulation points of the iterates, given by $z^* = \rho(u^*)$ for all $u^* \in$ accum(u^0).

We will see in the proof that the most demanding properties in assumptions B are used to prove a bound on the slope of iterates, i.e. the inequality $\operatorname{dist}\left(0, \partial E(u^{k+1})\right) \leq c||z^{k+1} - z^k||$ for some $c > 0$. Norm convergence to limit points lying on the boundary of $\operatorname{dom} h$ from arguments involving the KŁ-property is problematic, as the essential smoothness of h implies that $||\nabla h(y^k)|| \to \infty$ for $y^k \to y^* \in (\operatorname{dom} h \setminus \operatorname{int}\operatorname{dom} h)$ [20] so that there will be no fixed bound c. By requiring $\bar{Z} \subset \operatorname{int}\operatorname{dom} h$ we strengthen the assumption of $z^k \in \operatorname{int}\operatorname{dom} h$ from assumption A to the assumption that any prospective limit point z^* will also fulfill $z^* \in \operatorname{int}\operatorname{dom} h$. In comparison to [37], the assumption $\operatorname{dom} h = \mathbb{R}^m$ given therein is a straightforward implication of our more technical statement. Our assumption on the continuity of E and replaces their assumption that $E_{z^k}(z^{k+1}) \leq E_{z^k}(z) \, \forall z \in \operatorname{int} \operatorname{dom} h$ in this subsection.

The ingredients of our proof follow recent literature, e.g. [11, 37], however special care has to be taken as all estimates of slope and objective value are only relative to the outer sequence of $z^k = \rho(u^k)$.

Lemma 3.12 (Slope bound). *If the assumptions A and B hold, then $c < \infty$ exists, so that*

$$\text{dist}(0, \partial E(u^{k+1})) \leq c||z^{k+1} - z^k|| \quad \forall k \in \mathbb{N}. \tag{3.23}$$

Proof. Consider the optimality condition of the update equation, as all subproblems are solved exactly:

$$0 \in \partial R(u^{k+1}) + (J_\rho(u^{k+1}))^* \left(\nabla G(\rho(u^k)) + \frac{1}{\tau} \nabla h(\rho(u^{k+1})) - \frac{1}{\tau} \nabla h(\rho(u^k)) \right)$$

and reformulate to

$$\begin{aligned}(J_\rho(u^{k+1}))^* \Big(&\nabla G(\rho(u^{k+1})) - \nabla G(\rho(u^k)) \\ &- \frac{1}{\tau} \nabla h(\rho(u^{k+1})) + \frac{1}{\tau} \nabla h(\rho(u^k)) \Big) \in \partial R(u^{k+1}) + (J_\rho(u^{k+1}))^* \left(\nabla G(\rho(u^{k+1})) \right).\end{aligned}$$

Now we see that the left hand side is an element of $\partial E(u^{k+1})$, so that we can estimate its norm by

$$\text{dist}(0, \partial E(u^{k+1})) \leq \|J_\rho(u^{k+1})^*\|_{\text{op}} \left\| \nabla \left(\frac{1}{\tau} h - G \right) (z^{k+1}) - \nabla \left(\frac{1}{\tau} h - G \right) (z^k) \right\|,$$

where we have denoted the induced operator norm of $||\cdot||$ by $||\cdot||_{\text{op}}$. By Lemma 3.6 and the coerciveness of E, we know that $(z^k)_{k=1}^\infty$ is a compact subset of $\operatorname{dom} h$. Assumption B then guarantees that the sequence is further contained in $\operatorname{int} \operatorname{dom} h$. This allows us to extend the local Lipschitz continuity of $\frac{1}{\tau} h - G$, also from assumption B, to Lipschitz continuity on this compact set. Furthermore, we assumed $\rho \in C^1(\mathbb{R}^n, \mathbb{R}^m)$ which implies that its Jacobian $||J_\rho(u)^*||_{\text{op}}$ is also Lipschitz on the compact set $\bar{U}$ for $U = (u^k)_{k=1}^\infty$. These properties allow us to find a fixed constant c so that the inequality (3.23) holds for all $k \in \mathbb{N}$. ■

Before we now come to the main theorem, we first collect a few properties of the set accum(u^0), that will allow the application of Lemma 3.11.

Lemma 3.13. *E is constant and finite on accum(u^0), i.e. $E(u) = E(v) < \infty \, \forall u, v \in$ accum(u^0) and we have*

$$\lim_{k \to \infty} \text{dist}(u^k, \text{accum}(u^0)) = 0. \tag{3.24}$$

Proof. $(u_k)_{k=1}^\infty$ is bounded due to Corollary 3.7. We choose a subsequence (u^{k_l}) with $\lim_{l \to \infty} u^{k_l} = u^*$. From the continuity of E on (u^{k_l}) we infer $\lim_{l \to \infty} E(u^{k_l}) = E(u^*)$. We further know from Corollary 3.7 that the sequence of function values itself is convergent to some value E^*, so that E is finite and constant on all these limit points. Equation (3.24) is true for all bounded sequences. ■

The following proof is now a slight adaptation of usual strategies for convergence under the KŁ-property [11] or [37, 210], with the difference that we apply the KŁ-property to the set accum(u^0) instead of the set of critical points, which nevertheless fulfills $E(u^*) < E(u^k) < E(u^*) + \eta$ for any $u^* \in$ accum(u^0) as required in Lemma 3.11, due to the monotone descent of the algorithm. We then apply our previous results and find a global convergence in $z^k = \rho(u^k)$.

Theorem 3.14 (Global Convergence). *Under the assumptions A and B, the sequence z^k either has finite length, $\sum_{k=1}^\infty ||z^{k+1} - z^k|| < \infty$, and converges to a limit $z^* \in \operatorname{int} \operatorname{dom} h$, or can be terminated after a finite number of steps.*

Proof. To apply the KŁ-property, we need to verify that $E(u^*) < E(u^k)$ for all indices k that we consider and accumulation points $u^* \in \operatorname{accum}(u^0)$. Lemma 3.6 shows that $E(u^{k+1}) \leq E(u^k)$, due to the convexity of h. Now if for some index l we have $E(u^k) = E(u^*)$, then the monotonicity of the sequence of objective values implies $E(u^{k+1}) = E(u^k)$ for the next index $k+1$. Together with Lemma 3.6 this implies $D_h(z^{k+1}, z^k) = 0$ and by the strong convexity of the assumptions B, $z^{k+1} = z^k$. Furthermore it is possible that iterates fulfill $z^{k+1} = z^k$ without fulfilling $E(u^{k+1}) = E(u^k)$, as ρ is not bijective. In both cases, the algorithm can be terminated, as Lemma 3.12 implies that $0 \in \partial E(u^{k+1})$.

Now, conversely, assume that the algorithm does not terminate after a finite number steps. We may then choose $l \in \mathbb{N}$ large enough so that both $E^* < E(u^l) < E^* + \eta$ and $\operatorname{dist}(u^l, \operatorname{accum}(u^0)) < \varepsilon$ are fulfilled. The positive constants ε and η are the ones required by the KŁ-property of E w.r.t to the set $\operatorname{accum}(u^0)$. On this set, E is constant and finite, as discussed in Lemma 3.13. From Lemma 3.11, we then find that

$$\varphi'\left(E(u^k) - E^*\right) \operatorname{dist}\left(0, \partial E(u^k)\right) \geq 1$$

holds for any accumulation point $u^* \in \operatorname{accum}(u^0)$, as $E(u^*) = E^*$ and for all u^k with $k > l$. Now we can apply Lemma 3.12 to find that

$$\varphi'\left(E(u^k) - E^*\right) \geq \frac{1}{c||z^k - z^{k-1}||}. \tag{3.25}$$

Further, we can consider the descent from Lemma 3.6 and apply that h is m-strongly convex to obtain

$$E(u^k) - E(u^{k+1}) \geq \frac{1-\tau L}{\tau} D_h(z^{k+1}, z^k) \geq \frac{m(1-\tau L)}{2\tau} ||z^{k+1} - z^k||^2. \tag{3.26}$$

Analogously to [11, 37], we use the concavity of φ to analyze the difference of function values in φ:

$$\begin{aligned} \Delta_{k,k+1} &=: \varphi\left(E(u^k) - E^*\right) - \varphi\left(E(u^{k+1}) - E^*\right) \\ &\geq \varphi'\left(E(u^k) - E^*\right)\left(E(u^k) - E(u^{k+1})\right). \end{aligned}$$

Inserting (3.25) and (3.26) and denoting constant terms by c' we gain

$$\Delta_{k,k+1} \geq \frac{||z^{k+1} - z^k||^2}{c'||z^k - z^{k-1}||}.$$

Now we are entirely in the setting of [37, Theorem 6.2] and likewise reformulate to

$$2\sqrt{||z^k - z^{k-1}||c'\Delta_{k,k+1}} \geq 2||z^{k+1} - z^k||$$

and use the inequality $2\sqrt{ab} \leq a + b$ to gain

$$2||z^{k+1} - z^k|| \leq ||z^k - z^{k-1}|| + c'\Delta_{k,k+1}.$$

Summing these inequalities for $k = l+1, \dots, n$, then yields

$$\begin{aligned} 2\sum_{k=l+1}^{n} ||z^{k+1} - z^k|| &\leq \sum_{k=l+1}^{n} ||z^k - z^{k-1}|| + c' \sum_{k=l+1}^{n} \Delta_{k,k+1} \\ &= \sum_{k=l+1}^{n} ||z^{k+1} - z^k|| - ||z^{n+1} - z^n|| + ||z^{l+1} - z^l|| + c' \sum_{k=l+1}^{n} \Delta_{k,k+1} \\ &\leq \sum_{k=l+1}^{n} ||z^{k+1} - z^k|| + ||z^{l+1} - z^l|| + c'\Delta_{l+1,n+1}, \end{aligned}$$

where the last inequality is gained by telescoping all Δ. Reinserting the definition of $\Delta_{l+1,n+1}$ and simplifying then results in

$$\sum_{k=l+1}^{n} ||z^{k+1} - z^k|| \leq ||z^{l+1} - z^l|| + c'\varphi(z^l - z^{l+1}) - c'\varphi(z^n - z^{n+1}) < \infty$$

As φ is positive this implies that the whole sequence z^k is a Cauchy sequence and converges due to metric completeness. ■

Remark 3.15 (Strong convexity of h). The m-strong convexity might seem like a limiting assumption, yet it is always possible to construct a function $\tilde{h}$ that fulfills this property, if G is L-smooth adaptable for some h (see also the related discussion in [37]). First if $Lh - G$ is convex, then $L(h+w) - (G+Lw)$ is also convex for any function w, especially $w = \frac{m}{2}||\cdot||^2$. Define $\tilde{h} = h + w$, $\tilde{G} = G + Lw$, and $\tilde{R} = R - L(w \circ \rho)$. Now the new energy $\tilde{G} \circ \rho + \tilde{R}$ is equal to the old formulation, but the pair $(\tilde{G}, \tilde{h})$ is convex with $\tilde{h}$ being m-strongly convex. However we remark that the new function $\tilde{h}$ might make it more difficult to solve the resulting subproblem.

From the convergence result of Theorem 3.14 on the sequence z^k, we can return to u^k:

Corollary 3.16 (Convergence to critical points). *All accumulation points $u^* \in$ accum$E$ of $(u^k)_{k=1}^{\infty}$ are stationary points of E, i.e. $u^* \in$ crit$E$ and belong to the same outer sequence z^k so that $z^* = \rho(u^*)$.*

Proof. Combining the bound on the slope in Lemma 3.12 and global convergence of z^k's from Theorem 3.14 we immediately see that dist$(0, \partial E(u^{k+1})) \to 0$ as $k \to \infty$. Furthermore, we know that $E(u^k) \to E(u^*) = E^*$ so that all subsequences of u^k fulfill the definition of the limiting subdifferential and $0 \in \partial E(u^*)$. We know also that $\lim_{k\to\infty} z^k = \lim_{k\to\infty} \rho(u^k) = z^*$. Let $u^* \in$ accum(u^0) be arbitrary with the sequence by $u^{k_l} \to u^*$. Due to continuity of ρ we have $\rho(u^*) = \lim_{l\to\infty} \rho(u^{k_l}) = z^*$. ■

This result shows the connection between the 'auxiliary' outer sequence of gradient steps z^k, which converges globally, due to the KŁ-property and the sequence of actual update steps u^k. The algorithm converges to a stationary point of E and all accumulation points not only have the same value in E, but also in $G \circ \rho$, as $G(\rho(u^*)) = G(z^*) \; \forall u^* \in$ accum(u^0).

3.3.4 Global Convergence of the inner sequence

A necessary consequence of the previous subsection is that all accumulation points have an equal value R^* in R, hence are elements of the set $S = \{u \in \mathbb{R}^n \mid R(u) = R^*, \; \rho(u) = z^*\}$. Naturally, if this set is a singleton, then the subsequence convergence of the sequence $(u^k)_{k=1}^{\infty}$ extends to global convergence. The cardinality of the set C is however difficult to check a-priori. Nevertheless it turns out that if we finally also assume that the nonconvex subproblems are solved globally, then the convergence result follows from the familiar notion of uniqueness of R-minimizing solutions. We further remark that the assumption of global solutions to subproblems also allows us to weaken the continuity assumption made in Assumption B to lower semi-continuity of E.

Let us define R-minimizing in the following way, as given for example in [254, Def 3.24]:

Definition 3.17. *A vector $u^* \in \mathbb{R}^n$ is called R-minimizing with respect to a solution set $\{u \in \mathbb{R}^n \mid F(u) = v\}$ of an operator $F : \mathbb{R}^n \to \mathbb{R}^m$ and a vector $v \in \text{Im}(F)$, if $F(u^*) = v$ and*

$$R(u^*) \in \min \{R(u) \mid u \in \mathbb{R}^n, F(u) = v\}.$$

Now the uniqueness of such a vector directly corresponds to global convergence if the subproblems are solved globally:

Theorem 3.18 (Global Convergence). *If the subproblems are solved to global optimality, i.e. u^{k+1} fulfills $E_{u^k}(u^{k+1}) \leq E_{u^k}(u) \; \forall u \in \mathbb{R}^n$ and all assumptions hold, then*

$$u^* \in \text{accum}(u^0) \;\Rightarrow\; u^* \text{ is } R\text{-minimizing w.r.t } \rho(u) = z^*. \tag{3.27}$$

In particular, if the R-minimizing element w.r.t. $\rho(u) = z^*$ is unique, then the sequence of iterates u^k converges globally.

Proof. Let $u \in \{u \in \mathbb{R}^n \mid \rho(u) = z^*\}$ and $u^* \in \text{accum}(u^0)$ be arbitrary. As $z^* \in \operatorname{int}\operatorname{dom} h$, this implies to $u \in \rho^{-1}(\operatorname{int}\operatorname{dom} h)$. Rewriting the optimality assumption $E_{u^k}(u^{k+1}) \leq E_{u^k}(u) \;\; \forall u \in \rho^{-1}(\operatorname{dom} h)$ results in the inequality

$$R(u^{k+1}) - R(u) + \frac{1}{\tau} D_h(\rho(u^{k+1}), \rho(u^k)) - \frac{1}{\tau} D_h(\rho(u), \rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u^{k+1}) - \rho(u) \rangle \leq 0.$$

Taking the limit of $l \to \infty$ for a subsequence $u^{k_l} \to u^*$ with the knowledge that $\lim_{l\to\infty} \rho(u^{k_l}) = z^*$ and $\rho(u) = z^*$ by assumption, then reveals that $R(u^*) \leq R(u)$, showing that $u^* \in \text{accum}(u^0)$ is an R-minimizing solution to $\rho(u) = z^*$. If in particular, the set of R-minimizing solutions is already a singleton, then the result follows, as the set $\text{accum}(u^0)$ nonempty due to Corollary 3.7. ■

Example 3.19. *As an example, consider a simple periodic function $\rho : \mathbb{R}^n \to \mathbb{R}^n$, $\rho(u) = (\sin(u_1), \dots, \sin(u_n))$ and $R = ||\cdot||_p$ for $p > 0$. The R-minimizing solution to $\rho(u) = z$ is then unique for any $z \in$ Im(ρ). To see this consider that $\sin(x)$ is bijective on $[-\frac{\pi}{2}, \frac{\pi}{2}]$. For any level set $z_i \in [-1, 1]$ we can find a unique element u_i in this interval $[-\frac{\pi}{2}, \frac{\pi}{2}]$, so that $\sin(u_i) = z_i$. Due to the strict monotonicity of $||\cdot||_p$ on either $\mathbb{R}^+$or $\mathbb{R}^-$, any other element u_i that fulfills $\rho(u_i) = z_i$ must have a greater function value in R.*

3.4 Implementation Details

This section will focus on several interesting special cases of our general composite model (3.2) and also discuss possible pairs G, h.

3.4.1 Modeling

For several implementation examples it will be convenient to be a bit more specific with our choices of G, ρ and R. One example is the natural extension to several additive terms,

$$E(u) = \sum_{i=1}^m G_i \left(\sum_{j=1}^n \rho_{ij}(u_j) \right) + \sum_{j=1}^n r_j(u_j), \tag{3.28}$$

which was already mentioned briefly in (3.17). This formulation is interesting due to its straightforward interpretation as a way to optimize a function with n measurements of linear combinations of our m variables. Hence the task relates to nonlinear regression models and imaging with nonlinear measurements. It is also a natural discretization of a general nonlinear integral operator as defined for example in [231, 16].

However, defining $G : \mathbb{R}^{m\times n} \to \mathbb{R}$, $G(v) = \sum_{i=1}^m G_i(\sum_{j=1}^n v_{ij})$ and $\rho : \mathbb{R}^n \to \mathbb{R}^{m\times n}$, defined componentwise by $\rho_{ij}(u_j)$, for univariate functions G_i, ρ_{ij}, r_j, we see that this is just an instance of the general composite model . The maximal generalization would be achieved by taking $\rho_{ij} : \mathbb{R}^q \to \mathbb{R}^p$, $G_i : \mathbb{R}^p \to \mathbb{R}$, $r_j : \mathbb{R}^q \to \mathbb{R}$, although in practice q would have to be quite small if we wanted to solve the subproblems by an exhaustive search.

Writing out the majorizer to (3.28) with univariate functions under the assumption that $\sum_{i=1}^m L_i h_i - G_i$ is a convex function (as required for L-smooth adaptability, Definition 3.2) gives

$$E_{u^k}(u) = \sum_{i=1}^n D_{h_i}(\rho_{ij}(u_j), \rho_{ij}(u^k)) + \sum_{i=1}^n G_i' \left(\sum_{j=1}^m \rho_{ij}(u_j^k) \right) \sum_{j=1}^m \rho_{ij}(u_j) + \sum_{j=1}^n r_j(u_j), \tag{3.29}$$

up to constant terms. This reveals that the majorization function is separable if each h_i is chosen separable so that $h_i(u) = \sum_{j=1}^n h_{ij}(u_j)$, as the Bregman distance to these h_i is then also separable and the summation over all m parameters can be exchanged with the summation over all n 'measurements'. The resulting m independent 1D dimensional subproblems can then be solved efficiently.

Remark 3.20. This generalization is not only interesting for regression-type problems, where the outer sum naturally sums over all samples and the inner sum over a superposition of parametrized functions, but also for any sort of problem where it would make sense to split a function into the composition of a function and a super-position of simpler functions. As an example consider the 1-dimensional polynomial problem

$$P(x) = \left(\sum_{i=0}^{p} a_i x^i - f \right)^2 + \sum_{i=0}^{q} b_i x^i. \tag{3.30}$$

While it would be natural to choose $\rho : \mathbb{R} \to \mathbb{R}, \rho(x) = \sum_{i=0}^{p} a_i x^i - f$, i.e the inner polynomial, another possibility would be to choose $\rho : \mathbb{R} \to \mathbb{R}^p, \rho(u) = (a_0, \dots, a_n x^n)$ and likewise to set $G : \mathbb{R}^p \to \mathbb{R}, G(v) = (\sum_{j=1}^{p} v_j - f)^2$. A separable majorizer for this G using (3.29) would lead to different subproblems than before.

An interesting fact about the general composite model (3.2) is that we are actually allowed a great deal of freedom, as both G and ρ can be nonconvex. It is possible to insert any invertible function f and its inverse f^{-1} on dom ρ and solve the equivalent problem with $\tilde{G} = G \circ f$ and $\tilde{\rho} = f^{-1} \circ \rho$. As an example, consider the following model

$$E(u) = F\left(\prod_{j=1}^{n} g_j(u_j) \right),$$

where we have a product of parametrized functions $g_j : \mathbb{R} \to \mathbb{R}^+$. We can set $G : \mathbb{R}^n \to \mathbb{R}, G(v) = F\left(\exp(\sum_{j=1}^{n} v_j) \right)$ and $\rho_j : \mathbb{R} \to \mathbb{R}, \rho_j(u_j) = \log(g_j(u_j))$, and recover the additive superposition of parameters in (3.28).

We may freely use these possibilities due to Corollary 3.16. The proposed algorithm converges to the set of stationary solutions of E. This result is independent of the actual decomposition of E into G, ρ, R as long as the chosen triple G, ρ, R still fulfills all required conditions.

3.4.2 Choices for the Bregman Distance

Up to now, we always considered an abstract pair (G, h) fulfilling the conditions that both functions are smooth and $Lh - G$ and h are convex. Now we will detail several tangible instances of these functions.

The trivial case is present when G is a concave function. We are then allowed to choose an arbitrary convex function h, as $-G$ is itself convex. A natural choice is then to choose h as a linear function, as its Bregman distance then vanishes,

$$D_h(u, v) = \langle h, u \rangle - \langle h, v \rangle - \langle h, u - v \rangle = 0.$$

The resulting majorizer,

$$E_{u^k}(u) = \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k) \rangle + G(\rho(u^k)) + R(u), \tag{3.31}$$

is an instance of iterative reweighting related to variants discussed in [213, 212]. If R is convex and ρ is coordinate-wise convex, then the majorizer is even convex. When R is a convex function and ρ is an affine function, then we recover an instance of the difference of convex functions (DC) algorithm [278]. Note that for the second part of our analysis in section 3.3 to hold, we need to choose h strongly convex. We mention in passing that the results of Corollary 3.7 also hold relative to the Bregman distance $D_{-G}(\cdot, \cdot)$.

The standard case is present when G is L-smooth. We then choose $h = \frac{1}{2}|| \cdot ||^2$ and recover the usual Euclidean distance measure via $D_h(u, v) = \frac{1}{2}||u - v||^2$. Note that G can be L-smooth without being convex, for example when considering a smooth truncated quadratic function [9].

However, even when G is L-smooth, more advantageous functions h might exist. Consider the function $G : \mathbb{R}^m \to \mathbb{R}$,

$$G(v) = \frac{1}{2}||Av - f||^2, \tag{3.32}$$

for a matrix $A \in \mathbb{R}^{p \times m}$. The function is L-smooth with $L = ||A^T A||_{\mathrm{op}}$. However we can also inspect $Lh - G$ via its second derivative condition[2],

$$L\nabla^2 h(v) - A^T A \succeq 0 \quad \forall v \in \mathbb{R}^m. \tag{3.33}$$

We could of course choose $h = G$, as (3.32) is convex, thereby solving the original problem in each subproblem, but we are looking for functions h so that the subproblems are easy to solve. Such a choice is presented by $h = \frac{1}{2}|| \cdot ||_D^2$ with a diagonal matrix D. Choosing D so that $D - A^T A \succeq 0$ yields a diagonal preconditioning - we intrinsically find vector-valued step sizes by an appropriate choice of h.

An important and motivating property of the L-smooth adaptable property is however the inclusion of logarithmic functions, most prominently the Kullback-Leibler divergence as possible terms for G, even though this function is not globally L-smooth [18]. Consider the function $G : \mathbb{R}^m \to \overline{\mathbb{R}}$:

$$G(v) = \sum_{i=1}^{p} (Av)_i - f_i + f_i \log\left(\frac{f}{(Av)_i}\right), \tag{3.34}$$

for $A \in \mathbb{R}_+^{p \times m}$ and $f \in \mathbb{R}_{++}^p$ and the set $C = [0, \infty)^m$. An appropriate function h is given by $h(v) = -\sum_{i=1}^m \log(v_i)$. [18, Lemma 7] reveals that the appropriate constant is $L = ||f||_1$ so that $Lh - G$ is convex on $\operatorname{int} \operatorname{dom} h = (0, \infty)^m$. This function and related 'entropy' functions are possible choices for h, yet, as now the domain of h is strictly smaller than $\mathbb{R}^n$, one has to check, whether the energy fulfills $z^{k+1} \in \operatorname{int} \operatorname{dom} h$ and $z^* \in \operatorname{int} \operatorname{dom} h$ to guarantee well-posedness of the iterations and global convergence, respectively.

A general observation, see [71] or [214, Example 33], is that once we have gained a Bregman distance D_h from h, we may actually use a whole family of functions h_k as long as they majorize h while being convex,

$$h^k \in \{h^k \text{ essentially smooth}, \overline{\operatorname{dom} h^k} = C \mid h^k - h \text{ convex } \}.$$

The induced Bregman distance then fulfills

$$D_{h^k}(u, v) \geq D_h(u, v) \quad \forall u, v \in \mathbb{R}^n.$$

A specific instance of this observation is choosing h first and then implementing adaptive step-sizes in this fashion by varying h^k or the approach of [71] where a sequence $h^k = \frac{1}{2}|| \cdot ||_{A^k}^2$ is constructed with symmetric positive definite matrices A^k.

This is of course only a short overview of possible pairs (G, h), further examples can be found in [19, 18, 37, 214, 71].

3.4.3 An example of a non-separable, solvable subproblem

This section will continue the discussion started in the introductory section about specific subproblems. We have noted that the presented approach is especially interesting, if the considered subproblems (3.15) can still be solved to global optimality. An interesting for this are cases where the subproblems can be solved to global optimality by lifting [4, 228] or other relaxation strategies [152, 41, 133]

Models that include the total variation norm in place of the regularization term R are ubiquitous in imaging tasks [45, 245] and have been a major motivation in applications of our work. These models will appear as special instances of the discussed 'liftable' subproblems.

[2]We follow the notation that A is positive semi-definite if $A \succeq 0$.

We will start with a basic representation of functions that are amenable to lifting,

$$E(u) = \sum_{i=1}^{n} \nu_i(u_i, (Du)_i), \tag{3.35}$$

with continuous functions $\nu : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ that are convex in their second argument and where D denotes a finite-difference gradient. This is a natural discrete representation of the continuous model (3.10), which can be efficiently solved by functional lifting [228]. The choice of $\nu_i(x, y) = g_i(x) + ||y||_2$ then recovers a composite model with some term $G = \sum_{i=1}^{n} g_i$ and a regularizer R which is total variation [245]:

$$E(u) = \sum_{i=1}^{n} g_i(u_i) + ||Du||_1, \tag{3.36}$$

This is a successful strategy, but its application is limited by the fact that separability is needed. A much more general model would be

$$E(u) = G(\rho(u)) + \sum_{i=1}^{n} \gamma_i((Du)_i), \tag{3.37}$$

with ρ separable as before, G L-smooth adaptable and γ_i convex. Yet while the lifting scheme of [228] is not applicable due to the non-separability of G, this is nevertheless a special instance of our general problem (3.2). Indeed we can write down the majorizer, assuming a separable $h(u) = \sum_{i=1}^{n} h_i(u_i)$, as

$$E_{u^k}(u) = D_h(\rho(u), \rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u^k) \rangle + \sum_{i=1}^{n} \gamma_i((Du)_i), \tag{3.38}$$

up to constants. Now this majorizer is in turn a particular instance of (3.35), as the first two terms are separable, and we can now solve (3.37) by iteratively solving the lifting subproblems.

Furthermore, we can even exchange convexity of γ_i for differentiability. For arbitrary γ_i that are L-smooth adaptable, we can linearize the second term as well, in full analogy to (3.29), giving the majorizer

$$\begin{aligned} E_{u^k}(u) =& \sum_{i=1}^{n} d_{h_G}(\rho(u), \rho(u^k)) + \langle \nabla G(\rho(u^k)), \rho(u^k) \rangle \\ &+ \sum_{i=1}^{n} d_{h_{\gamma_i}}((Du)_i, (Du^k)_i) + \gamma_i'((Du^k)_i)(Du)_i. \end{aligned}$$

which is again an instance of (3.35). For concave γ_i this is particularly attractive as we can choose h_{γ_i} a as linear functions and just keep the linearization in full analogy to iterative reweighting as discussed in the previous subsection in (3.31).

While this approach greatly increases the applicability of lifting schemes, it is important to keep in mind that previous global optimality considerations for lifting schemes [228] do not translate to these generalized problems. From section 3.3 we only gain convergence to stationary points. We will use the next section to analyze the quality of solutions that we receive numerically.

3.4.4 Inertia

A small side note to the previous investigations that is nevertheless quite interesting in the context of nonconvex optimization is inertia. Once we have (3.15), we can just as well consider

$$\begin{aligned} E_{u^k, u^{k-1}}(u) =& \frac{1}{\tau} D_h(u, u^k) + \langle \nabla G(\rho(u^k)), \rho(u) - \rho(u^k) \rangle + G(\rho(u^k)) \\ &+ R(u) + \frac{\beta}{\tau} \left(D_h(\rho(u), \rho(u^k)) - D_h(\rho(u), \rho(u^{k-1})) \right), \end{aligned} \tag{3.39}$$

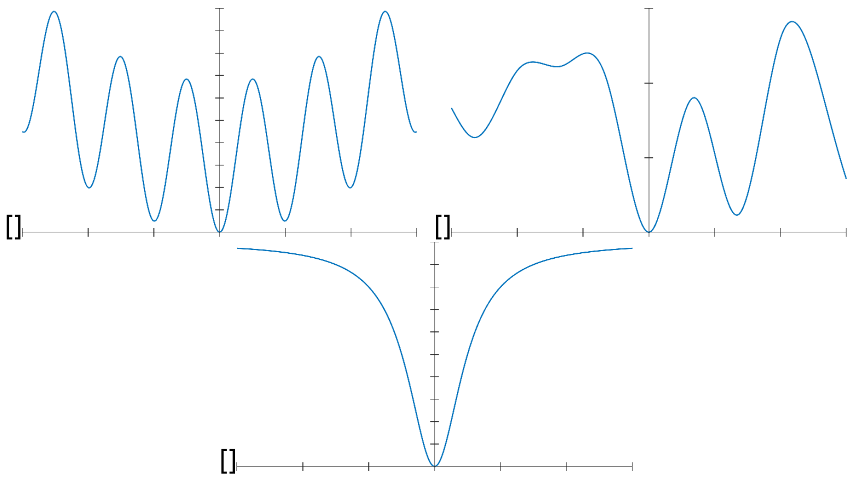


Figure 3.5: Example of nonlinearities used in synthetic experiment, (a) $x^2 - 10\cos(2\pi x)$ (Rastrigin's function [201]), (b) random spline function with 12 queries and (c) $\frac{x^2}{1+x^2}$.

inserting an inertial term into the generalized forward-backward equation analogous to [211].

Inertia can be quite valuable for first-order optimization, especially as we allow G to be nonconvex, but only utilize its gradient, i.e local information in each step. In practice we observe that inertia can sometimes help the algorithm to reach better minima or speed up the initial convergence for badly conditioned G. Also spurious stationary points can often be overcome. Furthermore, the additional cost of solving (3.39) is minuscule compared to the non-inertial variant.

However, due to the non-existence of the triangular inequality for Bregman distances, we cannot bound the iterations by a Lyapunov function in general as in previous work [211] and continue the proof of convergence with this Lyapunov function as a majorizer analogous to section 3.3. A related discussion and solution in the convex setting can be found in [116]. For the special cases of induced squared norms, i.e. $h = ||u||_A^2$, convergence still follows by adapting subsection 3.3.3 to the results of the recent work [210], but we omit a further discussion.

In practice this modification still works well in many cases, setting $\beta < 0.5$. It is also possible to backtrack in case of violations of Lyapunov function bounds.

3.5 Experimental results

In this section we analyze the proposed algorithm numerically. We will first consider a synthetic example, where we will be able to compare the algorithm to other methods easily. We will then move to an imaging application, the depth super-resolution from raw time-of-flight data.

3.5.1 Synthetic experiments

We analyze the following energy

$$\min_{u \in \mathbb{R}^n, u_i \in [a,b]} F_f(A\rho(u)) + R(u), \tag{3.40}$$

where we have a bounded interval $[a, b]$, an L-smooth adaptable function $F_f \circ A : \mathbb{R}^n \to \mathbb{R}$ and regularizer $R(u) = \sum_{i=1}^n r(u_i - u_i^*)$ with $r : \mathbb{R} \to \mathbb{R}$. ρ is chosen separable so that $\rho(u) = (\rho_1(u_1), \dots, \rho_n(u_n))$ with $\rho_i : \mathbb{R} \to \mathbb{R}$, whom we will in general choose equal, and omit the subscript. The nonlinearity r is aligned so that $\arg\min_x r(x) = 0$. We first draw $u^* \in [a, b]^n$ at random and then set $f = A\rho(u^*)$. We choose F_f as a measure of distance between f and $A\rho(u)$ that fulfills $u^* \in \arg\min_u F_f(A\rho(u))$ and $F_f(A\rho(u^*)) = 0$. Through this construction, we can guarantee that the drawn u^* will be a global minimizer of (3.40).

Now we vary the difficulty of this possibly nonconvex optimization problem in two ways. First we choose nonlinearities ρ, r as either

(1) *Simple* $\rho(x) = \exp(x)$, $r(x) = x^2$

(2) *Doable* $\rho(x) = x^2 - 10\cos(2\pi x)$ [201], cf. subsection 3.5.1, $r(x) = \frac{x^2}{1+x^2}$, cf. subsection 3.5.1

(3) *Difficult* ρ is a (coercive) piecewise cubic polynomial drawn by interpolating 12 values in $[a, b]$, $r(x) = -\text{sinc}(x)$

(4) *Very Difficult*ρ is a (coercive) piecewise cubic polynomial drawn by interpolating 12 values in $[a, b]$, $r(x) = x^2 - 10\cos(2\pi x)$.

This allows us to move from a nicely behaved, almost convex test case (1) to a nonconvex problem with a well-behaved minimizer (2), adding further oscillations in (3) and finally arriving at two "very nonconvex" functions in (4).

Then we vary the function $G = F_f \circ A$. Note that this function critically determines the interconnection of variables. If A is a diagonal matrix, then the problem is fully separable and can by solved by n separate 1D optimizations with a single step of the algorithm, but if A is a full matrix, then all variables are interdependent. Further, when A is a rectangular matrix, then the system of nonlinear equations is under-determined and the function landscape is (intuitively) not as well-behaved. Also, we are allowed to choose nonconvex functions F as long as G is still L-smooth adaptable.

(a) *Convex, local:* $F_f(v) = \frac{1}{2}||v - f||^2$,$A \in \mathbb{R}^{n\times n}$ is a random matrix whose entries are normally distributed relative to its diagonal. An appropriate essentially smooth function is $h(v) = \frac{1}{2}||v||_D^2$, where D is a diagonal matrix with entries $d_i = \sum_{j=1}^n |A^T A|_{ij}$

(b) *Convex, non L-smooth, local:* F_f is the KL-divergence $F_f(v) = \sum_{i=1}^n v_i - f_i \log(v_i)$, $A \in \mathbb{R}^{n\times n}$, is chosen as in (a). Here h is given by Burg's entropy $h(v) = \sum_{j=1}^n -\log(v_j)$ [18].

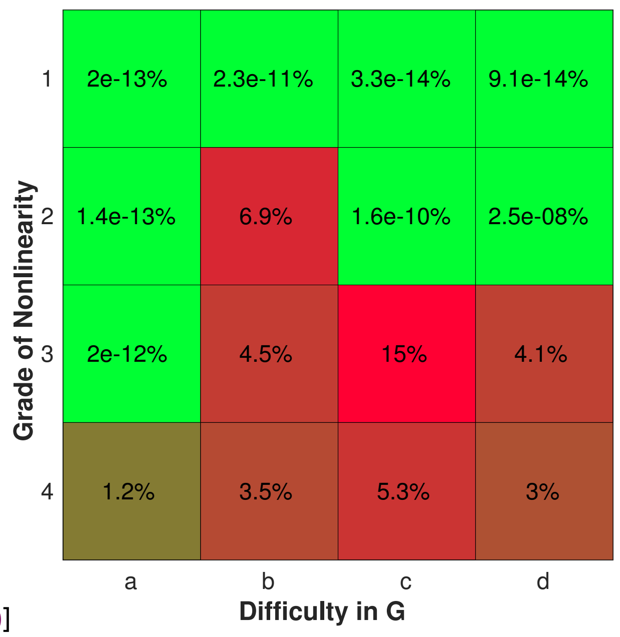
(c) *Convex, full:* $F_f(v) = \frac{1}{2}||v - f||^2$,$A \in \mathbb{R}^{m\times n}$ is a full random matrix with singular values in $[\frac{1}{\log(n)}, 1]$. $m = \frac{n}{3}$. Choose h as in (a).

(d) *Nonconvex, full:* F_f is a smooth-truncated quadratic [9], i.e. a smoothed version of $F_f(v) = \sum_{i=1}^m \min((v_i - f_i)^2, \lambda)$, $A \in \mathbb{R}^{m\times n}$ is a full random matrix with singular values in $[\frac{1}{\log(n)}, 1]$, $m = \frac{n}{3}$. Choose h as in (a).

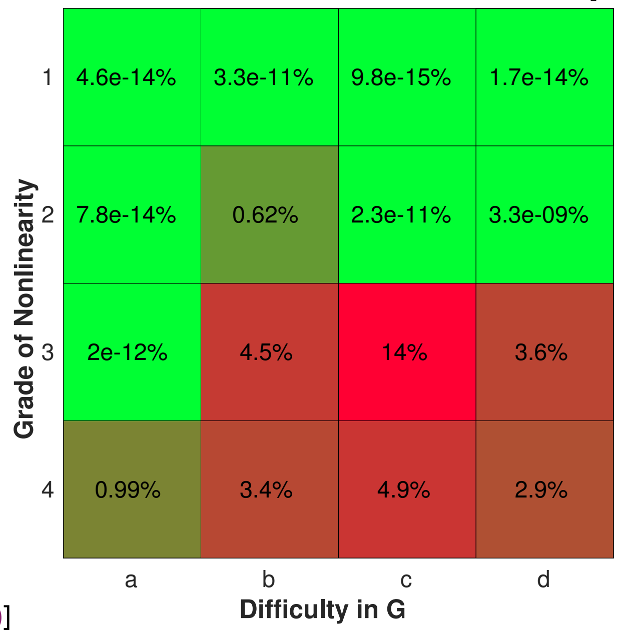
We run our method (3.15) without and with inertia, $\beta = 0.4$ (3.39). The subproblems in each iterations are fully separable, so we solve the 1D problems in parallel by exhaustive search with a sufficient amount of trial points and a parabolic refinement around the approximate minimizer to desired precision. The refinement is a standard technique, e.g. [128] for 1D local optimization and further references can be found, for example in book of Luenberger [181, pp. 217, 224]. This technique has also been used previously in imaging, for example, to refine exhaustive search procedures in the context of quadratic decoupling for stereo in [158].

To mitigate the risk of lucky initializations, we run the algorithm 25 times with random starting vectors for each test case and show the result which reached a median energy value. We set $n = 150$ and $[a, b]$ = $[-3, 3]$, respectively $[a, b] = [\epsilon, 3]$ for the Poisson case and implement the proposed method in MATLAB.

The results for all test cases can be found in Figure 3.6. We see that either increasing the difficulty in G or the difficulty of the nonlinearity makes the overarching optimization problem more difficult. Remarkably, our algorithm was able to find near-global optima for many test cases, especially the performance in row (2) is very good. However we see that the increased oscillations in (3) eventually degrade the quality of solutions. We also notice that differences can appear even for convex functions G in (a) and (b).

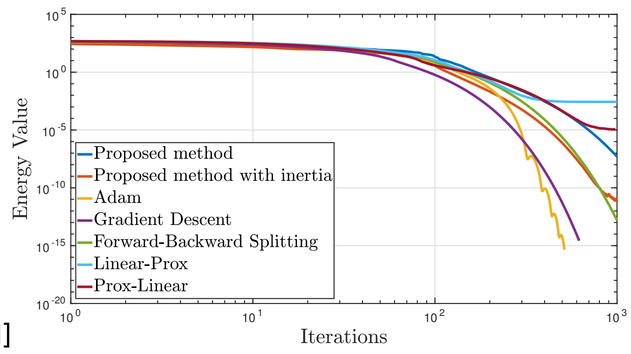


[Proposed method(3.15)]



[Proposed (with inertia) (3.39)]

Figure 3.6: Examination of different synthetic experiments. We increase the difficulty in G from left to right and in ρ, r from top to bottom, as detailed in section 3.5. In each cell we show the value $\frac{E(\bar{u})-E^*}{\tilde{E}}$, the value $E(\bar{u})$ reached by the algorithm minus the global minimum E^*, normalized by $\tilde{E}$, a (sampled) median of the energy values of the function, indicating the quality of the solution relative to the overall energy landscape.



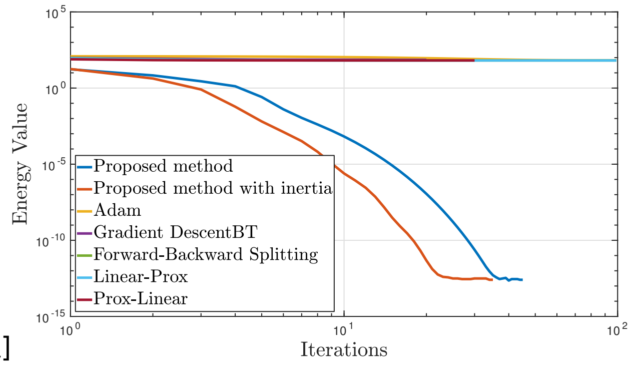


Figure 3.7: Convergence of various first-order methods for composite optimization to global optimality. This figure shows problem 1d, i.e. a problem where R, ρ are easy and G difficult, and 3a, the opposite case. Only our method provides favorable results for the difficult case 3a. Shown are the proposed method (3.15), proposed with inertia: (3.39), Adam: [145], gradient descent: (3.1), forward-backward splitting: (3.4), Linear-Prox: (3.5), Prox-Linear: (3.6)

The squared l^2 norm in (a) seems to be easier to optimize globally, although the disparity to (b) is also connected to the analytical step sizes, that we choose. Choosing larger stepsizes for (b), e.g. by backtracking, would recover a similar behavior to (a).

We now compare with other first-order nonlinear optimization methods, namely as mentioned in the related work section, gradient descent (3.1), forward-backward splitting (3.4), the inner linearization, 'prox-linear', (3.5) and the outer linearization (3.6).

To fairly evaluate all majorizers we generally solve the subproblems in forward-backward splitting (3.4), and outer linearization (3.6) to global optimality, again with exhaustive search and parabolic fitting. For prox-linear (3.5), the subproblems do not decouple and we apply a standard interior point solver in each iteration. We otherwise apply the same methodology as before. We compare the convergence of all algorithms to the global minimum for two characteristic cases, '1d' and '3a'. While the first case denotes simple ρ, R and difficult G, the second case denotes the opposite. We expect most methods to do well in the first case, but the second case is not as clear. Subsection 3.5.1 and subsection 3.5.1 show the results. It turns out that indeed all methods can reliably solve the first case, subsection 3.5.1, but only our method can find near-optimal solutions for the second test case, subsection 3.5.1. To compare our method to a modern 'aggressive' inertial variant that does not admit to a majorization-minimization framework, we also include the Adam optimizer [145], however while this optimizer can find better minima, it is still far off from the global solution in test case '3a'.

We can go a step further and compare the proposed algorithm to global optimization methods. See Figure 3.8 for a plot showing test case '2c' and the energetic difference to the global minimum versus the runtime of each the algorithms, with time in a log-scale. Previously mentioned algorithms are shown, as well as the MATLAB default implementations of a genetic algorithm [108], particle swarm [144], pattern search [13], simulated annealing [130] and multi-start methods [289]. All of these methods can reliably find global near-optimal solutions for low dimensions, however in our setting of $n = 150$ these methods fail to find a global minimizer within reasonable time constraints, as their efficiency decreases with the number of variables. In contrast, our method exploits the structure of the objective function, linearizing the convex outer function and solving the separable subproblems globally, and scales well into higher dimensions. We mention briefly that the apparent slow runtime of 'prox-linear' and 'linear-prox' for this test case is partly implementation related, as we solve 'prox-linear' with a generic interior point solver in each iteration, but also because both algorithms converge to very flat local minima.

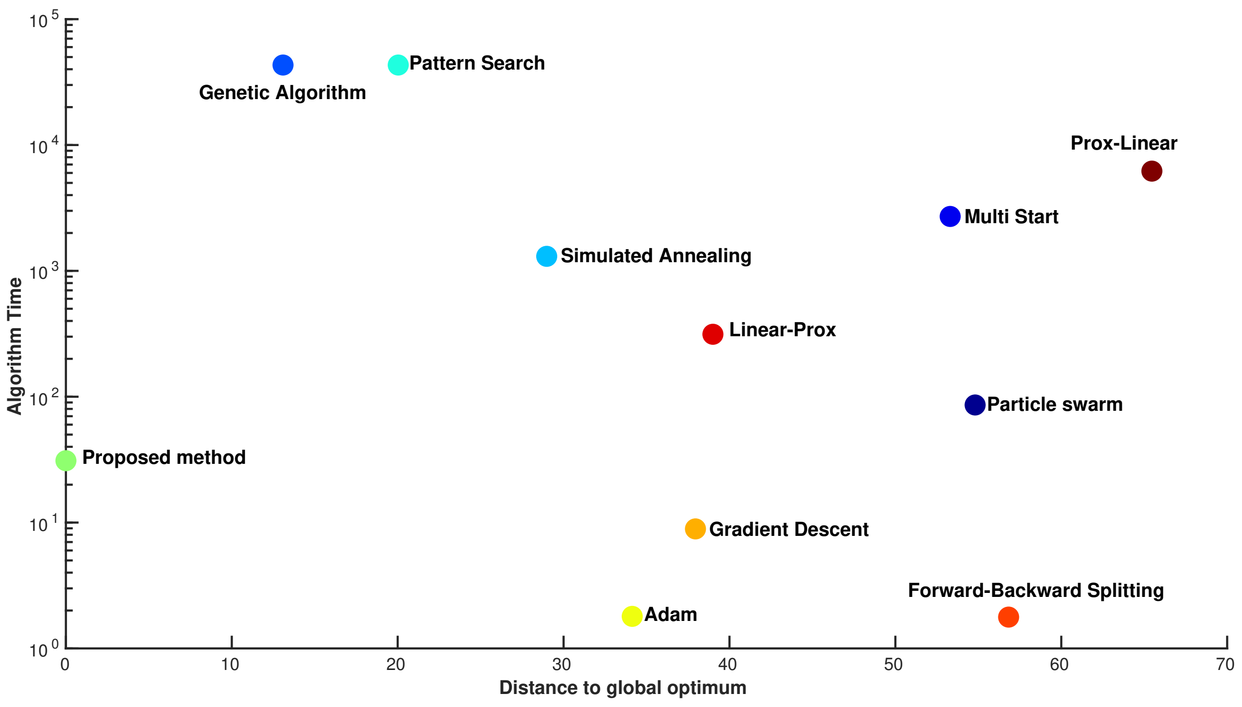
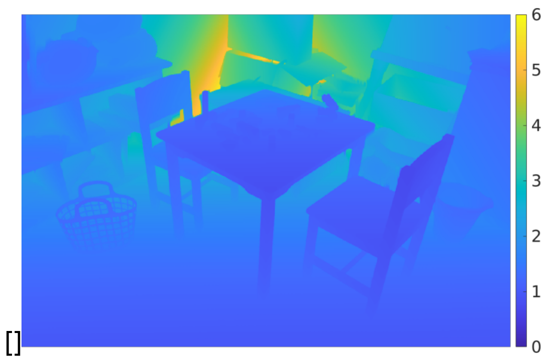


Figure 3.8: Evaluation of various optimization methods for test case ’2c’, in terms of energetic difference to global optimum vs. runtime in seconds.



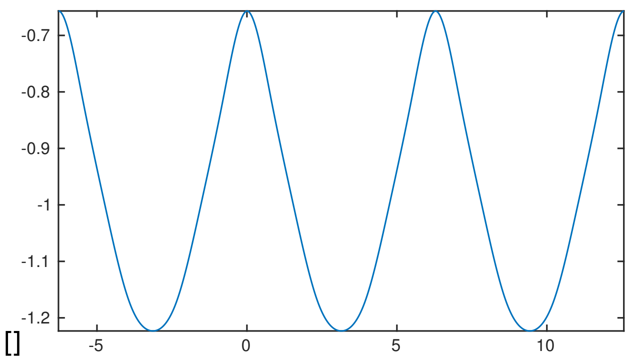


Figure 3.9: Ground Truth depth data [253] shown to the left and synthetic autocorrelation function generated from trapezoidal signal to the right.

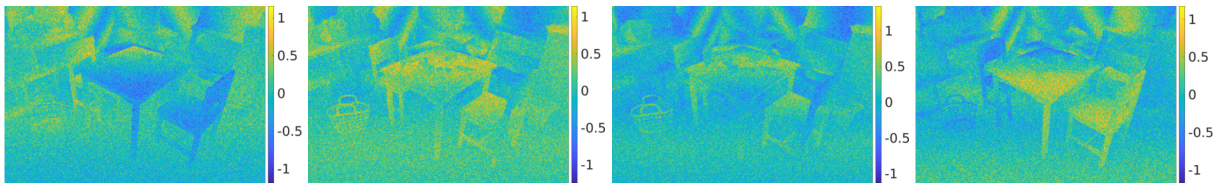


Figure 3.10: The four difference measurements y_{ij}, generated by equation (3.42) with subsampling and Gaussian noise.

3.5.2 Time-of-Flight Depth Reconstruction

Time-of-Flight cameras are used to recover depth images of a scene. They illuminate the scene with a continuous wave and measure the time of flight of reflecting waves. A modern hardware for this task are correlation photo-sensors, which directly measure the correlation of the incoming wave with a reference wave [162]. The inversion of this correlation computes the depth, however the process is highly non-linear and in practice often solved by assuming the incoming waves to be purely sinosoidal and computing the analytical inversion at each pixel separately. This introduces several systematic errors into the depth measurements, especially at lower frequencies [174]. Further, these sensors are have a relatively low resolution due to their complexity and measurements contain a significant amount of noise.

A recent work on time-of-flight super-resolution [300] shows a variational model which includes the precise reference wave, downsampling, blur and noise effects. They model the incoming wave as arbitrary periodic function and find it by thorough calibration. In [300], the resulting nonconvex energy model is solved by alternating local optimization in all variables. We will show that the problem can be solved with our approach and test on synthetic data.

In the following, we will assume the following imaging model of a time-of-flight system

$$\tilde{y}_{ij} = a_i g_i \left(\frac{4\pi f_i}{\lambda} u + \frac{2\pi j}{n} \right) + b_i =: k_{ij}(u) + b_i, \tag{3.41}$$

for measurement j in frequency f_i and g_i the 2π-periodic autocorrelation in frequency i that is either calibrated or otherwise known, e.g as a cosine. a_i is the amplitude in frequency f_i, n the number of measurements in each frequency f_i and λ the speed of light. To remove the background illumination b_i it is customary in Time-of-Flight literature to consider the difference measurements

$$y_{ij} = k_{ij}(u) - k_{i,j+\frac{n}{2}}(u) =: \rho_{ij}(u). \tag{3.42}$$

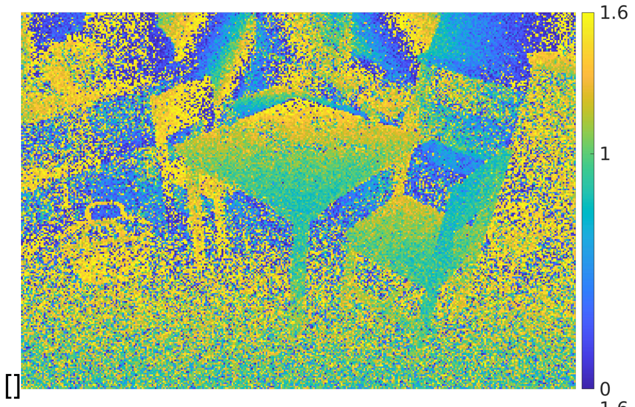
The problem of recovering a high-resolution depth image u from measurements y_{ij} can now be stated as the energy minimization of

$$E(u) = \sum_{i=0}^{m-1} \sum_{j=0}^{n/2-1} ||y_{ij} - K\rho_{ij}(u)||^2 + \alpha||\nabla u||, \tag{3.43}$$

see also [300]. K is the imaging operator, which is here a downsampling operator. The total variation regularization encourages a piecewise-constant depth solution.

The energy can be solved with the proposed method, because we can identify (3.43) with the previously introduced special case of a sum of several composite terms (3.28) and a total variation regularization (3.38) - we can find a majorizer in each iteration that can be solved by functional lifting.

In practice we solve all subproblems with sub-label accurate lifting as described in [197]. We initialize the algorithm with a constant depth of 1m and then iteratively update the nonconvex majorizer and solve the sublabel-accurate lifting problem. We use a primal-dual algorithm [54] to solve our subproblems and 'warm start' each inner iteration with the primal-dual variables from the previous step. We note



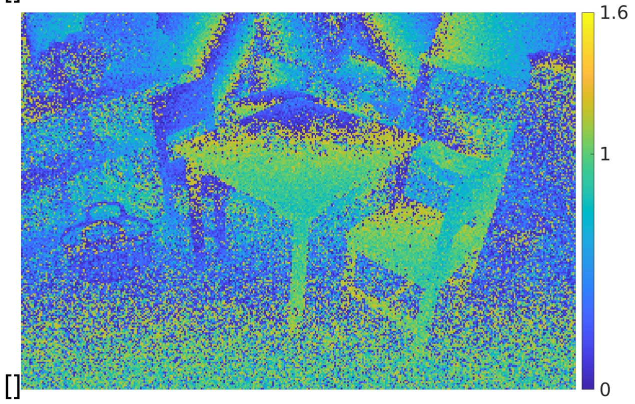


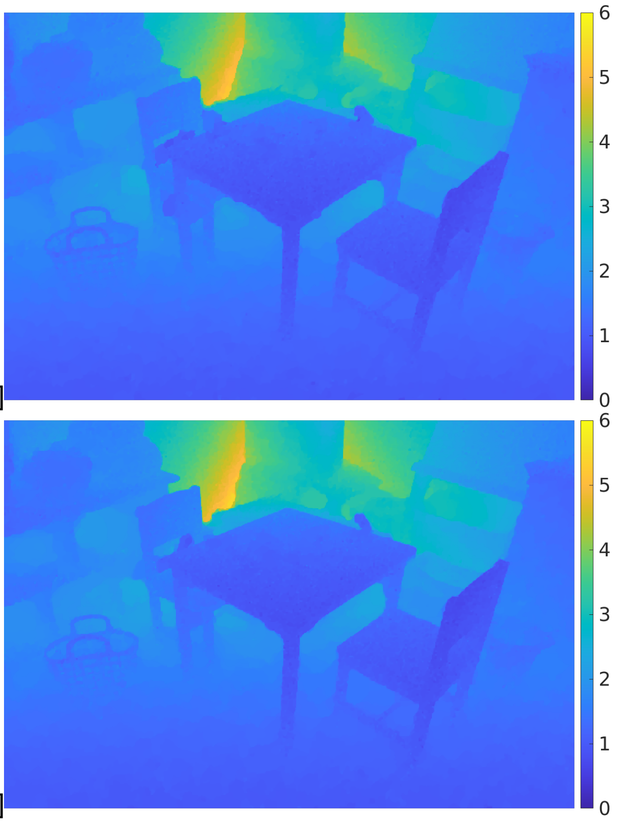
Figure 3.11: Classical closed form solution to depth recovery [162]. 90 MHz data to the left and 120 MHz to the right.

that this relaxation approach can possibly produce solutions that are convex combinations of global minimizers. To mitigate this problem, we monitor the energy of our inner iterations and terminate the algorithm if lifting cannot successfully minimize the majorizer, i.e. if any iterates u^{k+1} were to violate $E_{u^k}(u^{k+1}) \leq E(u^k)$, which was postulated in Assumption A. However such a violation could not be detected for the Time-of-Flight experiment shown here.

To test this procedure experimentally we generate synthetic data from a depth image of the Middlebury dataset [253] by applying (3.42). As a model for g_i we use the autocorrelation of a trapezoid signal. The autocorrelation signal and the ground truth depth are shown in Figure 3.9. We then apply downsampling by a factor of 2 to model the limited sensor size and add significant Gaussian noise to model the sensitivity of common ToF sensors. We generate two difference measurements in two frequencies, 90 MHz and 120 MHz. The ground truth data covers a depth ranging from 0.5 to 6m. The resulting measurements are outside the unambiguous range of both frequencies, so we expect a wrapping of data, which we want to resolve using both frequencies. We visualize the resulting four difference measurements in Figure 3.10. The noise level and severe data wrapping are apparent.

A classical inversion of the given data by the nonlinear closed form solution for sinusoidal data [162] is shown for each frequency in Figure 3.11. The solution is however contaminated by the nonlinear effects of noise and severe wrapping, note that we adjusted the colors for visualization purposes. Further heuristics would be required in a next step to combine both solutions to a final result, but we omit these due to the already significant distortion.

In contrast the solution by our algorithm is shown in subsection 3.5.2. For reference, the solution to the algorithm, initialized with the ground truth data is also visualized in subsection 3.5.2. We see that the recovered solution is near-optimal. The algorithm can accurately unwrap and upsample most of the



[]

[]

Figure 3.12: Solution by the proposed algorithm with lifted subproblems, upsampling factor of 2 to the left. To the right, proposed algorithm initialized with the ground truth, Figure 3.9, for reference.

depth data and only small areas around the left chair are misidentified.

Remark 3.21. The presented model assumes the knowledge of the signal amplitude a_i in each frequency by some preceding algorithm to streamline the presentation. If this information cannot be obtained robustly in practice, then the problem still falls into the problem category discussed in this paper, only the optimization variable $v = [u, a]$ is then vector-valued in depth and amplitude at each pixel (see the maximal generalization discussion in subsection 3.4.1). Yet, vector-valued variables can still be accounted for efficiently via the vectorial lifting shown in the works [269, 165]. The overall algorithm remains unchanged, only the subproblems are solved by vector-valued lifting instead.

We close this section by mentioning briefly that the presented composition of a matrix and a nonlinear wrapping operator is not entirely unique to Time-of-Flight reconstruction. A very related energy is present in nonlinear MRI reconstruction, see [291] for further reading.

3.6 Conclusions

In conclusion we proposed an optimization strategy for composite problems with simple, but highly-nonlinear, inner functions and L-smooth adaptable outer functions. We construct nonconvex majorizing functions and show that an iterative minimization of these functions leads to the convergence of energy values under weak assumptions as well as the convergence of the iterates to critical points of the energy under more restrictive assumptions. Our approach has several attractive properties. It generates a set of feasible iterates and it is very easy to use large step-sizes analytically, as these are independent of the Lipschitz properties of the inner function. Our convergence results naturally extend previous work. In practice, we extensively analyze the algorithm on synthetic examples, where the sub-problems can be solved globally and show that it can find better minima than related methods. Lastly we show an intended application. The use of recent functional lifting techniques to solve the nonconvex majorizer, critically allows us to find visually appealing solutions to the complicated composite problem of time-of-flight reconstruction from noisy low resolution data.

This new appendix contains additional unpublished material regarding Geiping & Moeller [2018][104].

3.A Reformulation as continuous Majorizer

In the main text, only discrete composite optimization problems such as (3.28) where discussed. However, the proposed algorithm can also be applied to Fredholm-type integral-operator problems such as, for example

$$E(u) = \int_\Omega G(x, \mathcal{K}(u)(x))\, dx + \int_\Omega |Du|\, dx \tag{3.44}$$

for $G \in C^1(\Omega)$ and $\mathcal{K}$ a well defined nonlinear integral operator and the measure $|Du|$ is an example for a regularization. This allows us to consider sub-problems analogous to (3.15):

$$\begin{aligned} E_{\bar{u}}(u) = &\int_{\Omega\times\Omega} D_{h_G}\left(k(x,y,u(y)), k(x,y,\bar{u}(y))\right)\, dy\, dx \\ &+ \int_{\Omega\times\Omega} \frac{\partial G}{\partial x}(x, \mathcal{K}(\bar{u})(x)) \cdot \left(k(x,y,u(y)) - k(x,y,\bar{u}(y))\right)\, dy\, dx + \int_\Omega |Du| \end{aligned}$$

but we can simplify this to

$$\begin{aligned} E_{\bar{u}}(u) = &\int_{\Omega\times\Omega} D_{h_G}\left(k(x,y,u(y)), k(x,y,\bar{u}(y))\right)\, dx\, dy \\ &+ \int_\Omega \left\{ \frac{\partial G}{\partial x}(x, \mathcal{K}(\bar{u})(x)) \int_\Omega k(x,y,u(y)) - k(x,y,\bar{u}(y))\, dx \right\} dy + \int_\Omega |Du| \end{aligned}$$

which can be solved globally by lifting as the inner parts can be written to be depending on a pointwise nonlinearity $\rho(y, u(y))$ and $|Du|$ is convex in Du, assuming integrablility w.r.t to this new ρ.

3.B Details regarding Grid Search

Univariate functions ρ can be optimized by grid search with parabolic refinement as mentioned in the main work. This appendix details algorithms for this usecase.

This is the case if the majorizer $E_{u^k}(u)$ is non-convex, but separable into univariate ρ. Furthermore, we assume some Lipschitz property on ρ and R and a bounded domain.

Uniform Grid Search: [204] Given a function $e : [a,b] \to \mathbb{R}$ with Lipschitz constant L for a nonempty , bounded interval $[a,b]$ and a desired accuracy $\varepsilon > 0$.

- Set $p = \lfloor \frac{L}{2\varepsilon} \rfloor = p$ and construct a uniform grid of $p+1$ trial points x_i via $x_i = \frac{i}{p}(b-a)$ for $i = 0, \dots, p$.
- Find $i^* \in \arg\min_i e(x_i)$ and set $\bar{x} = x_{i^*}$

As a result of applying this search we reach a near-optimal function value:

Lemma 3.22 ([204, Theorem 1.1.1]). *Let e^* be the optimal value of the given function $e : [a,b] \to \mathbb{R}$ with Lipschitz constant L. Then the function value of $\bar{x}$, the result of uniform grid search is near-optimal as*

$$e(\bar{x}) - e^* \leq \frac{L}{2p}$$

For our purposes, we combine this step with a parabolic curve fitting:

Refined Grid Search [181, pp. 217, 224]: Given are functions $\rho(x), r(x) : [a,b] \to \mathbb{R}$, $d_h(x,y) : \mathbb{R}^{\to}\mathbb{R}$ and $g'(x) : \mathbb{R} \to \mathbb{R}$, a parameter p denoting the granularity of grid search and a parameter ε denoting the accuracy.
Construct a uniform grid of p trial points x_i via $x_i = \frac{i}{p-1}(b-a)$ for $i = 0, \dots, p$ and look-up tables $[\rho(x_1), \dots, \rho(x_p)]$ and $[r(x_1), \dots, r(x_p)]$. For given x^0 initialize $z^0 = \rho(z^0)$.
Then during the iterations of the main problem solve the following sub-problem for each 1D sub-problem of $E_{u^k}(u)$:

- Set $e_k(x) = d_h(\rho(x), z^k) + g'(z^k)\rho(x) + r(x)$
- Construct the vector E_k with entries $e_k(x_i)$ for $i = 1, \dots, p$ from the look-up tables for ρ and r and find $i^* \in \arg\min_{1 \le i \le p}[E_k]_i$
- Initialize the triple $(x_{i^*-1}, x_i, x_{i^*+1}) =: (x_1, x_2, x_3)$ and iterate the following for m iterations:
  - Fit a parabola through x_1, x_2, x_3 and minimize it to find x_4
  - Evaluate $e_k(x_4)$ and update the triple to a new triple (x_1', x_2', x_3') so that $e_k(x_1') \ge e_k(x_2') \le e_k(x_3')$ and $x_1' \le x_2' \le x_3'$

  then set $x^{k+1} = x_2$ and add the evaluations of $\rho(x_4)$ and $r(x_4)$ to the look-up table, setting p to $p + m$ and set $z^{k+1} = \rho(x_2)$.
- Return the triple $(x^{k+1}, z^{k+1}, e(x^{k+1}))$.

By virtue of this construction we have that $e_k(u^{k+1}) \le e_k(u^k)$ and subsequently $E_{u^k}(u^{k+1}) \le E(u^k)$ and that dist$(0, \partial E_{u^k}(u^{k+1}))$ is small for sufficient m. Also note that, for the special case of $m = 1$, the number of evaluations of ρ, r is independent of the number of iterations in the outer problem. The refined grid search is highly attractive for separable problems, due to its potential for parallelization. All subproblems can be solved in parallel. Further, the usually most demanding task in each subproblem, the $\mathcal{O}(p)$ complex minimization of the vector e_k can also be parallelized.

CHAPTER 4

Parametric Majorization for Data-Driven Energy Minimization Methods

Contextualization

This chapter reprints the publication "Parametric Majorization for Data-Driven Energy Minimization Methods", published as conference publication at the International Conference on Computer Vision (ICCV 2019) with co-author Michael Moeller (Geiping & Moeller [2019]).

Previous works, such as Chapter 2 and Chapter 3, as well as Geiping et al., Görlitz et al. [2018, 2019], considered only the problem of solving an energy minimization problem that was already given - modeled by statistical considerations and handcrafted knowledge. With the advent of deep learning in computer vision and the growing success of learning, this position of relying only on handmade variational models seemed untenable - modern deep learning techniques are able to learn and improve many aspects of feed-forward models relying on the growing corpus of image data available, for example, via the internet. Unfortunately there is a catch for energy-based models: Deep neural networks as introduced in (1.4) are generally single-level feed-forward functions that can be learned directly as parametrized mappings $n(y, \theta)$ from inputs to known outputs. Energy models on the other hand, are only implicitly parametrized (through the energy function) and thus learning an energy model formally requires the solution to the bilevel optimization problem shown in (1.6). This is generally difficult in practice and especially for non-differentiable energy models it is unclear how to proceed.

Michael Moeller posed the initial strategy of approximately learning energy models by minimizing by (sub)-gradient penalties, which reappears as a special case in (4.17), but it was not yet clear when and why such an approximation strategy would be successful and how it could be improved. Jonas Geiping constructed the theoretical considerations in section 4.3, where we find that a 'collapse' of these approximations can be avoided if they are constructed and constrained to be majorizers of the full bilevel optimization problem, in analogy to composite majorization considered in the previous chapter, i.e. Geiping & Moeller [2018], where the outer function is now the upper-level bilevel problem and the inner function is the solution operator of the energy model. In further analogy we then introduce iterative majorizers in subsection 4.3.4.

Interesting to machine learning practitioners is that such a majorization strategy naturally recovers the contrastive loss function, also known as perceptron loss or generalized SSVM training. An unpublished additional appendix was added to this thesis that clarifies this connection, especially considering maximum-margin principles. The connections to many works in bilevel literature are also discussed in a new extended related work section, also featured in the appendix, alongside additional details and extended experiments of the analysis operator learning example with filter visualizations.

Helpful guidance, suggestions and corrections were contributed by Michael Moeller throughout this work, who further proposed and implemented the example concerning CT reconstruction. Jonas Geiping proposed and proved the properties of approximate majorization and connections to contrastive loss terms, implemented the segmentation and analysis operator learning examples and contributed mainly to the writing of this publication. Open-source code for this work is publicly available at https://github.com/JonasGeiping/ParametricMajorization.

Abstract

Energy minimization methods are a classical tool in a multitude of computer vision applications. While they are interpretable and well-studied, their regularity assumptions are difficult to design by hand. Deep learning techniques on the other hand are purely data-driven, often provide excellent results, but are very difficult to constrain to predefined physical or safety-critical models. A possible combination between the two approaches is to design a parametric energy and train the free parameters in such a way that minimizers of the energy correspond to desired solution on a set of training examples. Unfortunately, such formulations typically lead to bi-level optimization problems, on which common optimization algorithms are difficult to scale to modern requirements in data processing and efficiency. In this work, we present a new strategy to optimize these bi-level problems. We investigate surrogate single-level problems that majorize the target problems and can be implemented with existing tools, leading to efficient algorithms without collapse of the energy function. This framework of strategies enables new avenues to the training of parameterized energy minimization models from large data.

4.1 Introduction

Energy minimization methods, also referred to as variational methods, are a classical tool in computer vision [245, 57, 77, 52]. The idea is to define a data-dependent cost function E that assigns a value to each candidate solution x. The desired optimal solution is then the target solution with the lowest energy value. This methodology has several advantages, for one, it is characterized by an *explicit model* - namely the energy function to be minimized - and an implicit inference method - how we compute the minimizer of this energy is a separate problem. This duality allows a fruitful analysis, leading to controllable methods with provable guarantees that are paramount in many critical applications [242, 239, 305]. Furthermore, explicit knowledge over the model structure allows for explainable and clear modifications when the method is applied in a related task [68].

Conversely, deep learning approaches [166], specifically deep feed-forward neural networks work by very different principles. The methodology of deep learning is characterized by *implicit* models and explicit inference. The solution to the problem at hand is given directly by the output of the learned feed-forward structure. This is advantageous in practice and crucial for the efficient training of neural networks, however the underlying model of the problem structure is now only implicitly contained in the responses of the network. Deep neural networks have fundamentally changed the state-of-the-art in various computer vision applications, due to these properties as the inference operations are learned directly from large amounts of training data. These approaches are able to learn expressive and convincing mechanisms, examples of which can be found not only in recognition tasks (e.g. [155]), but also in denoising [308], optical flow [190, 129] or segmentation tasks [178, 243, 63]. Yet, as the underlying model is only implicitly defined and 'hidden' in the network structure, it is difficult to modify it for applications in other domains or to guarantee specific outputs. Domain adaptation is still an active field of research and several examples, for instance in medical imaging [8, 94], have demonstrated the need for possibly model-based physically plausible output restrictions. This problem is most strikingly demonstrated by the phenomenon of adversarial examples [275] - the existence of input data, that, when fed through the network, leads to highly erroneous solutions. While one would expect that such behavior is possibly unavoidable in recognition tasks [257, 194], it should not be a factor in low-level computer vision applications.

Reviewing these two methodologies, we would - of course - prefer to have the best of both worlds. We would like to use both the large amounts of data at our disposal and our far-reaching domain knowledge in many tasks to train explicit models with a significant number of free parameters, so that their optimal solutions are similar to directly trained feed-forward networks.

A promising candidate for such a combination of learning- and model based approaches are *parametrized energy minimization methods*. The idea of such methods is to define an energy E that depends on the candidate solutions x, the input data y and parameters θ,

$$\begin{aligned} E : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^s &\to \mathbb{R}, \\ (x, y, \theta) &\mapsto E(x, y, \theta), \end{aligned} \tag{4.1}$$

such that for a good choice of parameters θ, the argument $x(\theta) = \arg\min_x E(x, y, \theta)$ that minimizes the energy over all x is as close a possible to the desired true solution x^*.

To train such parametric energies, assume we are given N training samples $\{(x_i^*, y_i)\}_{i=1}^N$ and a continuous *higher-level* loss function $l : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, which measures the deviation of solutions of the model to the given training samples. Determining the optimal parameters θ then becomes a *bi-level optimization problem* combining both the higher-level loss function and the lower-level energy,

$$\min_{\theta \in \mathbb{R}^s} \sum_{i=1}^N l(x_i^*, x_i(\theta)), \tag{4.2}$$

$$\text{subject to} \quad x_i(\theta) = \arg\min_{x \in \mathbb{R}^n} E(x, y_i, \theta). \tag{4.3}$$

Usual first-order learning methods are difficult to apply in this setting. For every gradient computation it is necessary to compute a derivative of the $\arg\min$ operation of the lower-level problem, which is even further complicated if we consider parametrized non-smooth energy models which are wide-spread in computer vision [77, 52].

Therefore, the goal of this paper is to analyze bi-level optimization problems and identify strategies that allow for efficient approximate solutions. We investigate single-level minimization problems with simple constraints without second-order differentiation, which are applicable even to non-smooth energies. Such forms allow scaling the previously limited training of energy minimization methods in computer vision to larger datasets and increase the effectiveness in applications where it is critical that the solution follows a specific model structure.

In the remainder of this paper we analyze the bi-level optimization problem to develop a rigorous understanding of sufficient conditions for a single-level surrogate strategy for continuous loss functions l and convex, non-smooth lower-level energies E to be successful. We introduce the concept of a *parametric majorization function*, show relations to structured support vector machines and provide several levels of parametric majorization functions with varying levels of exactness and computational effort. We extend our approximations to an iterative scheme, allowing for repeated evaluations of the approximation, before illustrating the proposed strategies in computer vision applications.

4.2 Related Work

The straightforward way of optimizing bi-level problems is to consider *direct descent methods* [153, 251, 74]. These methods directly differentiate the higher-level loss function with respect to the minimizing argument and descend in the direction of this gradient. An incomplete list of examples in image processing is [47, 68, 66, 67, 79, 80, 110, 119, 120]. This strategy requires both the higher- and lower-level problems to be smooth and the minimizing map to be invertible. This is usually facilitated by implicit differentiation, as discussed in [249, 157, 67, 68]. In more generality, the problem of directly minimizing θ without assuming that smoothness in E leads to optimization problems with equilibrium constraints (MPECs), see [26] for a discussion in terms of machine learning or [83, 82, 84] and [74]. This approach also applies to the optimization layers of [7], which lend themselves well to a reformulation as a bi-level optimization problem.

Unrolling is a prominent strategy in applied bi-level optimization across fields, i.e. MRF literature [14, 189] in deep learning [312, 64, 58, 173] and in variational settings [215, 164, 163, 113, 114, 238]. The problem is transformed into a single level problem by choosing an optimization algorithm $\mathcal{A}$ that produces an approximate solution to the lower level problem after a fixed number of iterations. $x(\theta)$ is then replaced by $\mathcal{A}(y, \theta)$. Automatic differentiation [111] allows for an efficient evaluation of the gradient of the upper-level loss w.r.t to this reduced objective

$$\min_{\theta} \sum_{i=1}^{N} l(x_i^*, \mathcal{A}(y_i, \theta)). \tag{4.4}$$

In general these strategies are very successful in practice, *because* they combine the model and its optimization method into a single feed-forward process, where the model is again only implicitly present. Later works [69, 65, 113, 114] allow the lower-level parameters to change in between the fixed number of iterations, leading to structures that model differential equations and stray further from underlying modelling. As pointed out in [146], these strategies are more aptly considered as a set of nested quadratic lower-level problems.

Several techniques have been developed in the field of structured support vector machines (SSVMs) [282, 72, 5, 287] that are very relevant to the task of learning energy models, as SSVMs can be understood as bi-level problems with a lower-level energy that is linear in θ and often a non-continuous higher-level loss. Various strategies such as margin rescaling [282], slack rescaling [287, 297], softmax-margins [107] exist and have also been applied recently in the training of computer vision models in [147, 73], we will later return to their connection to the investigated strategies.

4.3 Bi-Level Learning

We now formalize our learning problem. We assume the lower-level energy E from (4.1) to be convex (but not necessarily smooth) in its first variable $x \in \mathbb{R}^n$ and to depend continuously on input data $y \in \mathbb{R}^m$ and parameters $\theta \in \mathbb{R}^s$. We assume its minimizer $x(\theta)$ to be unique. For our higher-level loss function (4.2) $l : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, we assume that it fulfills $l(x, y) \geq 0, l(x, x) = 0$ for all x, y and is differentiable in its second argument.

Note that this formulation of bi-level optimization problems directly generalizes classical supervised (deep) learning with a network $\mathcal{N}(\theta, y)$ via the quadratic energy $E(x, y_i, \theta) = \frac{1}{2}||x - \mathcal{N}(\theta, y_i)||^2$, for which $x_i(\theta) = \mathcal{N}(\theta, y_i)$.

Preliminaries (Convex Analysis): Let us summarize our notation and some fundamental results from convex analysis. We refer the reader to [21] for more details. We denote by $\partial E(x)$ the set of subgradients of a convex function E at x. We define the Bregman distance between two vectors relative to a convex function E by $D_E^p(x, y) = E(x) - E(y) - \langle p, x - y\rangle$ for a subgradient $p \in \partial E(y)$, intuitively the Bregman distance measures the difference of the energy at x to its linear lower bound around y. $E^*(p) = \sup_x \langle p, x\rangle - E(x)$ is the convex conjugate of E. x is a minimizer of the energy E if and only if $0 \in \partial E(x)$ or equivalently by convex duality $x \in \partial E^*(0)$. E is m-strongly convex if $D_E^p(x, y) \geq \frac{m}{2}||x - y||^2$ for all x, y. Conversely, if E is m-strongly convex, then E^* is $\frac{1}{m}$-strongly smooth, i.e. $D_{E^*}(p, q) \leq \frac{2}{m}||p - q||^2$. Furthermore $D_E^p(x, y) = D_{E^*}^x(p, q), q \in \partial E(x)$ holds for all Bregman distances [44]. We consider parametrized energies in several variables, yet we always assume (sub)-gradients, Bregman distances and convex conjugates to be with respect to the first argument x.

4.3.1 Majorization of Bi-level Problems

As previously discussed, directly solving the bi-level problem as posed in (4.2) and (4.3) is tricky. We need to implicitly differentiate the minimizing argument $x_i(\theta)$ for all N samples just to apply a first-order method in θ - which is in stark contrast to our goal of finding efficient and scalable algorithms.

Let us instead look at the problem from a very different angle and entertain the idea that the loss function l is actually of secondary importance to us. We really only want to find parameters θ so that our training samples are well reconstructed, $x_i^* \approx x_i(\theta)$. If we go so far as to assume that the loss value of our optimal parameters θ^* is zero, meaning that minimizers of our energy are perfectly able to reconstruct our training samples, then the bi-level problem is reduced to a single-level problem, inserting $x_i^* = x_i(\theta^*)$:

$$\min_\theta \quad \text{s.t. } 0 \in \partial E(x_i^*, y_i, \theta), \tag{4.5}$$

which we could solve via

$$\min_\theta \sum_{i=1}^N ||q_i||^2 \quad \text{s.t. } q_i \in \partial E(x_i^*, y_i, \theta) \tag{4.6}$$

This train of thought is closely interconnected to the notion of separability in Support Vector Machine methods [292], where it is assumed that given training samples are linearly separable, which is equivalent to assuming that the classification loss is zero on the training set.

However minimizing (4.6) is often not a good choice. A simple example is $E(x, y, \theta) = (\theta x - y)^2$, i.e. we simply try to learn a positive scaling factor θ between x and y. Problem (4.5) can then be written as $\min_\theta \sum_i (\theta^2 x_i^* - \theta y_i)^2$ and is trivially minimized by $\theta = 0$. Such a solution makes E independent of x such that every x becomes a minimizer. This phenomenon is referred to as *collapse* of the energy function [168, 167] in machine learning literature, and clearly cannot be a good strategy to learn a scaling factor.

Interestingly, the scaling problem can be reformulated into a reasonable (non-collapsing) problem, if we require (4.6) to *majorize* the bilevel problem: If we consider the higher-level loss function $l(x_i^*, x_i(\theta)) = (x_i^* - x_i(\theta))^2$, then our surrogate problem $\sum_i (\theta^2 x_i^* - \theta y_i)^2$ is clearly not a majorizer for arbitrary θ. However, if we consider a reformulation of the energy to $E(x) = (x - \frac{1}{\theta} y)^2$, then this reformulation leads to a *majorizing* surrogate $\sum_i (x_i^* - \frac{1}{\theta} y_i)^2$. Minimizing θ now leads to learning the desired scaling factor.

Our toy example motivates us to formalize the concept of majorizing surrogates:

Definition 4.1 (Parametrized Majorizer). *Given a bi-level optimization problem in the higher level loss* $l(x, y)$ *and lower-level energy* $E(x, y, \theta)$*, we call the function* $S(x, y, \theta) : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^s \to \mathbb{R}$ *a parametrized majorizer, if*

$$\begin{aligned} &\forall \theta \in \mathbb{R}^s: && l(x, x(\theta)) \leq S(x, y, \theta) \\ &\forall \theta \in \mathbb{R}^s \quad \text{s.t.} && l(x, x(\theta)) = 0 \implies S(x, y, \theta) = 0 \end{aligned}$$

hold for any $x, y \in \mathbb{R}^n \times \mathbb{R}^m$.

This definition allows us to formalize our objective further. We investigate replacing the bi-level optimization problem (4.2), (4.3) by the minimization of a suitable parametrized majorizer, i.e.

$$\min_{\theta \in \mathbb{R}^s} \sum_{i=1}^N S(x_i^*, y_i, \theta). \tag{4.7}$$

An immediate conclusion of Definition 4.1 is that the function S now certifies our progress as $S(x, y, \theta) = 0$ implies $l(x, x(\theta)) = 0$. Moreover, our goal is to choose majorizers S in such a way that they yield **single-level** problems (4.7), meaning it is not necessary to differentiate an $\arg\min$ operation to minimize them or to solve an equally difficult reformulation, making them significantly easier to solve.

4.3.2 Single-Level Majorizers

One possible way to find a majorizer that satisfies the previously postulated properties is by considering the majorizer naturally induced through the Bregman distance of the lower level energy. We assume the following condition

$$l(x, z) \leq D_{E_\theta}(x, z) \quad \forall x, z \in \mathbb{R}^n, \theta \in \mathbb{R}^s, \tag{4.8}$$

and propose the surrogate problem

$$\min_{\theta} \sum_{i=1}^{N} D_{E_\theta}\left(x_i^*, x_i(\theta)\right). \tag{4.9}$$

Condition (4.8) is an assumption on both the loss function and the energy. It thus delineates the class of bi-level problems that can be attacked with this majorization strategy. However this condition is quite general. For a large class of loss functions, we only need the energy to contain a term that also induces the loss function, a property also known as (relative) strong convexity [285, 180]:

Proposition 4.2. *If the loss function $l(x,y)$ is a Bregman distance induced by a strictly convex function $w : \mathbb{R}^n \to \mathbb{R}$, i.e. $l(x,y) = D_w(x,y)$, then assumption (4.8) is fulfilled if the energy E is w-strongly convex, i.e. if $E(x) - w(x)$ is still a convex function.*

Proof: We write E as $E(x) = \hat{E}(x) + w(x)$ and apply the additive separability of Bregman distances to find $D_E(x,y) = D_{\hat{E}}(x,y) + D_w(x,y)$, which is greater than or equal to $D_w(x,y)$, as $D_{\hat{E}}(x,y)$ is non-negative due to the convexity of $\hat{E}$. For the usual euclidean loss, this property reduces to strong convexity:

Example 4.3. *If the loss function is given by a squared Euclidean loss, $l(x,y) = \frac{1}{2}||x-y||^2$ and the energy is m-strongly convex, then assumption (4.8) is fulfilled for the energy $\frac{1}{m}E$.*

The question remains whether the proposed surrogate problem (4.9) is efficiently solvable. We especially wanted to circumvent the differentiation of $x(\theta)$. However $D_E\left(x_i^*, x_i(\theta)\right)$ is much easier to solve, in comparison to the original bi-level problem, as we can see in both its primal and its dual formulation. First, from a primal viewpoint, we have

$$\begin{aligned} &D_E\left(x_i^*, x_i(\theta)\right) \\ =&E(x_i^*, y_i, \theta) - E(x_i(\theta), y_i, \theta) - \langle p_i, x_i^* - x_i(\theta)\rangle, \end{aligned}$$

for some subgradient $p_i \in \partial E(x_i(\theta))$ which we have not specified yet. But, as $0 \in \partial E(x_i(\theta))$ as $x_i(\theta)$ is by definition a solution to the lower-level problem, we may take $p = 0$ and simplify to

$$E(x_i^*, y_i, \theta) - E(x_i(\theta), y_i, \theta).$$

Now $x_i(\theta)$ is contained solely in E and we can write

Bregman Surrogate:

$$D^0_{E_\theta}\left(x_i^*, x_i(\theta)\right) = \max_{x \in \mathbb{R}^n} E(x_i^*, y_i, \theta) - E(x, y_i, \theta). \tag{4.10}$$

This surrogate function is already much simpler than the original bi-level problem. We can minimize (4.10) either by alternating minimization in θ and maximization in x or by jointly optimizing both variables. However, the problem is still set up as a saddle-point problem which is not ideal for optimization.

Remark 4.4. Interestingly, this discriminative formulation is not wholly unfamiliar. We can understand this as an appropriate generalization of generalized perceptron training [168, 167, 279] as discussed as far back as [244]. See the appendix for further details. In vein of this comparison, conditions 1 and 2 from e.g. [168], i.e. conditions on the existence of a margin between the optimal solution and other candidate solutions central to (S)SVM methods [292, 281, 284] are reflected in Proposition 4.2 in the convex continuous setting. Due to continuity of the energy and loss function we cannot obey a fixed margin, yet we impose that the energy grows at least as fast as the loss function, when we move away from the optimal solution.

We can resolve the saddle-point question by analyzing the surrogate (4.9) from a dual standpoint, as by Bregman duality [28]

$$D^0_{E_\theta}(x_i^*, x_i(\theta)) = D^{x_i^*}_{E_\theta^*}(0, q_i) \tag{4.11}$$

for $q_i \in \partial E(x_i^*, y, \theta)$. Contrasting this formulation with our initial goal of penalizing the subgradient as in (4.6), we see that the Bregman distance induced by E^* is the natural 'distance' by which to penalize the subgradient in the sense that penalizing the subgradient at x_i^* with this generalized distance recovers a majorizing surrogate.

We can further simplify the dual formulation by applying Fenchel's theorem:

$$D^{x_i^*}_{E_\theta^*}(0, q_i) = E(x_i^*, y_i, \theta) + E^*(0, y_i, \theta). \tag{4.12}$$

Computing $E^*(0)$ is exactly as difficult as minimizing E (as $E^*(0) = \min_x E(x)$), so we need to rewrite this surrogate in a tractable manner. To do so, we assume that E can be additively decomposed into two parts,

$$E(x, y, \theta) = E_1(x, y, \theta) + E_2(x, y, \theta), \tag{4.13}$$

where both E_1 and E_2 are convex in their first argument and their convex conjugates are simple to compute. Exploiting that $E^*(0) = \min_z E_1^*(-z) + E_2^*(z)$ yields

$$D^{x_i^*}_{E_\theta^*}(0, q_i) = \min_{z \in \mathbb{R}^n} E(x_i^*, y_i, \theta) + E_1^*(-z, y, \theta) + E_2^*(z, y, \theta). \tag{4.14}$$

In comparison to the primal formulation in (4.10), we have now reformulated the problem from a saddle point problem (minimizing in θ and maximizing in x) to a pure minimization problem, which is easier to handle. This is a generalization of the dual formulation discussed in the linear context of SSVMs for example in [281, 284].

However for both variants we still need to handle an auxiliary variable. We can trade some of this computational effort for a weaker majorizer by making specific choices for z in (4.14). To illuminate these choices we introduce the function $W_E(p, x) = E^*(p) + E(x) - \langle p, x \rangle$ [237, 46], which allows us to write

$$D^{x_i^*}_{E_\theta^*}(0, q_i) = \min_{z \in \mathbb{R}^n} W_{E_1,\theta}(-z, x_i^*) + W_{E_2,\theta}(z, x_i^*). \tag{4.15}$$

Note that $W_E(p, x) = 0$ if $p \in \partial E(x)$. As such choosing either $-z \in \partial E_1(x_i^*)$ or $z \in \partial E_2(x_i^*)$ allows us to simplify the problem further. This is especially attractive if E is differentiable, as then both surrogates can be computed without auxiliary variables. We will denote these as *partial* surrogates, owing to the fact that we minimize only one term in (4.15)

Partial Surrogate:

$$\min_{z \in \partial E_2(x_i^*, y_i, \theta)} W_{E_1,\theta}(-z, x_i^*). \tag{4.16}$$

Effectively, this reduces the requirements of (4.14), as only the convex conjugate of E_1 needs to be computed. By symmetry, the other partial surrogate follows analogously.

We can finally also return to the previously discussed gradient penalty (4.6). If our energy E is $m(\theta, y)$-strongly convex, then its convex conjugate is strongly smooth and we can bound the dual formulation (4.11) via

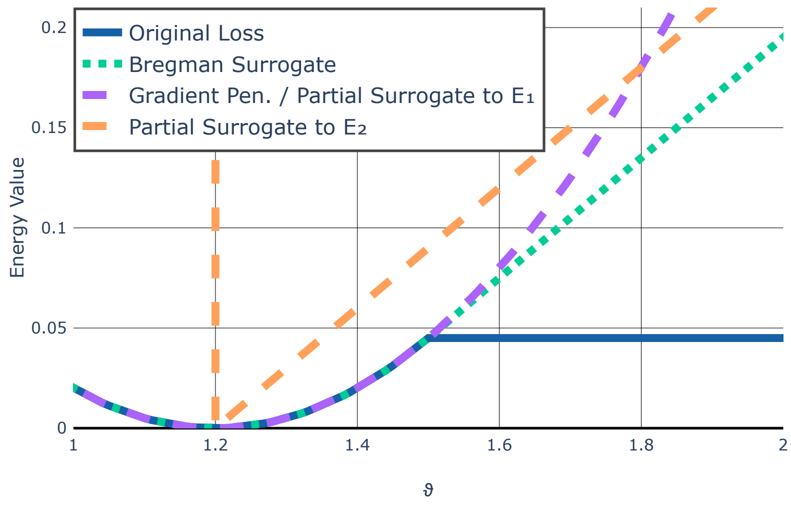


Figure 4.1: Visualization of surrogate functions for the bi-level problem given in (4.18). The blue line marks the original bi-level problem, the green dots marks the Bregman distance surrogate discussed in (4.10). The orange curve marks the partial surrogate obtained from (4.15) by inserting $z = \nabla E_1(x^*)$, whereas the purple line marks the other partial surrogate (4.16) which is equivalent to the gradient penalty (4.17) here.

Gradient Penalty

$$\frac{1}{m(\theta, y_i)} ||q_i||^2 \quad \text{s.t. } q_i \in \partial E(x_i^*, y_i, \theta). \tag{4.17}$$

While this formulation allows us to minimize an upper bound on the bi-level problem without either auxiliary variables or knowledge about E_1^* or E_2^*, it also is the crudest over-approximation among the considered surrogates as the following proposition illustrates.

Proposition 4.5 (Ordering of parametric majorizers). *Assuming the condition $l(x, z) \leq D_{E_\theta}(x, z)$ from (4.8), we find that the presented parametric majorizers can be ordered in the following way:*

$$\begin{aligned} l(x_i^*, x(\theta)) &\leq D_{E_\theta}^0(x_i^*, x_i(\theta)) = D_{E_\theta^*}^{x_i^*}(0, q_i) \\ &\leq \min_{z \in \partial E_2(x_i^*)} W_{E_1}(-z, x_i^*) \\ &\leq \frac{1}{m(\theta, y)} ||q_i||^2 \quad \text{s.t. } q_i \in \partial E(x_i^*, y, \theta). \end{aligned}$$

The Bregman surrogate (4.10) majorizes the original loss function and is in turn majorized by the partial surrogate (4.16) which is majorized by the gradient penalty (4.17) under the assumption of strong convexity.

Proof. See appendix. ■

As a clarifying example, we can simplify these majorizers in the differentiable setting:

Example 4.6 (Differentiable Energy). *Let E be differentiable and $m(\theta, y)$-strongly convex, then the majorizers in Proposition 4.5 are given by*

$$\begin{aligned} l(x_i^*, x(\theta)) &\leq D_{E_\theta}(x_i^*, x_i(\theta)) = D_{E_\theta^*}(0, \nabla E(x_i^*, y_i, \theta)) \\ &\leq W_{E_1}(-\nabla E_2(x_i^*), x_i^*) \\ &\leq \frac{1}{m(\theta, y)} ||\nabla E(x_i^*, y_i, \theta)||^2. \end{aligned}$$

4.3.3 Intermission: One-Dimensional Example

Let us illustrate our discussion with a toy example. We consider the non-smooth bi-level problem of learning the optimal sparsity parameter θ in the bi-level problem:

$$\min_{\theta \in \mathbb{R}} \frac{1}{2} |x^* - x(\theta)|^2, \tag{4.18}$$

$$\text{subject to} \quad x(\theta) = \arg\min_x \frac{1}{2}|x - y|^2 + \theta |x|. \tag{4.19}$$

As the lower-level energy is 1-strongly convex and the upper level loss is quadratic $l(x, y) \leq D_{E_\theta}(x, y)$ holds. Detailed derivations of all three surrogate functions of this example can be found in the appendix. Figure 4.1 visualizes these surrogates, plotting their energy values relative to θ. Due to the low dimensionality of the problem, all surrogate functions coincide with the original loss function at the optimal value of θ. It is further interesting to note that the Bregman surrogate is exactly identical with the original loss function in the vicinity of the optimal value, due to the low dimensionality of the example.

4.3.4 Iterative Majorizers

We used subsection subsection 4.3.2 to construct a series of upper bounds to facilitate a trade-off between efficiency and exactness. However what happens if we are not satisfied with the exactness of the Bregman surrogate (4.9)? This setting can happen especially if x^* and $x(\theta)$ are significantly incompatible and subsequently $l(x^*, x(\theta))$ is large, even for optimal θ. For example if we try to optimize only a few hyper-parameters we might not at all expect $x(\theta)$ to be close to x^*. This discussion can again be linked to the notion of 'separability' in SVM approaches [292]: The quality of the majorizing strategy is directly related to the level of 'separability' of the bi-level problem.

However, we can use the previously introduced majorizers iteratively. To do so we need to develop a majorizer that depends on a given estimate $\bar{x}$.

Proposition 4.7. *Under the standing assumption that $l(x, y) \leq D_{E_\theta}(x, y)$ (4.8) and if the loss function is induced by a strictly convex function $w : \mathbb{R}^n \to \mathbb{R}$, i.e. $l(x, y) = D_w(y, x)$, we have the following inequality:*

$$l(x, y) \leq l(x, z) + \langle \nabla_z l(x, z), y - z \rangle + D_E(z, y). \tag{4.20}$$

Proof. It holds that $l(x, y) = D_w(y, x)$ which is equivalent to $D_w(y, z) + D_w(z, x) - \langle \nabla w(x) - \nabla w(z), z - y \rangle$ by the Bregman 3-Point inequality [62, 285]. Using the standing assumption and that $\nabla w(x) - \nabla w(z) = \nabla_x D_w(x, z)$ we find the proposed inequality. ■

Assume we are given an estimated solution $\bar{x}_i$, then we can use this estimate to rewrite our bound to

$$\begin{aligned} l(x_i^*, x_i(\theta)) \leq & l(x_i^*, \bar{x}_i) + \langle \nabla l(x_i^*, \bar{x}_i), x_i(\theta) - \bar{x}_i \rangle \\ & + D_E(\bar{x}_i, x_i(\theta)). \end{aligned} \tag{4.21}$$

This is a linearized variant of the parametric majorization bound and as such a nonconvex composite majorizer in the sense of [104], as such a key property of majorization-minimization techniques remains in the parametrized setting, choosing $\bar{x}_i = x_i(\theta^k)$:

Proposition 4.8 (Descent Lemma). *The iterative procedure given by repeatedly minimizing the right-hand side of (4.21) in θ and setting $\bar{x}_i = x_i(\theta^k)$ is guaranteed to be stable, i.e. not to increase the bi-level loss:*

$$\sum_{i=1}^{N} l\left(x_i^*, x_i(\theta^{k+1})\right) \leq \sum_{i=1}^{N} l\left(x_i^*, x_i(\theta^k)\right) \tag{23}$$

Proof. See appendix. ■

However this algorithm cannot be applied directly, as we would still need to differentiate $x_i(\theta)$ appearing in the linearized part. Nevertheless, we can use both Fenchel's inequality $\langle p, x\rangle \leq E(x) + E^*(p)$ and the previously established $D_{E_\theta}(x, x(\theta)) = E(x, y, \theta) - E(x(\theta), y, \theta)$ to find an over-approximation to the iterative majorizer of Proposition 4.8:

$$\begin{aligned}
& l(x_i^*, x_i(\theta)) \\
\leq\ & l(x_i^*, \bar{x}_i) - \langle \nabla l(x_i^*, \bar{x}_i), \bar{x}_i\rangle \\
& \quad + E^*\left(\nabla l(x_i^*, \bar{x}_i), y_i, \theta\right) + E(x_i(\theta), y_i, \theta) \\
& \quad + E(\bar{x}_i, y_i, \theta) - E(x_i(\theta), y_i, \theta) \\
=\ & l(x_i^*, \bar{x}_i) - \langle \nabla l(x_i^*, \bar{x}_i), \bar{x}_i\rangle \\
& \quad + E(\bar{x}_i, y_i, \theta) + E^*\left(\nabla l(x_i^*, \bar{x}_i), y_i, \theta\right)
\end{aligned}$$

This estimate reveals that we can approximate the iterative majorizer much like the previously discussed surrogates:

Iterative Surrogate

$$E(\bar{x}_i, y, \theta) + E^*\left(\nabla l(x_i^*, \bar{x}_i), y_i, \theta\right) + C, \tag{4.22}$$

as the constant $C = l(x_i^*, \bar{x}_i) - \langle \nabla l(x_i^*, \bar{x}_i), \bar{x}_i\rangle$ does not depend on θ. We essentially return to (4.12) and only the input to E and E^* changes with respect to $\bar{x}_i$. This strategy recovers the previous majorizer as a special case:

Corollary 4.9. *If we linearize around $\bar{x}_i = x_i^*$, then we recover the Bregman surrogate of (4.9).*

Proof. If $\bar{x}_i = x_i^*$, then $l(x_i^*, \bar{x}_i) = 0$ and $\nabla l(x_i^*, \bar{x}_i) = 0$ by the properties of the differentiable loss function. As such the constant term C is zero and $E^*\left(\nabla l(x_i^*, \bar{x}_i), y_i, \theta\right) = E^*(0, y_i, \theta)$ so that we recover (4.12) which is equivalent to the Bregman surrogate (4.9). ■

We can use this surrogate to form an efficient approximation to a classical majorization-minimization strategy as in [273, 187, 186, 127]. Notably the 'tightness' of the majorization is violated by the over-approximation, i.e. inserting θ^k into the majorizer does not recover $l(x_i^*, x_i(\theta^k))$. We iterate

$$\begin{aligned}
\theta^{k+1} = \arg\min_\theta \sum_{i=1}^{N} & E^*\left(\nabla l(x_i^*, x_i(\theta^k)), y_i, \theta\right) \\
& + E\left(x(\theta^k), y_i, \theta\right)
\end{aligned} \tag{4.23}$$

As the application of this iterative scheme reduces to a simple change from (4.12) to (4.22), we can easily apply it in practice to further increase the fidelity of the surrogate by solving a sequence of fast surrogate optimizations. We initialize the scheme with $\bar{x}_i = x_i^*$ as suggested from Corollary 4.9 and either stop iterating or reduce the step size of the surrogate solver if the higher-level objective is increased after an iteration.

4.4 Examples

This section will feature several experiments[1] in which we will illustrate the application of the investigated methods. We will show two concepts of new applications that are possible in parametrized variational settings, subsection 4.4.1 and subsection 4.4.2. We then show an application to image denoising in subsection 4.4.3.

4.4.1 Computed Tomography

Making only specific parts of a variational model learnable is especially interesting for computed tomography (CT). An image x is to be reconstructed from data $y = Ax + n$ that is formed by applying the radon transform to the image x and adding noise n. While first fully-learning based solutions to this problem exist (e.g. [137, 141]), suitable networks are difficult to find not only due to the ill-posedness of the underlying problem, but also due to the well-justified concerns about fully learning-based approaches in medical imaging [8]. To benefit from the explicit control of the data fidelity of the reconstruction, we consider to introduce a learnable linear correction term into an otherwise classical reconstruction technique via

$$x_i(\theta) = \arg\min_x \frac{1}{2}\|Ax - y_i\|_2^2 + \beta R(x) + \langle x, \mathcal{N}(\theta, y_i)\rangle,$$

for a suitable network $\mathcal{N}$ (we chose 8 blocks of 3×3 convolutions with 32 filters, ReLU activations, and batch-normalization, and a final 5×5 convolution), and R denoting the Huber loss of the discrete gradient of x.

As both convex conjugates are difficult to evaluate in closed-form, we choose the gradient penalty (4.17), which is a parametric majorizer for euclidean loss if A has full rank (and practically even works beyond this setting, as it majorizes $||A(x-y)||^2$ even for rank-deficient A). According to (4.17) we consider

$$\min_{\theta \in \mathbb{R}^s} \sum_{i=1}^n \|A^*Ax_i^* - A^*y_i + \beta \nabla R(x_i^*) + \mathcal{N}(\theta, y_i)\|_2^2,$$

train on simulated noisy data and test our model on the widely-used Shepp-Logan phantom. Figure 4.2 illustrates the the resulting reconstruction, as well as the best reconstruction using the variational approach without the additional linear correction term after a grid-search for the optimal β. As we can see, the surrogate trained the linear correction term well enough to improve the PSNR of the reconstruction by almost 2dB. Moreover, the influence of the linear correction term can still be visualized and the data fidelity can easily be controlled via a suitable weighting. We visualize the correction map in the appendix.

4.4.2 Variational Segmentation

For a very different (and non-smooth) example, consider the task of learning a variational segmentation model [57, 53, 77, 207]. We are interested in learning a model whose minimizer coincides with a (semantic) segmentation of the input data. The lower-level problem is given by

$$x(\theta) = \arg\min_x -\langle \mathcal{N}(\theta, y), x\rangle + ||Dx||_1 + h(x), \tag{4.24}$$

where $h(x) = \sum_{j=1}^n x_i \log(x_i) + I_\Delta(x)$ is the entropy function on the unit simplex Δ [24]. $\mathcal{N}(\theta, y)$ is some parametrized function that computes the potential of the segmentation model, this can be a deep neural network, as we only require convexity in x and not in θ. D is a finite-differences operator, so that the overall total variation (TV) term $||Dx||_1$ measures the perimeter of a segmentation x if $x \in \{0, 1\}^n$. The entropy function crucially not only leads to a strictly convex model but also represents the structure of a usual learned segmentation method. Without the perimeter term, a solution to the lower-level problem would be given by

$$x(\theta) = \nabla h^*(\mathcal{N}(\theta, y)). \tag{4.25}$$

[1]An implementation of these experiments can be found at https://github.com/JonasGeiping/ParametricMajorization.

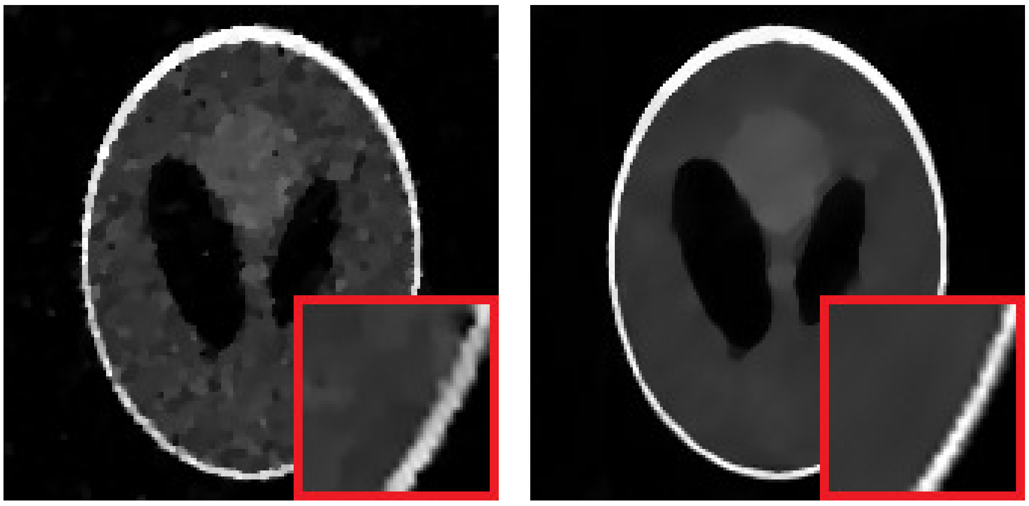


Figure 4.2: Learning a linear correction term for a Huber-regularized CT reconstruction problem using the gradient penalty (4.6).

Due to [240, P.148], ∇h^* is exactly the $\mathrm{softmax}$ function, so that (4.25) is equivalent to applying a parametrized function $\mathcal{N}$ and then applying the $\mathrm{softmax}$ function to arrive at the final output, a usual image recognition pipeline during training. As a higher-level loss, we choose $\log$ loss

$$\sum_{i=1}^{N} -\langle x_i^*, \log(x_i(\theta))\rangle = \sum_{i=1}^{N} D_h(x_i^*, x_i(\theta)) \tag{4.26}$$

so that the bi-level problem without the perimeter term is equivalent to minimizing the cross-entropy loss of $\mathcal{N}(\theta, y)$. With the inclusion of the perimeter term, however, we cannot find a closed-form solution for $x(\theta)$ need to consider bi-level optimization. But, as the log-loss (4.26) can be written as a Bregman distance relative to h, our primary assumption $l(x,z) \leq D_{E_\theta}(x,z)$ (4.8) is fulfilled and we can consider the Bregman surrogate problem in the dual setting of (4.14):

$$\min_\theta \sum_{i=1}^{N} \min_{z_i} W_h(\mathcal{N}(\theta, y_i) - z_i, x_i^*) + W_{TV}(z_i, x_i^*), \tag{4.27}$$

which we can rewrite to

$$\begin{aligned}\min_\theta \sum_{i=1}^{N} \min_{||p_i||\leq 1} h^* \left(\mathcal{N}(\theta, y_i) - D^T p_i\right) \\ - \langle \mathcal{N}(\theta, y_i), x_i^*\rangle + ||Dx_i^*||_1.\end{aligned} \tag{4.28}$$

We note that this is essentially a cross-entropy loss with an additional additive term p_i, that is able to balance out incoherent output of $\mathcal{N}(\theta, y_i)$ that would lead to erroneous segmentations with a higher perimeter. Furthermore, the training process is still convex w.r.t to $\mathcal{N}(\theta, y_i)$, in contrast to unrolling schemes. The iterative model (4.23) has a very similar structure, including the gradient of the loss into (4.28).

To validate this setup, we choose $\mathcal{N}$ to be given by a simple convolutional linear model. We draw a small subset of the `cityscapes` dataset and compare the cross entropy model of (4.25) with the total variation bi-level model of (4.28) and its partial and iterative applications. Figure 4.3 visualizes the training accuracy over training iterations. We find that the proposed approach is able to improve the segmentation accuracy of the linear model significantly. We refer to the appendix for further details.

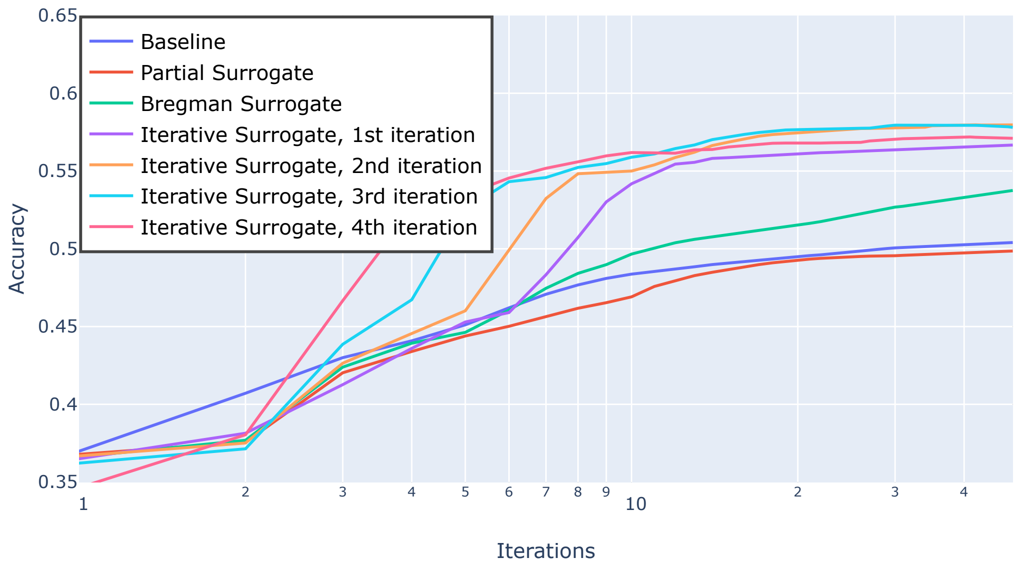


Figure 4.3: Training accuracy for the variational segmentation model discussed in subsection 4.4.2 for a linear model $\mathcal{N}(\theta, y_i)$. Directly training a cross-entropy loss without the perimeter term, training the Bregman surrogate (4.28), the Partial surrogate (4.16) and four iterations of the iterative scheme are compared. We find that the end-to-end training with the perimeter term increases the segmentation accuracy. We also see that a small number of iterations in the iterative scheme is sufficient for a practical CV task.

Model	PSNR	T	PSNR(Iter.)	TT
Total Variation	27.41	-	-	-
3 3x3 Filters	26.66	00:34	27.66	02:21
48 7x7 Filters	27.41	02:45	28.03	03:11
96 9x9 Filters	27.46	01:43	28.03	02:22

Table 4.1: Training time (T) in minutes for each surrogate computation and PSNR on the test dataset for various gray-scale filters for the energy model in (4.30) with and without the iterative process of (4.22) and total time (TT) for the iterative process are compared to total variation with optimal regularization parameter. Note that training time varies mostly due to differing iteration counts. The results of the convex model of [68] are reproduced.

4.4.3 Analysis Operator Models

Finally, we illustrate the behaviour of our approach on a practically relevant model, learning a set of optimal convolutional filters for denoising [245, 68]. We consider the parametric energy model

$$x(\theta) = \arg\min_x \frac{1}{2}||x - y_i||^2 + ||D(\theta)x||_1, \tag{4.29}$$

with $D(\theta)$ denoting the convolution operator to be learned, which is prototypical for many other image processing tasks. We consider square loss $l(x, y) = \frac{1}{2}||x - y||^2$ as a higher loss function and apply our approach. A Bregman surrogate for this model has the form

$$\min_\theta \sum_{i=1}^N \min_{||p_i||\leq 1} ||D(\theta)x_i^*||_1 + \frac{1}{2}||D^T(\theta)p_i - y_i||^2. \tag{4.30}$$

Model (4.29) was previously considered in [68, 66], where it was solved via implicit differentiation. We repeat the setup of [68] and train a denoising model on the BSDS dataset [188]. Refer to the appendix for the experimental setup and optimization strategy.

Table 4.1 shows both PSNR values achieved when training $D(\theta)$ as convolutional filters as well as training time. In comparison to [68], we find strikingly, that we can train a convex model with similar performance to the convex model in [68], while being an order of magnitude faster than the original approach. Furthermore in [68], the necessary training time jumps from 24 hours for 48 7x7 filters to 20 days for 96 9x9 filters - in our experiment the training time is almost unaffected by the number of parameters, and in this example actually smaller as the larger model converges faster. Also this analysis validates that the iterative process is crucial to reaching competitive PSNR values.

4.5 Conclusions

We investigated approximate training strategies for data-driven energy minimization methods by introducing *parametric majorizers*. We systematically studied such strategies in the framework of convex analysis, and proposed the Bregman distance induced by the lower level energy as well as over-approximations thereof as suitable majorizers. We discussed an iterative scheme that shows promise for applications in computer vision, particularly due to its scalability as shown by its application to image denoising.

This appendix is the original appendix of Geiping & Moeller [2019][105].

4.A Convex Analysis in Section 3

4.A.1 Details for Derivation of (4.11) to (4.12)

Equation (4.11) in the main paper describes the application of Bregman duality:

$$D^0_{E_\theta}(x_i^*, x_i(\theta)) = D^{x_i^*}_{E_\theta^*}(0, q_i) \quad q_i \in \partial E(x_i^*, y_i, \theta), \tag{4.11}$$

which is a common application of the following identity [44, 28]:

Lemma 4.10 (Bregman Identity). *Consider a convex lsc. function $E : \mathbb{R}^n \to \mathbb{R}$ with a subgradient $p \in \partial E(y)$. Then, the following identity holds:*

$$D^p_E(x, y) = D^x_{E^*}(p, q), \quad q \in \partial E(x)$$

Proof. This property follows from equality (Fenchel's identity) in the Fenchel-Young inequality $E(x) + E^*(p) = \langle p, x \rangle \iff p \in \partial E(x)$. To see this we write

$$D^p_E(x, y) = E(x) - \langle p, x \rangle - E(y) + \langle p, y \rangle$$

and apply Fenchel's identity for p, y to find

$$D^p_E(x, y) = E(x) - \langle p, x \rangle + E^*(p)$$

We then introduce any $q \in \partial E(x)$ by writing $\langle p, x \rangle = \langle p - q + q, x \rangle$ and apply Fenchel's identity again: ■

$$D^p_E(x, y) = E^*(p) - E^*(q) - \langle x, p - q \rangle = D^x_{E^*}(p, q)$$

The step from (4.11) to (4.12) is simply the first step of this derivation:

$$\begin{aligned} D_{E_\theta}(x_i^*, x_i(\theta)) &= E(x_i^*, y_i, \theta) - \langle 0, x_i^* \rangle + E^*(0, y_i, \theta) \\ = D^{x_i^*}_{E_\theta^*}(0, q_i) &= E(x_i^*, y_i, \theta) + E^*(0, y_i, \theta) \end{aligned} \tag{4.12}$$

as $p_i = 0$ is a subgradient of E at $x_i(\theta)$ and q_i at x_i^*.

4.A.2 Details for Derivation of (4.14) to (4.15)

A crucial subtlety of Lemma 4.10 is that this identity holds for any $q \in \partial E(x)$ and the choice of subgradients is irrelevant, the Bregman distance is equal for all choices. This motivates the introduction of the W-function $W_E(p, x) = E^*(p) + E(x) - \langle p, x \rangle$. This function is convex in either p or x and always non-negative. It can be understood as measuring the deviation of p from subgradients of x as a direct implementation of the Fenchel-Young inequality. As such it is 0 exactly if $p \in \partial E(x)$. Previous usage of this function can be found for example in [46, 237]. For Legendre functions [19], i.e. functions where both E and E^* are (essentially) smooth, the connection to Bregman distances is immediate:

$$W_E(p, x) = D^p_E(x, \nabla E^*(p)),$$

for non-smooth functions this is also a part of the proof of Lemma 4.10, replacing $\nabla E^*(p)$ by $y \in \partial E^*(p)$. As such, we can write (4.12) as

$$D^{x_i^*}_{E^*}(0, q_i) = W_{E_\theta}(0, x_i^*). \tag{4.12}$$

The introduction of this function then allows us to show that

$$W_E(0, x_i^*) = \min_z W_{E_1,\theta}(-z, x_i^*) + W_{E_2,\theta}(z, x_i^*) \tag{4.15}$$

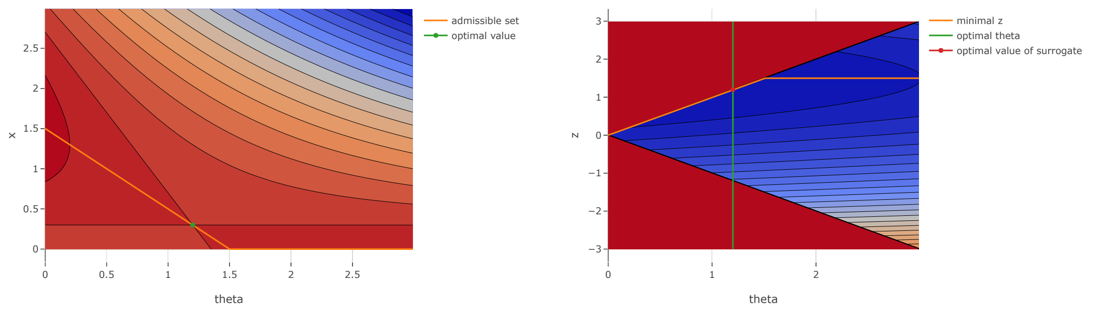


Figure 4.4: Visualization of the Bregman surrogate problem in primal formulation (left) and dual formulation (right). The problem in visualized over all (x, θ), respectively (z, θ). The admissible $x(\theta)$ are marked in orange in the left contour plot and the optimal $z(\theta)$ one the right. The optimal value in θ is marked in green in both plots.

under the assumption in (4.13), that E can be written as $E_1 + E_2$, with both functions convex. We recognize this as the clear extension of the infimal convolution property $E^*(0) = \min_z E_1^*(-z) + E_2^*(z)$ (which itself can be understood as Fenchel's duality theorem applied to E_1, E_2) to these functions, in the smooth setting this could be written via

$$D_{E^*}^{x_i^*}(0, \nabla E(x_i^*)) = \min_z \; D_{E_1^*}(-z, \nabla E_1(x_i^*)) \\ + D_{E_2^*}(z, \nabla E_2^*(x_i^*)).$$

We arrive at (4.15) from (4.14) by rewriting E in (4.14):

$$\begin{aligned} \text{(4.14)} \quad & \min_z E_1(x_i^*, y_i, \theta) + E_2(x_i^*, y_i, \theta) \\ & + E_1^*(-z, y_i, \theta) + E_2^*(z, y_i, \theta) \\ & = \min_z E_1(x_i^*, y_i, \theta) + E_2(x_i^*, y_i, \theta) + \langle z, x_i^* \rangle \\ & + E_1^*(-z, y_i, \theta) + E_2^*(z, y_i, \theta) - \langle z, x_i^* \rangle \\ \text{(4.15)} \quad & = \min_z W_{E_1,\theta}(-z, x_i^*) + W_{E_2,\theta}(z, x_i^*). \end{aligned}$$

4.A.3 Proof of Proposition 2

Proposition 4.2 (Ordering of parametric majorizers). *Assuming the condition $l(x,z) \leq D_{E_\theta}(x,z)$ from (4.8), we find that the presented parametric majorizers can be ordered in the following way:*

$$\begin{aligned} l(x_i^*, x(\theta)) &\leq D_{E_\theta}^0(x_i^*, x_i(\theta)) = D_{E_\theta^*}^{x_i^*}(0, q_i) \\ &\leq \min_{z \in \partial E_2(x_i^*)} W_{E_1}(-z, x_i^*) \\ &\leq \frac{1}{m(\theta, y)} ||q_i||^2 \qquad \text{s.t. } q_i \in \partial E(x_i^*, y, \theta). \end{aligned}$$

The Bregman surrogate majorizes the original loss function and is in turn majorized by the partial surrogate which is majorized by the gradient penalty under the assumption of $m(\theta, y)$-strong convexity of E_1.

Proof. The first inequality follows directly by the assumption $l(x,z) \leq D_{E_\theta}(x,z)$. The second inequality is the application of Bregman Duality discussed in Lemma Lemma 4.10. From (4.15) we now see that

$D^{x_i^*}_{E_\theta}(0, q_i)$, $q_i \in \partial E(x_i^*, y_i, \theta)$ can be written as a minimum over z. Clearly choosing a non-optimal z yields an upper bound to this minimal value. Without loss of generality, we choose $z \in \partial E_2(x_i^*)$ so that $W_{E_2,\theta}(z, x_i^*)$ is equal to zero.

Now we assume that E is $m(\theta, y)$-strongly convex. We subsume this strong convexity term in E_1 again without loss of generality so that E_1 is strongly convex. By convex duality [21], this implies that E_1^* is $m(\theta, y)$ strongly smooth, i.e. $D^x_{E_1^*}(p, q) \leq \frac{1}{2m(\theta,y)}||p-q||^2$. Following (4.12), we write

$$\begin{aligned} W_{E_1^*}(-z, x_i^*) = D^{x_i^*}_{E_1^*}(-z, r) \quad & z \in \partial E_2(x_i^*, y_i, \theta), \\ & r \in \partial E_1(x_i^*, y_i, \theta) \\ \leq \frac{1}{2m(\theta, y)}||-z-r||^2 & \\ = \frac{1}{2m(\theta, y)}||q_i||^2 \quad & q_i \in \partial E(x_i^*, y_i, \theta), \end{aligned}$$

under mild assumptions on the additivity of subgradients of E_1 and E_2. ■

4.A.4 Derivation of the surrogate functions for the example in 4.3.3

Subsection 4.3.3 discusses the non-smooth bi-level problem given in (4.18) and (4.19):

$$\min_{\theta \in \mathbb{R}} \frac{1}{2}|x^* - x(\theta)|^2, \tag{4.18}$$

$$\text{subject to} \quad x(\theta) = \arg\min_x \frac{1}{2}|x-y|^2 + \theta|x|. \tag{4.19}$$

for both $x^*, y \in \mathbb{R}$. In this setting, the 'primal' formulation of the Bregman surrogate is given by

$$\min_\theta \max_x \frac{1}{2}|x^* - y|^2 - \frac{1}{2}|x-y|^2 + \theta\left(|x^*| - |x|\right) \tag{4.10 ex.}$$

whereas the 'dual' formulation is given by

$$\min_\theta \min_{|z| \leq \theta} \frac{1}{2}|x^* - y|^2 + \theta|x^*| + \frac{1}{2}|z-y|^2. \tag{4.12 ex.}$$

Note that this problem is convex in z, θ as the epigraph constraint $|z| \leq \theta$ is convex. Both (equivalent!) variants are visualized in Figure 4.4. We see that the saddle-point of the primal formulation and the minimizer of the dual formulation correctly coincide with the optimal θ.

Moving forward, we set $E_1(x, y) = \frac{1}{2}|x-y|^2$ and $E_2(x, \theta) = \theta|x|$ to compute the two partial surrogates. Firstly $W_{E_1,\theta}(-z, x^*), z \in \partial E_2(x^*)$ leads to

$$\min_\theta \frac{1}{2}|x^* - y + q|^2, \quad q \in \partial|x^*|, \tag{4.16 ex.1}$$

where we take $q = \operatorname{sign}(x^*)$ as $x^* \neq 0$ in our example. As E_1 is a quadratic function, this is also equivalent to the gradient penalty in (4.17). The second partial surrogate, $W_{E_2,\theta}(z, x^*), z \in \partial E_1(x^*)$ can be written as

$$\begin{aligned} & \min_\theta \theta|x^*| + I_{|\cdot| \leq \theta}(x^* - y) - \langle x^*, x^* - y\rangle \\ = & \min_{|x^* - y| \leq \theta} \theta|x^*| + C. \end{aligned} \tag{4.16 ex.2}$$

Figure 4.4 here and Figure 1 in the main paper both arise from the data point $x^* = 0.3, y = 1.5$.

To give some more details on the fact that the Bregman surrogate is exactly identical with the original loss function in the vicinity of the optimal value, note that this is caused by the special structure of the Bregman distance of the absolute value, $D_{|\cdot|}(x, y)$ as $D_{E_\theta}(x, y)$ decomposes into $\frac{1}{2}|x-y|^2 + \theta D_{|\cdot|}(x, y)$. This function is equal to the higher-level loss function as soon as the signs of x^* and $x(\theta)$ coincide and as such the majorizer is exact, even if it is much easier to compute.

4.A.5 Proof of Proposition 4

Subsection 4.3.4 describes an iterative procedure for repeated application of the majorization strategies discussed in subsection 4.3.2 This scheme was based on the result of Proposition 4.5:

$$l(x,y) \leq l(x,z) + \langle \nabla_z l(x,z), y-z\rangle + D_E(z,y), \tag{4.20}$$

inserting $x = x_i^*, y = x_i(\theta), z = x_i(\theta^k)$ leads to

$$\begin{aligned} l(x_i^*, x_i(\theta)) \leq l(x_i^*, x_i(\theta^k)) + D_{E_\theta}(x_i(\theta^k), x_i(\theta)) \\ + \langle \nabla l(x_i^*, x_i(\theta^k)), x_i(\theta) - x_i(\theta^k)\rangle. \end{aligned} \tag{4.20b}$$

Equation (4.20), respectively (4.20b), lead to a monotone descent of the higher-level loss, as shown in Proposition 4:

Proposition 4.4 (Descent Lemma). *The iterative procedure given by*

$$\begin{aligned} \theta^{k+1} = \arg\min_\theta \sum_{i=1}^N & l(x_i^*, x_i(\theta^k)) \\ & + \langle \nabla l(x_i^*, x_i(\theta^k)), x_i(\theta) - x_i(\theta^k)\rangle \\ & + D^0_{E_\theta}(x_i(\theta^k), x_i(\theta)) \end{aligned}$$

is guaranteed to be stable, i.e. not to increase the bi-level loss:

$$\sum_{i=1}^N l\left(x_i^*, x_i(\theta^{k+1})\right) \leq \sum_{i=1}^N l\left(x_i^*, x_i(\theta^k)\right) \tag{4.23}$$

Proof of Proposition 4. θ^{k+1} is a minimizer of the iterative scheme. Therefore, evaluating the iteration at θ^{k+1} leads to a lower value than evaluating at θ^k:

$$\begin{aligned} & \sum_{i=1}^N l(x_i^*, x_i(\theta^k)) + \langle \nabla l(x_i^*, x_i(\theta^k)), x_i(\theta^{k+1}) - x_i(\theta^k)\rangle \\ & \quad + D^0_{E_{\theta^{k+1}}}(x_i(\theta^k), x_i(\theta^{k+1})) \\ \leq & \sum_{i=1}^N l(x_i^*, x_i(\theta^k)) + \langle \nabla l(x_i^*, x_i(\theta^k)), x_i(\theta^k) - x_i(\theta^k)\rangle \\ & \quad + D^0_{E_{\theta^k}}(x_i(\theta^k), x_i(\theta^k)) \\ = & \sum_{i=1}^N l(x_i^*, x_i(\theta^k)) \end{aligned}$$

Now the left-hand-side is also equivalent to (4.20b) evaluated at θ^{k+1}. Applying the inequality in (4.20b) for all $i = 1, \dots, N$ we find

$$\sum_{i=1}^N l(x_i^*, x_i(\theta^{k+1})) \leq \sum_{i=1}^N l(x_i^*, x_i(\theta^k)).$$

■

Remark 4.5. The iterative scheme given in (4.23), i.e.

$$\begin{aligned} \theta^{k+1} = \arg\min_\theta \sum_{i=1}^N & E^*\left(\nabla l(x_i^*, x_i(\theta^k)), y_i, \theta\right) \\ & + E\left(x(\theta^k), y_i, \theta\right). \end{aligned} \tag{4.22}$$

is an over-approximation of the iterative scheme discussed in Proposition 4. As such we expect the results of Proposition 4 to hold only approximately as stated in the main paper.

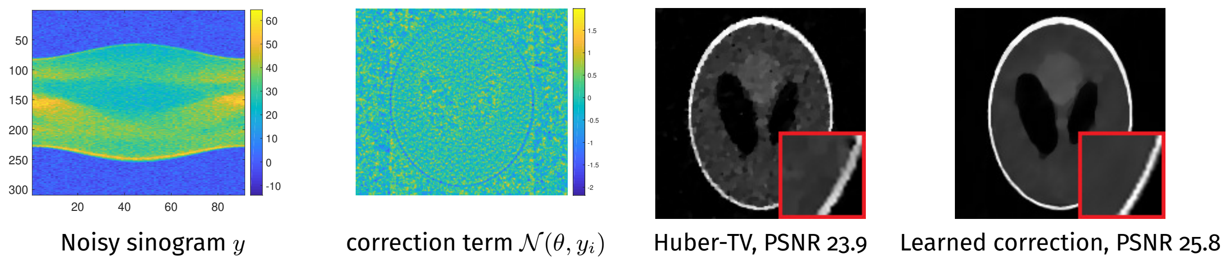


Figure 4.5: Illustrate our results for learning a linear correction term for a Huber-reguralized CT reconstruction problem. In reference to Figure 2 in the main paper we also visualize input data and the learned linear correction map. The predicted linear correction term can be visualized and inspected, and its influence can easily be quantified or explicitly scaled via a parameter.

4.B Experimental Setup

This section will add additional details to the experiments presented in the paper[2].

4.B.1 CT - Additional Details

The implementation of the CT example in subsection 4.4.1 is straightforward. We generate pairs (y_i^*, x_i^*) of noisy sinograms and ground truth images and optimize

$$\min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{n} \|A^* A x_i^* - A^* y_i + \beta \nabla R(x_i^*) + \mathcal{N}(\theta, y_i)\|_2^2.$$

We test our model on the widely-used Shepp-Logan phantom, comparing the learned model with a pure Huber-TV solution, for which we found the optimal parameter β by grid search. This setup was implemented in Matlab. To visualize the linear correction term, we repeat an extended version of Figure 4.2.

4.B.2 Segmentation - Additional Details

The segmentation experiment shown in Figure 4.3 of the main paper shows the results of training the variational model in (4.24), which corresponds to an augmented cross-entropy term, as discussed in subsection 4.4.2.

The partial surrogate implemented in Figure 4.3 is a direct application of (4.16) to the segmentation setting, giving

$$\min_{\theta} \sum_{i=1}^{N} \min_{p_i \in \partial ||D x_i^*||} D_h \left(x_i^*, \nabla h^* \left(\mathcal{N}(\theta, y_i) - D^T p_i \right) \right),$$

where the computation of the auxiliary variable p_i is simplified. Note further that the gradient penalty cannot be applied in this setting, as the segmentation energy E is not strongly convex. Similarly, the iterative approach can be computed to be

$$\min_{\theta} \sum_{i=1}^{N} \min_{||p_i|| \leq 1} h^* \left(\frac{x_i^*}{x_i(\theta^k)} + \mathcal{N}(\theta, y_i) - D^T p_i \right) \\ - \langle \mathcal{N}(\theta, y_i), x_i(\theta^k) \rangle$$

which is still convex in $\mathcal{N}(\theta, y)$, but the input arguments now take previous solutions into account.

To emphasize the convexity of the setup, we choose $\mathcal{N}(\theta, y_i)$ as a linear convolutional network of $3x3x3$ filters for each target class. We accordingly optimize the resulting convex minimization problems by an

[2]Refer also to the implementations hosted on https://github.com/JonasGeiping/ParametricMajorization

optimal convex optimization method, namely FISTA [25]. To solve the inference problem in Eq. (25) we apply usual strategies and optimize via a primal-dual algorithm [54] - to increase the speed we adapt a recent variant [55] and consider the Bregman-Proximal operator in the primal sub-problem for which we use the entropy function h described in the paper, paralleling [24, 216].

We draw four images and their corresponding segmentations from the `cityscapes` data set [75] and implement the proposed procedures in PyTorch [219]. For Figure 3 we drew the first four images, which we resized to 128x256 pixels. To visualize the improvement over the iterations, we initialize the subsequent iterations of the iterative scheme again with the initial value of θ, so that the training accuracy curves in Figure 3 are comparable. This is of course not strictly necessary and θ could be initialized with the current estimate in every iteration. We also point out that we visualize the actual training accuracy in Figure 3, meaning the percentage of successfully segmented pixels after *hard argmax* of the results of the algorithms.

4.B.3 Analysis Operators - Additional Details

For this experiment we considered the task of learning an 'analysis operator' $D(\theta)$, i.e. a set of convolutional filters θ^k so that $D(\theta) = \sum_{k=1}^K \theta_k * x$ for a set of K filters. Due to anisotropy, we can write the resulting minimization problem as

$$x(\theta) = \arg\min_x \frac{1}{2}||x-y||^2 + \sum_{k=1}^K ||\theta_k * x||_1.$$

We repeat the experimental setup of [68] and train this model on image pairs x^*, y of noise-free and noisy image patches, to learn filters that result in a convex denoising model [67, 68]. To do so we draw a batch of 200 $64x64$ image patches from the training set of the Berkeley Segmentation data set [188], convert the images to gray-scale and add Gaussian noise. To compare with [68] and [308] we do not clip the noisy images and use Matlab's `rgb2gray` routine to generate this data. Further, as in [68], we do not optimize directly for the convolutional filters, but instead decompose each filter into a DCT-II basis, where we learn the weight of each basis function, excluding the constant basis function [125]. Before training we initialize these weights by orthogonal initialization [252] with a factor of 0.01, respectively 0.001 for the larger 9x9 filters.

To solve the training problem we minimize (4.30) jointly in $\theta, \{p_i\}_{i=1}^N$. We do this efficiently by taking steps toward the optimal weights with the 'Adam' optimization procedure [145] with a step size $\tau = 0.1$ (although gradient descent with momentum or FISTA [25] are also valid options). We use a standard accelerated primal-dual algorithm [54] to solve the convex inference problem. For the iterative procedure we repeat this process, computing $x(\theta^k)$ after every minimization of (4.30), inserting it as a factor into E^* and repeating the optimization. If the iterative procedure increases the loss value, we reduce the step size τ of the majorizing problem and repeat the step. If reducing the step size does not successfully improve the result for several iterations, we terminate the algorithm.

We implement this setup in PyTorch [219] and refer to our reference implementation for further details.

For total variation denoising, which corresponds to choosing $D(\theta)$ as the gradient operator with appropriate scaling, $\alpha\nabla$, we use grid search to find the optimal scaling parameter α.

We report execution times for a single minimization of (4.30) for different filter sizes in Table 4.1 as well as total time for an iterative procedure. These timings are reported for a single *GeForce RTX 2080Ti* graphics card.

The following sections contain unpublished additional material.

4.C Extended Overview of Related Work

Bilevel optimization is an incredibly general formalism. This overview concentrates mainly on optimization algorithms relevant to the task of training parametrized energy models. We are especially interested in methods that scale to a large number of trainable parameters. For a more general introduction to bilevel optimization, we refer to [82, 84].

In essence, we are faced with a problem of hyperparameter optimization, when considering the bilevel optimization of energy parameters. If the dimensionality of the parameter space is small, then grid search or Bayesian methods [195, 264, 29] can be applied. Linear bilevel problems allow us to guarantee the solution to be an extremal value of the upper-level polyhedron. As such 'vertex enumeration' techniques can find the optimal solution in finite time [50, 30]. However as complexity increases, branch-and-bound strategies [15] or evolutionary algorithms [262] are necessary. None of these approaches scale well to larger dimensions.

Scalable approaches to bilevel optimization usually consider *direct descent* methods [153, 251, 74]. The gradient of the loss function with respect to the parameters θ is computed and a descent step in this direction is taken. For this approach it is necessary that the considered energy is sufficiently smooth, as the computation of the gradient in respect to the $\arg\min$ requires implicit differentiation. As an example of such a strategy, consider the strategy considered in [67, 66]: For the bilevel problem of (4.2) and (4.3), rewrite the problem to a problem with equality constraints:

$$\min_{\theta \in \mathbb{R}^p} R(\theta) + \sum_{i=1}^N l(x_i^*, x_i) \tag{4.31}$$

$$\text{s.t. } 0 = \nabla_x E(x_i, y_i, \theta) \quad \forall i = 1, \dots, N. \tag{4.32}$$

This can be rewritten in terms of a Lagrangian

$$\mathcal{L}(x, \theta, \lambda) = R(\theta) + \sum_{i=1}^N l(x_i^*, x_i) + \langle \nabla_x E(x_i, y_i, \theta), \lambda \rangle, \tag{4.33}$$

so that an iterative descent algorithm takes the form

$$\begin{cases} x^{k+1} &= \arg\min_x E(x, y, \theta^k) \\ \lambda^{k+1} &= \left[\nabla_x^2 E(x^{k+1}, y, \theta^k)\right]^{-1} (-\nabla l(x^*, x^{k+1})) \\ \theta^{k+1} &= \text{DescentStep}\left(\theta^k, \nabla_\theta \langle \nabla_x E(x^{k+1}, y, \theta^k), \lambda^{k+1} \rangle\right) \end{cases} \tag{4.34}$$

This algorithm requires the evaluation and inversion of the Hessian of E which comprises a large computational effort. The inverse matrix can usually not be stored in memory for practical parametrized energies, so that the inverse has to be computed iteratively from evaluations of the Hessian-vector product. Higher-order methods, such as a Newton method as considered in [157] require derivatives of even higher order.

Furthermore the complexity increases if non-smooth energies E are considered. The problem of directly minimizing θ without assuming that E is smooth leads to optimization problems with equilibrium constraints (MPECs), see [26] for a discussion in terms of machine learning or in more generality [83, 82, 84] and [74]. To remedy this issue, elaborate smoothing techniques are usually employed when optimizing nonsmooth parametrized energies. This approach has been successfully applied to many problems of parameter optimization in image processing, as attested for example in [48, 68, 66, 67, 79, 80, 110, 119, 120].

A very different, but highly practical strategy is to employ 'unrolling'. The bilevel problem is transformed into a single level problem by choosing an iterative optimization algorithm $\mathcal{A}$, that produces an approximate solution to the lower-level problem after a fixed number of iterations. $x(\theta)$ is then replaced by the approximation $\mathcal{A}(y,\theta)$. Automatic differentiation [111] allows for an efficient evaluation of the gradient of the upper-level loss w.r.t to this reduced objective

$$\min_\theta R(\theta) + \sum_{i=1}^n l(x_i^*, \mathcal{A}(y_i,\theta)). \tag{4.35}$$

Unrolling is a prominent strategy across fields, e.g. MRF literature [14, 189] in deep learning [312, 64, 58, 173] and in variational settings [215, 164, 163, 113, 114, 238]. This technique can also be applied to nonsmooth lower-level energies, if the optimization algorithm $\mathcal{A}$ is chosen carefully not to contain non-differentiable operations [215, 216, 163]. Returning to the discussion of implicit and explicit models in the introduction of this chapter, unrolling often works well *because* it combines the parametrized energy and its optimization algorithm into a single explicit process. However it is usually unclear how easy it is to separate the optimized parameters from the chosen algorithm and the chosen number of steps, i.e. formalizing to what extent this method learns an optimal energy E for arbitrary optimization algorithms and precisions.

Later works [69, 65, 113, 114] further develop this strategy to allow the lower-level parameters to change in between the fixed number of iterations, leading to residual convolutional neural network [117] type structures that model differential equations, but cannot be cast as energy formulations. As pointed out in [146], these strategies are more aptly considered as a set of nested quadratic lower-level problems.

A set of very different tools is applied in structured prediction, although the parameter learning task is similar. Structured prediction is a generalization of linear classification methods, in particular support vector machines [292, 76]. The goal [209, 287, 281, 78] is to find an optimal discriminant function, that is generally assumed to be linearly dependent on a vector of parameters $\theta \in \mathbb{R}^p$ and a feature representing function $\psi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$, so that the discriminant function is given by

$$f(y,\theta) = \arg\max_{x\in X} \langle \theta, \psi(x,y)\rangle. \tag{4.36}$$

Usually the output x is drawn from a discrete and structured output space X (hence the name). Structured prediction is generally concerned with output spaces which are discrete, but where the maximization in (4.36) is still feasibly solvable, e.g. conditional random fields [274], context-free grammar [283] and others [235, 282, 280]. Given a set of training samples the parameter optimization problem is also a bilevel problem - finding the optimal parameters θ, so that the regularized empirical risk $R(\theta)+\sum_{i=1}^N l(x_i^*, f(y,\theta))$ is small. Yet, this objective is not directly optimized, instead an upper bound is constructed that can be feasibly minimized (Note that the structured prediction problem is nonsmooth, due to the complicated structure of the constraint set X). If the training data is *separable*[3] and a classification loss is considered, then the training problem is

$$\min_{\theta\in\mathbb{R}^p} R(\theta) \quad \text{s.t. } \langle \theta, \psi(x_i^*,y) - \psi(x_i,y)\rangle \geq 1 \quad \forall x_i \in X, \tag{4.37}$$

whereas for an arbitrary loss function and arbitrary data [287, 281], the training problem is given by

$$\min_{\theta\in\mathbb{R}^p, \xi\in\mathbb{R}^N_+} R(\theta) + \frac{C}{N}\sum_{i=1}^N \xi_i \tag{4.38}$$

$$\text{s.t. } \langle \theta, \psi(x_i^*,y) - \psi(x_i,y)\rangle \geq l(x_i^*,x_i) - \xi_i \quad \forall x_i \in X. \tag{4.39}$$

The difference in comparison to the previously discussed methods is striking, yet structured prediction methods are successfully used in training a wealth of computer vision applications [276, 95, 310] even beyond for more complicated prediction strategies, e.g. [147, 73], where $f(y,\theta) = \arg\min_{x\in\Delta}\langle n(\theta,y),x\rangle + TV(x)$ is given by a neural network $n(\theta,y)$ and combined with total variation regularization during prediction.

[3]We will return to the concept of separability in the next section in greater detail.

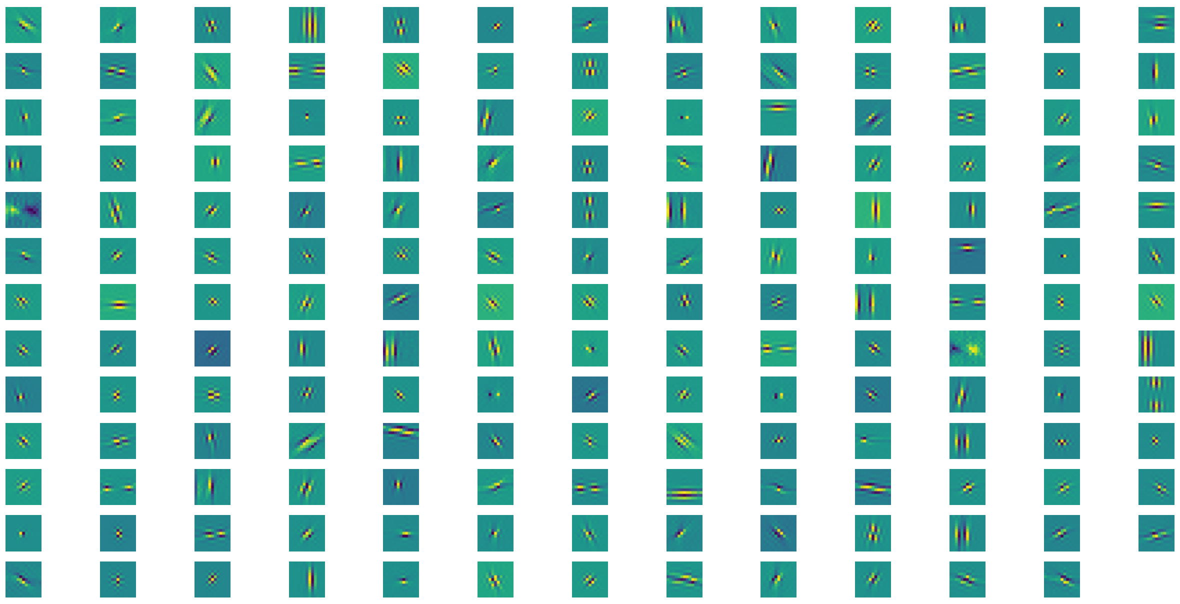


Figure 4.6: 168 Convolutional kernels of size 13x13 trained for the denoising task.

The underlying principle in (4.36) is *perceptron* loss [244]. In a more general setting known as structured perceptron loss [72, 175], generalized [168, 167] or multiclass perceptron loss . This strategy is applied to find parameters of a parametrized energy by optimizing

$$\min_{\theta} \sum_{i=1}^{N} \max_{x} E(x_i^*, y_i, \theta) - E(x, y_i, \theta), \tag{4.40}$$

although usually not written down in this form, but as a coordinate or stochastic gradient descent over all samples, where in each step, the approximate solution $\min_x E(x, y_i, \theta^k)$ is computed and then a gradient step in θ is taken. However this approach, while simple, does not take a loss function or regularization into account [209]. [168] further remarks that a practical problem with (4.40), especially for parametrized energies with high degrees of freedom is the phenomenon of *energy collapse*, where the optimal solution of (4.40) is given by an energy that assigns equal values to all $x \in \mathbb{R}^n$.

Finally, we also mention a quite different view onto bilevel optimization. Essentially, bilevel optimization can also be understood as a nonconvex composite optimization problem [171, 88]. Considering the decomposition of [104], we can solve a composite optimization problem $G(\rho(\theta)) + R(\theta)$ as a majorization-minimization scheme, iterating

$$\theta^{k+1} \in \arg\min_{\theta \in \mathbb{R}^p} \langle \nabla G(\rho(\theta^k)), \rho(\theta) - \rho(\theta^k) \rangle + R(\theta) + D_h(\rho(\theta), \rho(\theta^k)). \tag{4.41}$$

This is our bilevel optimization problem, only the function $\rho : \mathbb{R}^p \to \mathbb{R}^n$ is highly complicated and non-smooth.

4.D Analysis Operator Learning - Additional Figures

Additional visualization of the learned filters can be found in Figure 4.6 and Figure 4.7. An example of an image denoised by these filters is visualized in Figure 4.8.

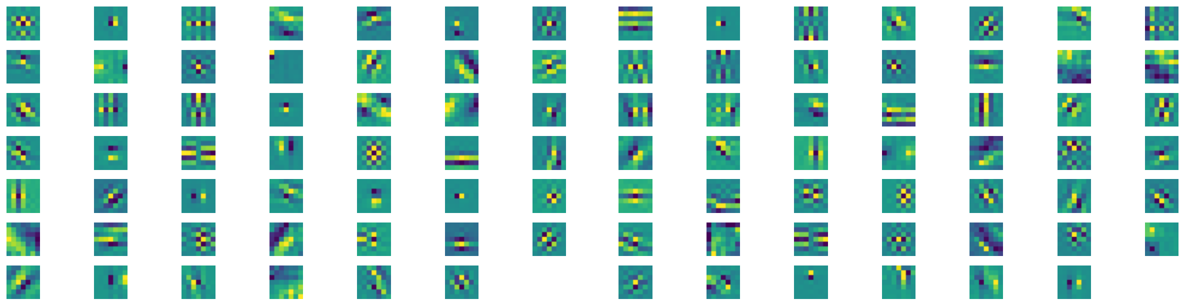


Figure 4.7: Filter for single-Level Bilevel Surrogate on the left, Filters after iterative bilevel surrogate to the right.

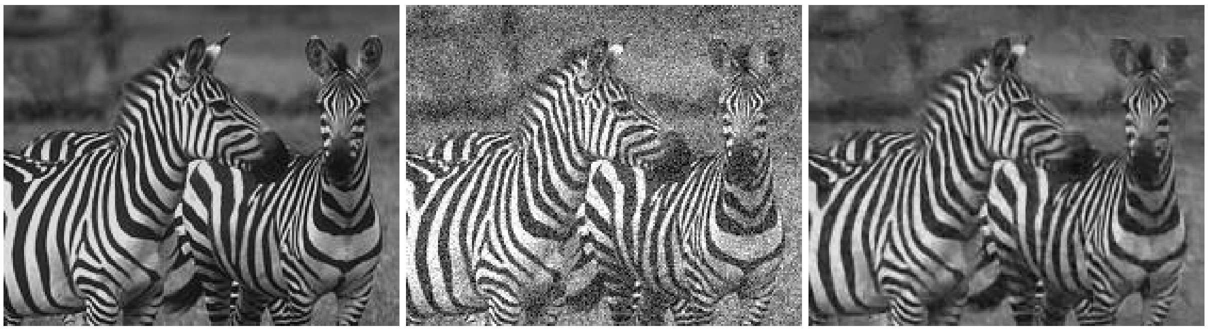


Figure 4.8: Denoising by a learned anaylsis operator. From left to right: ground truth, noisy data and denoised image. Image taken from the BSDS test set[188]

4.E Derivation from Support Vector Machine Principles

This appendix will focus on introducing the concept of parametric majorization derived in the main work for the training of nonsmooth parametric energies from first principles from a perspective of appropriate generalizations of training methods and concepts for support vector machines [292], and their extensions to structured prediction [287, 281] even to our more general bilevel problems. This will lead to a substantially different viewpoint onto the discussed parametric optimization, but recover the same functionality.

We will first focus on clarifying additional definitions. The bilevel problem posed in (4.2) and (4.3) is only well-defined if the minimizer of E is unique. This requirement is not to be understated. Consider the simplest example of finding the optimal $\theta \in \mathbb{R}$, so that minimizing the energy $E(x, y, \theta) = (\theta x - y)^2$ recovers the optimal scale factor between $x \in \mathbb{R}$ and $y \in \mathbb{R}$. The minimizer is unique for almost all θ, but in the case of $\theta = 0$, where any $x \in \mathbb{R}$ is a solution to the energy minimization problem. Classical bilevel literature distinguishes *optimistic* and *pessimistic* approaches [84] to the non-uniqueness of the minimization problem. In the first case, returning the best solution among the set solutions to the lower-level energy, as measured by the value of the higher-level loss and the second returning the worst. For our purposes, the optimistic strategy is problematic, as we do not which solution to pick in the unsupervised case, i.e. when minimizing the energy model after training with new data.

To remedy this problem, we set the following assumptions.

Assumptions:

- The energy E is convex w.r.t to its first argument for all $y \in \mathbb{R}^m$ and $\theta \in \mathbb{R}^p$ and continuous w.r.t to its second and third argument, $\operatorname{dom} E$ is nonempty.

- The energy E has a unique minimizer w.r.t to its first argument for all $y \in \mathbb{R}^m$ and $\theta \in \mathbb{R}^p$.
- The loss function l is continuous in both arguments and fulfills $l(a,b) \geq 0,\ l(a,a) = 0\ \forall a,b \in \mathbb{R}^n$.

It is crucial for the stability of any training procedure, that the energy is well-posed for any data y and parameters θ, although we could of course in practice reduce these assumptions to necessarily hold only for data y from the expected data distribution and for the optimal parameters θ^* and the sequence $\{\theta^k\}_{i=1}^K$ of parameters evaluated during training.

4.E.1 The Separable Case

We can draw a parallel to the literature on structured prediction and generalize the notion of *separability*. A structured model is separable if parameters θ exist, so that the empirical risk (the loss over all training samples) is zero [287].

Definition 4.6 (Separable). *We define the problem of training an energy model E to be separable on a training set $\{(x_i^*, y_i)\}_{i=1}^N$ if optimal parameters θ^* exist, so that the higher-level loss l is zero:*

$$0 = \min_\theta \sum_{i=1}^N l(x_i^*, x_i(\theta)). \tag{4.42}$$

Clearly this property is both related to the 'expressiveness' of possible mappings $x_i(\theta) = \arg\min_x E(x, y_i, \theta)$ and to the simplicity of the training set. Returning to the example of TV denoising, the problem is separable, if all training example x_i^* are generated via TV denoising of unknown input images y with an unknown (but fixed) parameter $\hat{\theta}$. Conversely, if we consider arbitrary pairs (x_i^*, y_i) of natural images and their noisy counterparts, then the training problem is likely not to be separable. Yet even for an arbitrary vectors (x^*, y) , the energy $E(x,y,\theta) = \frac{1}{2}||x-\theta||^2$ is separable, as long as the pair (x^*, y) is the full training set (Learning $x(\theta) = \theta = x^*$).

We now make use of the additional assumption that $l(a,b) = 0 \implies a = b$. Under this assumption, separability implies a statement in terms of the energy as the set of constraints given by

$$E(x, y_i, \theta) \geq E(x_i^*, y_i, \theta) \qquad \forall x \in \mathbb{R}^n\ \forall i = 1, \dots, N, \tag{4.43}$$

paralleling the usual SVM definition, i.e [287]. While the set of constraints given above is feasible in some structured prediction tasks, where x is drawn from a discrete set, or can be approximated through cutting planes schemes [138], this is not the case for the continuous setting, that we consider. However, due to convexity of E, we are always able to efficiently reduce the constraint set by minimizing over x, so that we are left with N constraints

$$\min_{x \in \mathbb{R}^n} E(x, y_i, \theta) = E(x_i^*, y_i, \theta) \qquad \forall x \in \mathbb{R}^n\ \forall i = 1, \dots, N. \tag{4.44}$$

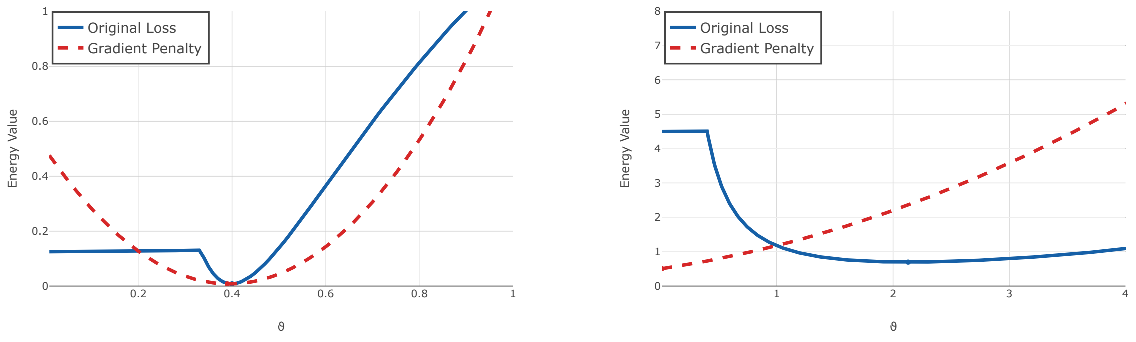
We could consider this as a minimization problem and find

$$\min_\theta R(\theta) \quad \text{s.t.}\ \min_{x \in \mathbb{R}^n} E(x, y_i, \theta) = E(x_i^*, y_i, \theta) \tag{4.45}$$

$$\forall x \in \mathbb{R}^n\ \forall i = 1, \dots, N. \tag{4.46}$$

However as we assume E to have a unique minimizer, the constraints are equally stated as $x_i^* = x_i(\theta)$ and we can use the convexity of E to reduce the bilevel problem to the single level problem without auxiliary minimizations,

$$\min_\theta R(\theta) \quad \text{s.t.}\ 0 \in \partial E(x_i^*, y_i, \theta), \tag{4.47}$$



a) Separable Data **b)** Nonseparable Data

Figure 4.9: Visualization of Example 4.8. The true bilevel loss (in blue) and the gradient penalty of (4.48) (in red), visualized for separable data $(x^*, y) = (0.5, 3)$ and nonseparable data $(x^*, y) = (3.0, 2.5)$. Note that the minimizers coincide in the separable case.

that can be solved as a standard (possibly non-convex) constrained optimization problem. If $R(\theta) = 0$ and E is differentiable, then we can even formulate an unconstrained minimization problem

$$\min_\theta \sum_{i=1}^n ||\nabla E(x_i^*, y_i, \theta)||^2, \tag{4.48}$$

which we will denote as *gradient penalty* from now on. In some sense, this will be our prototypical example of a surrogate function for the full bilevel problem.

Remark 4.7 (Separability and value optimization). The separability condition is also closely related to value constrained bilevel reformulations [247, 266], as it allows to rewrite the bilevel problem to

$$\min_{\theta \in \mathbb{R}^p} R(\theta) + \sum_{i=1}^N \min_{x \in \mathbb{R}^n} l(x_i^*, x)$$
$$\text{s.t. } E(x, y, \theta) \leq E(x_i^*, y_i, \theta).$$

This is only an equivalent reformulation for separable data.

Example 4.8. *As an example, consider the parametrized energy*

$$E(x, y, \theta) = \frac{\theta}{2}|x - y|^2 + |x|, \tag{4.49}$$

with the analytical solution $x(\theta) = \max(y - \frac{1}{\theta}, 0)$ *and the assorted bilevel training problem*

$$\min_{\theta \geq \varepsilon} \frac{1}{2}|x - x(\theta)|^2 + \frac{\gamma}{2}||\theta||^2 \quad \text{s.t. } x(\theta) = \arg\min_x E(x, y, \theta).$$

For scalar values, we can visualize the higher-level objective in Figure Figure 4.9. This example, also discussed in [157] is especially interesting in light of our discussion about separability. If we only consider a single data point (x^*, y) *as training datum, then the problem is separable if* $0 \leq x^* \leq y + \frac{1}{\varepsilon}$ *and non-separable else. We find that the single level surrogate of (4.48) can sucessfully minimize the bilevel problem in the separable case of Figure 4.9a, but not in the nonseparable case of Figure 4.9b.*

The case detailed in Example 4.8 is a generalization of the fact that a perceptron algorithm is sufficient to learn linearly separable data [209].

4.E.2 The Notion of Margin

Crucially, a support vector machine constructs a maximum margin hyperplane, increasing the stability and robustness of the classification. It is worthwhile to explore generalizations of this concept to arbitrary parametric energies. A classical SVM for binary classification (using the $0-1$ loss function $l(x,y) = \mathbf{1}_{x=y}(x,y)$) constructs a margin variant of (4.43) by enforcing the constraints

$$E(x, y_i, \theta) \geq E(x_i^*, y_i, \theta) + m, \tag{4.50}$$

$\forall x_i^* \neq x \in \mathbb{R}^n\ \forall i = 1, \ldots, N$, and maximizing the margin $m \geq 0$. This concept of a fixed margin can only be considered for energies whose minimizers can be drawn from a discrete set.

Remark 4.9. For parametric energies that are linear in θ (as discussed in both structured prediction and support vector machines), maximizing a margin m and regularizer $R(\theta) = I_{||\theta||\leq 1}(\theta)$ is equivalent to the usual SVM margin formulation

$$\min_\theta \frac{1}{2}||\theta||^2 \quad \text{s.t. } E(x, y_i, \theta) \geq E(x_i^*, y_i, \theta) + 1. \tag{4.51}$$

However this reformulation cannot be applied for arbitrary parametric energies.

For continuous energies, we need to consider a generalization of the *margin rescaling* of large margin methods for structured prediction [287, 280]. We construct a set of constraints by

$$E(x, y_i, \theta) \geq E(x_i^*, y_i, \theta) + m l(x_i^*, x), \tag{4.52}$$

$\forall x \in \mathbb{R}^n\ \forall i = 1, \ldots, N$. Without the linearity of E however as in [282, 287] the problem of finding parameters θ with maximal margin,

$$\max_{\theta, m} m \quad \text{s.t. } E(x, y_i, \theta) \geq E(x_i^*, y_i, \theta) + m l(x_i^*, x) \tag{4.53}$$

$\forall x \in \mathbb{R}^n\ \forall i = 1, \ldots,$ is difficult to solve.

Nevertheless, we can again use the convexity of E to provide structural information analytically. We do this by equating the notion of margin with (relative) strong convexity. We define a margin for continuous, convex parametric energies.

Definition 4.10. *A convex parametric energy $E : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ to has a margin of $m(\theta)$, relative to a loss function $l : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ if*

$$D_{E_\theta}(x, \hat{x}) \geq m(\theta) l(x, \hat{x}) \qquad \forall x, \hat{x} \in \operatorname{dom} E. \tag{4.54}$$

Due to the linearity of the Bregman distance, we can thus define a parametrized energy with normalized margin. If a parametric energy E fulfills the margin condition, $\frac{1}{m(\theta)}E$ is guaranteed to have a margin greater or equal to 1 for all $\theta \in \mathbb{R}^p$ satisfying $m(\theta) \neq 0$.

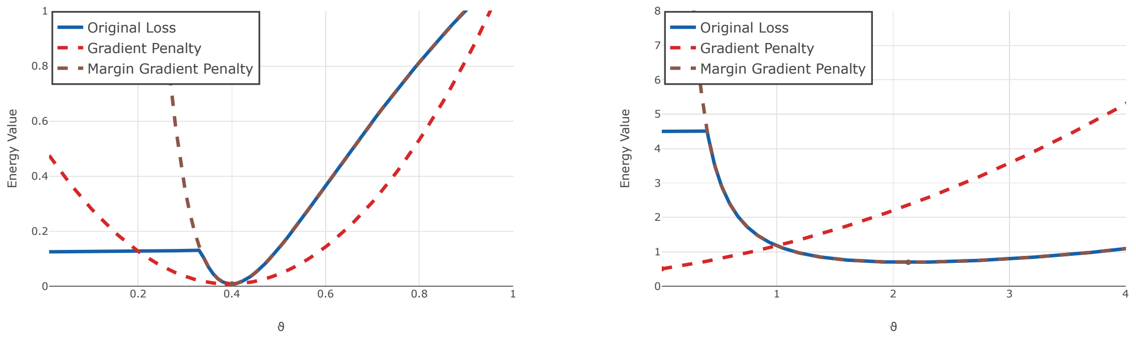
Instead of including (4.54) as a constraint during optimization, we thus compute $m(\theta)$ beforehand analytically. This constraint is equivalent to (4.52), as $0 \in \partial E(x_i^*, y_i, \theta)$ and we thus recover it as a special case of (4.54), inserting $\hat{x} = x_i^*$ for all $i = 1, \ldots, N$ samples. This definition is furthermore congruous with the definition of [168, 167] for continuous energies, where a continuous margin is defined by requiring that $E(x_i^*, y_i, \theta) + m \leq \min_{||\hat{x}-x_i^*||\geq \varepsilon} E(\hat{x}, y, \theta)$, which is a special case of (4.54), taking $l(x,y) = ||x-y||^2$.

Example 4.11 (Margin examples).

1. *Hyperparameter optimization for a convex regularizer $R : \mathbb{R}^n \to \mathbb{R}$ leads to the parametric energy*

$$E(x, y, \theta) = \frac{\theta}{2}||x-y||^2 + R(x). \tag{4.55}$$

   This energy has a margin of $m(\theta) = \theta$ w.r.t to $l(x, \hat{x}) = \frac{1}{2}||x-\hat{x}||^2$. The renormalized energy $E(x, y, \theta) = \frac{1}{2}||x-y||^2 + \frac{1}{\theta}R(x)$ has a margin of 1.



a) Separable Data **b)** Nonseparable Data

Figure 4.10: Visualization of Example 4.12. The true bilevel loss (in blue), the gradient penalty of (4.48) (in red), and the gradient penalty with normalized margin (4.58) (in brown) visualized for separable data $(x^*, y) = (0.5, 3)$ and nonseparable data $(x^*, y) = (3.0, 2.5)$. Note that all minimizers coincide in the separable case, but in the non-separable case, the normalized margin is necessary.

2. *Binary classification via the parametric energy*

(4.56)
$$E(x, y, \theta) = -x\langle\theta, y\rangle + I_{[-1,1]}(x),$$

obeys a margin of $m(\theta) = |\langle\theta, y\rangle|$ *w.r.t to hinge loss* $l(x, \hat{x}) = \max(1 - x\hat{x}, 0)$.

3. *Logistic regression, implemented via the parametric energy*

(4.57)
$$E(x, y, \theta) = -x\langle\theta, y\rangle + x\log(x) + (1 - x)\log(1 - x),$$

obeys a margin of 1 *w.r.t. to negative log-likelihood loss* $l(x, \hat{x}) = -x\log(\hat{x})$.

The loss function penalizes deviations of candidate solutions $x(\theta)$ and ground truth data x^*. Our training is consistent with a margin m, if non-optimal solutions $\hat{x}$ increase the value of the energy relative to the optimal solution, at least as fast as they increase the loss function, relative to the optimal solution.

We can apply Definition 4.10 to the previously discussed gradient penalty by rescaling the parametric energy with $m(\theta)$, which is a constant with respect to E, leading to energies with a constant margin of $m(\theta) = 1$. This leads to the gradient penalty

(4.58)
$$\min_\theta \sum_{i=1}^{n} ||\frac{1}{m(\theta)}\nabla E(x_i^*, y_i, \theta)||^2.$$

4.E.3 The Non-separable Case

Nevertheless, most interesting examples of parametrized energies result in training problems that are not separable, i.e. where we cannot expect to find parameters θ^* so that the $\sum_{i=1}^{N} l(x_i^*, x_i(\theta^*))$ is zero. This is especially the case if we consider the main virtue of parametrized energies: We want to constrain our parametrization with a known model, instead of the full expressibility of, for example a neural network $n(\theta, y)$ given by the energy $E(x, y, \theta) = ||x - n(\theta, y)||^2$. If we are constraining our energy, then we might expect not to perfectly reconstruct the training set, but instead yield robust results for unseen data.

A direct consequence of non-separability of E is that x_i^* is no longer equivalent to $x_i(\theta^*)$ and

(4.59)
$$\min_x E(x, y_i, \theta) \leq E(x_i^*, y_i, \theta).$$

As a consequence, we introduce slack variables ξ_i in analogy to soft-margin SVMs and penalize deviations from (4.44):

$$\begin{aligned}\min_{\theta,\xi} & R(\theta) + \sum_{i=1}^N \xi_i \\ \text{s.t. } & \xi_i \geq 0, \quad E(x, y_i, \theta) \geq E(x_i^*, y_i, \theta) - \xi_i \\ & \forall x \in \mathbb{R}^n \; \forall i = 1, \dots, N.\end{aligned} \tag{4.60}$$

This constrained problem is equivalently stated as

$$\min_\theta R(\theta) + \sum_{i=1}^N \max\left(E(x_i^*, y_i, \theta) - \min_x E(x, y_i, \theta), 0\right), \tag{4.61}$$

which we can rewrite, using the Bregman distance as

$$\min_\theta R(\theta) + \sum_{i=1}^N D_{E_\theta}(x_i^*, x_i(\theta)), \tag{4.62}$$

as $0 \in \partial E(x_i(\theta), y_i, \theta)$ by definition and (4.59) holds. The definition of margin in Definition 4.10 now immediately yields that (4.62) is a majorizer of the higher-level loss.

This is again a generalized perceptron loss, as discussed in (4.40), yet the collapse of the parametric energy is remedied by the margin requirement of Definition 4.10.

Example 4.12. *Returning to Example 4.8, we find that the margin actually matters, shown in Figure 4.10. The simple gradient penalty in (4.48) fails to recover the optimal parameters of the bilevel problem in the nonseparable case. However, scaling the parametric energy with $\frac{1}{m(\theta)} = \frac{1}{\theta}$ as in (4.58) fulfills the margin condition. This variant of the gradient penalty leads to an accurate minimizer in both the separable and the non-separable case.*

Remark 4.13 (Alternative Generalization of Margin Rescaling). Instead of the previously discussed derivation, a different surrogate optimization problem can be derived from (4.52) by fixing the margin m to maximized to 1, irregardless of the parametric energy. Introducing slack variables to this objective, cf. [280], leads to the constraint set

$$E(x_i^*, y_i, \theta) + l(x_i^*, x_i) - \xi_i \leq E(x_i, y, \theta) \tag{4.63}$$

$\forall x_i \neq x_i^*, \; \forall i = 1, \dots, N$, which can be rewritten to the generalized hinge loss formulation of

$$\begin{aligned}&\min_\theta R(\theta) \\ &+ \sum_{i=1}^N \max\left(E(x_i^*, y_i, \theta) + \max_x l(x_i^*, x) - E(x, y_i, \theta), 0\right).\end{aligned} \tag{4.64}$$

This is also an upper bound to the original bilevel loss. We will return to this single-level surrogate in the experimental section. The hinge formulation (4.64) is however not scale-invariant. If we scale the loss function l by a constant value, then the overall contribution of l to the surrogate changes.[287] also notes that this formulation potentially wrongly weighs the learned parameters as the sought margin is proportional to the loss, see also the related discussion in [287, Sec. 2.2.5] and [263].

4.F On Hessian Inversion

We know that the precise gradient with respect to the parameters of the bilevel problems is given by the inverse of the Hessian of the energy. Yet, how can optimizing the surrogate provided above return a good approximation of the true parameters, seemingly without taking second-order information into account? We briefly analyze this question for an energy E that is Legendre and twice-continuously differentiable. We not that we do not minimize the Bregman distance with respect to the first argument (as usual), the first argument is fixed to x^*, but with respect to the second argument. The gradient w.r.t to the second argument is given by

$$\nabla_x D_{E_\theta}(\hat{x}, x) = \nabla^2 E(x)(\hat{x} - x). \tag{4.65}$$

Due to our assumptions, we know that

$$\frac{\partial}{\partial \theta} x(\theta) = -(\nabla^2 E(x(\theta)))^{-1} (\frac{\partial}{\partial \theta} \nabla_x E(x(\theta)))$$

by the inverse function theorem (refer also [216]). Chaining both derivatives to find the gradient of $D_{E_\theta}(x^*, x(\theta))$ cancels the second-order derivative, i.e. by measuring the proximity of x^* and $x(\theta)$ with respect to E itself, we are choosing the only measure of proximity, where the second-order derivative cancels out.

We can also view minimizing $D_E(\hat{x}, x)$ as constructing a right Bregman-Moreau envelope [22, 23]. We may write

$$D_{E_\theta}(\hat{x}, x(\theta)) = \min_{x \in \mathbb{R}^n} I_{\nabla E(\cdot, y, \theta) = 0}(x) + D_{E_\theta}(\hat{x}, x). \tag{4.66}$$

If the term on the right hand side is coercive w.r.t to x and E is a Legendre function, then this is a proper envelope [23], the right Bregman projector onto the set $\nabla E(\cdot, y, \theta)$ (which is of course $x(\theta)$). In turn the gradient of the envelope (4.66) w.r.t to x is given by [23]

$$\nabla E(\hat{x}, y, \theta) - \nabla E(x(\theta), y, \theta) \quad \forall \hat{x} \in \operatorname{dom} E, \tag{4.67}$$

which is again the familiar gradient arising from a contrastive objective.

4.G Generalization of the Iterative Surrogate

The iterative majorizer in subsection 4.3.4 is based around the expansion of the loss function via Proposition 4.7. This linearization is however only applicable if $l(x, z) = D_w(z, x)$. In more general terms however, the iterative algorithm can be developed for arbitrary Bregman-distance induced loss functions $l(x, z) = D_w(x, z)$ via

$$l(x, z) \leq l(x, y) + \langle \nabla w(z) - \nabla w(y), y - x \rangle + D_E(y, z), \tag{4.68}$$

using the Bregman 3-point lemma [18] and the standing majorizer assumption $l(y, z) \leq D_E(y, z)$. This linearization immediately gives rise to a primal iterative surrogate variant:

Primal Iterative Surrogate Variant

$$\max_x E(\bar{x}_i, y, \theta) - E(x, y_i, \theta) + \langle \nabla w(x), \bar{x}_i - x_i^* \rangle + C \tag{4.69}$$

and dual formulation

Dual Iterative Surrogate Variant

$$E(\bar{x}_i, y, \theta) + (E \circ \nabla w^*)^* (\bar{x}_i - x_i^*, y_i, \theta) + C. \tag{4.70}$$

This dual variant is equivalent to the main version in subsection 4.3.4 if and only if the reference function w induces a symmetric Bregman distance, i.e. $D_w(x, y) = D_w(y, x)$.

CHAPTER 5

Inverting Gradients - How easy is it to break privacy in federated learning?

Contextualization

This chapter reprints the publication "Inverting Gradients - How easy is it to break privacy in federated learning", published as conference publication at the Conference on Neural Information Processing Systems (NeurIPS 2020) with co-authors Hartmut Bauermeister, Hannah Dröge and Michael Moeller (Geiping et al. [2020]´[101])

This work represents the first of two applications of optimization theory in the field of machine learning security studied in this thesis. Technically, the task in this work is fundamentally an optimization question - if the gradient of a neural network was computed based on given (secure) data, can the computation be inverted to recover this data? From a security standpoint however, the optimization question is intricately linked to data privacy. In this work we consider scenarios where an attacker tries to uncover private data, based on the parameters of a given machine learning model. If the optimization problem can be solved, then a user's privacy is broken.

In the context of previous chapters, this scenario is highly related to the bilevel optimization problem of

$$\min_x ||\theta^* - \theta(x)||^2 \quad \text{s.t. } \theta(x) = \arg\min_\theta \sum_{i=1}^N \mathcal{L}(x_i, y_i, \theta),$$

where, given the loss function of machine learning model $\mathcal{L}$ with parameters θ and data $\{x_i, y_i\}_{i=1}^N$, the problem is to recover the data $x = (x_1, \dots, x_N)$[1] used to train a model with parameters θ. This optimization problem is highly difficult to solve, due to the large possible space of data points that could generate similar parameters $\theta(x)$ and complexity of the bilevel objective - effectively making private data relatively secure. However, in collaborative learning scenarios, such as federated learning, not only the parameters θ, but also current mini-batch gradient information is available to an adversary. This allows for an attack that directly matches the gradient information, making this scenario considerably less secure, as the full complexity of the bilevel optimization is circumvented by a single-level gradient matching problem that only operates on a subset of all data.

This idea of this project and application in security was proposed by Jonas Geiping. Hannah Dröge and Hartmut Bauermeister conducted initial experiments for CNNs and fully connected models. Jonas Geiping

[1]Note that starting from this chapter, the notation of input data x and output data y switches to the common machine learning notation, instead of input data y and output data x as in variational optimization.

reviewed related work, constructed the threat model and introduced the cosine similarity loss (which was initially introduced in [102], see Chapter 6) and optimization strategies based on adversarial literature, and implemented the attack for federated SGD and federated averaging threat models for CIFAR-10 and ImageNet. Hartmut Bauermeister and Michael Moeller contributed the theoretical analysis for fully-connected layers in section 5.3. Hannah Dröge conducted experiments on CIFAR-10 shown in section 5.5, section 5.C and section 5.E, encompassing comparisons to previous work and empirical analysis. Hartmut Bauermeister conducted experiments on federated averaging in section 5.6 and section 5.E. Jonas Geiping conducted the experiments on ImageNet in the teaser and section 5.4, section 5.E, the multi-batch scenario on CIFAR-100, ablation study and threat models and experiments for dishonest settings. Michael Moeller proposed the experiments on translational invariance, greatly improved the structure of this work, compared to an earlier draft and wrote the broader impact section with Jonas Geiping. Code for this work is publicly available at https://github.com/JonasGeiping/invertinggradients.

Abstract

The idea of federated learning is to collaboratively train a neural network on a server. Each user receives the current weights of the network and in turns sends parameter updates (gradients) based on local data. This protocol has been designed not only to train neural networks data-efficiently, but also to provide privacy benefits for users, as their input data remains on device and only parameter gradients are shared. But how secure is sharing parameter gradients? Previous attacks have provided a false sense of security, by succeeding only in contrived settings - even for a single image. However, by exploiting a magnitude-invariant loss along with optimization strategies based on adversarial attacks, we show that is is actually possible to faithfully reconstruct images at high resolution from the knowledge of their parameter gradients, and demonstrate that such a break of privacy is possible even for trained deep networks. We analyze the effects of architecture as well as parameters on the difficulty of reconstructing an input image and prove that any input to a fully connected layer can be reconstructed analytically independent of the remaining architecture. Finally we discuss settings encountered in practice and show that even averaging gradients over several iterations or several images does not protect the user's privacy in federated learning applications.

5.1 Introduction

Federated or collaborative learning [70, 260] is a distributed learning paradigm that has recently gained significant attention as both data requirements and privacy concerns in machine learning continue to rise [191, 140, 301]. The basic idea is to train a machine learning model, for example a neural network, by optimizing the parameters θ of the network using a loss function $\mathcal{L}$ and exemplary training data consisting of input images x_i and corresponding labels y_i in order to solve

$$\min_{\theta} \sum_{i=1}^{N} \mathcal{L}_\theta(x_i, y_i). \tag{5.1}$$

We consider a distributed setting in which a *server* wants to solve (5.1) with the help of multiple *users* that own training data (x_i, y_i). The idea of federated learning is to only share the gradients $\nabla_\theta \mathcal{L}_\theta(x_i, y_i)$ instead of the original data (x_i, y_i) with the server which it subsequently accumulates to update the overall weights. Using gradient descent the server's updates could, for instance, constitute

$$\theta^{k+1} = \underbrace{\theta^k - \tau \sum_{i=1}^{N}}_{\text{server}} \underbrace{\nabla_\theta \mathcal{L}_{\theta^k}(x_i, y_i)}_{\text{users}}. \tag{5.2}$$

The updated parameters θ^{k+1} are sent back to the individual users. The procedure in (5.2) is called *federated SGD*. In contrast, in *federated averaging* [154, 191] each user computes several gradient descent

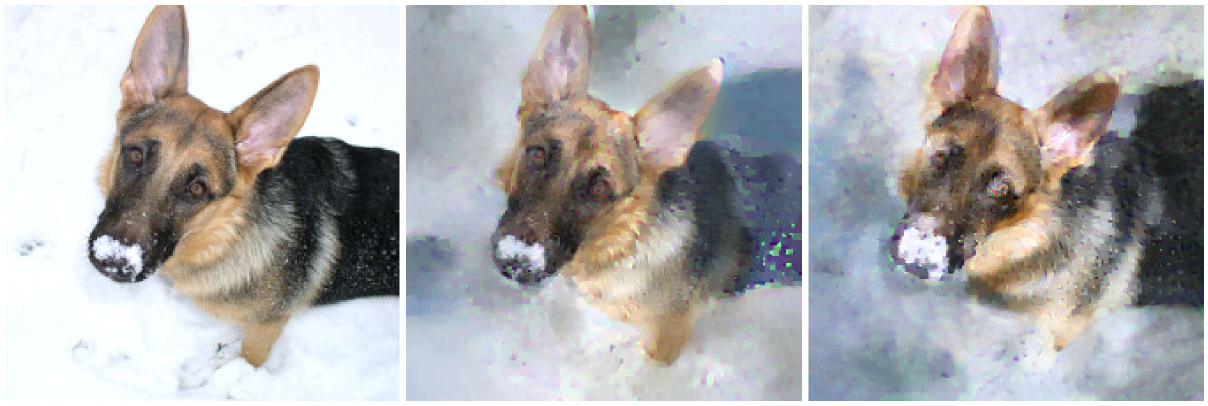


Figure 5.1: Reconstruction of an input image x from the gradient $\nabla_\theta \mathcal{L}_\theta(x, y)$. Left: Image from the validation dataset. Middle: Reconstruction from a trained ResNet-18 trained on ImageNet. Right: Reconstruction from a trained ResNet-152. In both cases, the intended privacy of the image is broken. Also note that previous attacks cannot recover ImageNet-sized data [314].

steps locally, and sends the updated parameters back to the server. Finally, information about (x_i, y_i) can be further obscured, by only sharing the mean $\frac{1}{t}\sum_{i=1}^{t} \nabla_\theta \mathcal{L}_{\theta^k}(x_i, y_i)$ of the gradients of several local examples, which we refer to as the *multi-image* setting.

Distributed learning of this kind has been used in real-world applications where user privacy is crucial, e.g. for hospital data [139] or text predictions on mobile devices [38], and it has been stated that "Privacy is enhanced by the ephemeral and focused nature of the [Federated Learning] updates" [38]: model updates are considered to contain less information than the original data, and through aggregation of updates from multiple data points, original data is considered impossible to recover. In this work we show analytically as well as empirically, that parameter gradients still carry significant information about the supposedly private input data as we illustrate in Figure 5.1. We conclude by showing that even *multi-image federated averaging* on realistic architectures does not guarantee the privacy of all user data, showing that out of a batch of 100 images, several are still recoverable.

Threat model: We investigate an *honest-but-curious* server with the goal of uncovering user data: The attacker is allowed to separately store and process updates transmitted by individual users, but may *not* interfere with the collaborative learning algorithm. The attacker may not modify the model architecture to better suit their attack, nor send malicious global parameters that do not represent the actually learned global model. The user is allowed to accumulate data locally in section 5.6. We refer to the supp. material for further commentary and mention that the attack is near-trivial under weaker constraints on the attacker.

In this paper we discuss privacy limitations of federated learning first in an academic setting, honing in on the case of gradient inversion from one image and showing that

- Reconstruction of input data from gradient information is possible for realistic deep architectures with both, trained and untrained parameters.
- With the right attack, there is little "defense-in-depth" - deep networks are as vulnerable as shallow networks.
- We prove that the input to any fully connected layer can be reconstructed analytically independent of the remaining network architecture.

Then we consider the implications that the findings have for practical scenarios, finding that

- Reconstruction of multiple, separate input images from their averaged gradient is possible in practice, over multiple epochs, using local mini-batches, and even for a local gradient averaging of up to 100 images.

5.2 Related Work

Previous related works that investigate recovery from gradient information have been limited to shallow networks of less practical relevance. Recovery of image data from gradient information was first discussed in [225, 224] for neural networks, who prove that recovery is possible for a single neuron or linear layer. For convolutional architectures, [294] show that recovery of a single image is possible for a 4-layer CNN, albeit with a significantly large fully-connected (FC) layer. Their work first constructs a “representation" of the input image, that is then improved with a GAN. [314] extends this, showing for a 4-layer CNN (with a large FC layer, smooth sigmoid activations, no strides, uniformly random weights), that missing label information can also be jointly reconstructed. They further show that reconstruction of multiple images from their averaged gradients is indeed possible (for a maximum batch size of 8). [314] also discuss deeper architectures, but provide no tangible results. A follow-up [311] notes that label information can be computed analytically from the gradients of the last layer. These works make strong assumptions on the model architecture and model parameters that make reconstructions easier, but violate the threat model that we consider in this work and lead to less realistic scenarios.

The central recovery mechanism discussed in [294, 314, 311] is the optimization of an euclidean matching term. The cost function

$$\arg\min_{x} ||\nabla_\theta \mathcal{L}_\theta(x, y) - \nabla_\theta \mathcal{L}_\theta(x^*, y)||^2 \tag{5.3}$$

is minimized to recover the original input image x^* from a transmitted gradient $\nabla_\theta \mathcal{L}_\theta(x^*, y)$. This optimization problem is solved by an L-BFGS solver [176]. Note that differentiating the gradient of $\mathcal{L}$ w.r.t to x requires a second-order derivative of the considered parametrized function and L-BFGS needs to construct a third-order derivative approximation, which is challenging for neural networks with ReLU units for which higher-order derivatives are discontinuous.

A related, but easier problem, compared to the full reconstruction of input images, is the retrieval of input attributes [193, 100] from local updates, e.g. does a person that is recognized in a face recognition system wear a hat. Information even about attributes unrelated to the task at-hand can be recovered from deeper layers of a neural network, which can be recovered from local updates.

Our problem statement is furthermore related to model inversion [97], where training images are recovered from network parameters after training. This provides a natural limit case for our setting. Model inversion generally is challenging for deeper neural network architectures [309] if no additional information is given [97, 309]. Another closely related task is inversion from visual representations [86, 85, 185], where, given the output of some intermediate layer of a neural network, a plausible input image is reconstructed. This procedure can leak some information, e.g. general image composition, dominating colors - but, depending on the given layer it only reconstructs similar images - if the neural network is not explicitly chosen to be (mostly) invertible [134]. As we prove later, inversion from visual representations is strictly more difficult than recovery from gradient information.

5.3 Theoretical Analysis: Recovering Images from their Gradients

To understand the overall problem of breaking privacy in federated learning from a theoretical perspective, let us first analyze the question if data $x \in \mathbb{R}^n$ can be recovered from its gradient $\nabla_\theta \mathcal{L}_\theta(x, y) \in \mathbb{R}^p$ analytically.

Due to the different dimensionality of x and $\nabla_\theta \mathcal{L}_\theta(x, y)$, reconstruction quality is surely is a question of the number of parameters p versus input pixels n. If $p < n$, then reconstruction is at least as difficult as image recovery from incomplete data [49, 28], but even when $p > n$, which we would expect in most computer vision applications, the difficulty of regularized “inversion" of $\nabla_\theta \mathcal{L}_\theta$ relates to the non-linearity of the gradient operator as well as its conditioning.

Interestingly, fully-connected layers take a particular role in our problem: As we prove below, the input to a fully-connected layer can always be computed from the parameter gradients analytically independent of the layer's position in a neural network (provided that a technical condition, which prevents zero-gradients, is met). In particular, the analytic reconstruction is independent of the specific types of layers that precede or succeed the fully connected layer, and a single input to a fully-connected network can always be reconstructed analytically without solving an optimization problem.The following statement is a generalization of Example 3 in [224] to the setting of arbitrary neural networks with arbitrary loss functions:

Proposition 5.1. *Consider a neural network containing a biased fully-connected layer preceded solely by (possibly unbiased) fully-connected layers. Furthermore assume for any of those fully-connected layers the derivative of the loss $\mathcal{L}$ w.r.t. to the layer's output contains at least one non-zero entry. Then the input to the network can be reconstructed uniquely from the network's gradients.*

Proof. In the following we give a sketch of the proof and refer to the supplementary material for a more detailed derivation. Consider an unbiased full-connected layer mapping the input x_l to the output, after e.g. a ReLU nonlinearity: $x_{l+1} = \max\{A_l x_l, 0\}$ for a matrix A_l of compatible dimensionality. By assumption it holds $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(x_{l+1})_i} \neq 0$ for some index i. Then by the chain rule x_l can be computed as $\left(\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(x_{l+1})_i}\right)^{-1} \cdot \left(\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(A_l)_{i,:}}\right)^T$. This allows the iterative computation of the layers' inputs as soon as the derivative of $\mathcal{L}$ w.rt. a certain layer's output is known. We conclude by noting that adding a bias can be interpreted as a layer mapping x_k to $x_{k+1} = x_k + b_k$ and that $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}x_k} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}b_k}$. ■

Another interesting aspect in view of the above considerations is that many popular network architectures use fully-connected layers (or cascades thereof) as their last prediction layers. Hence the input to those prediction modules being the output of the previous layers can be reconstructed. Those activations usually already contain some information about the input image thus exposing them to attackers. Especially interesting in that regard is the possibility to reconstruct the ground truth label information from the gradients of the last fully-connected layer as discussed in [311]. Finally, Proposition 5.1 allows to conclude that for any classification network that ends with a fully connected layer, reconstructing the input from a parameter gradient is strictly easier than inverting visual representations, as discussed in [86, 85, 185], from their last convolutional layer.

5.4 A Numerical Reconstruction Method

As image classification networks rarely start with fully connected layers, let us turn to the numerical reconstruction of inputs: Previous reconstruction algorithms relied on two components; the euclidean cost function of (5.3) and optimization via L-BFGS. We argue that these choices are not optimal for more realistic architectures and especially arbitrary parameter vectors. If we decompose a parameter gradient into its norm magnitude and its direction, we find that the magnitude only captures information about the state of training, measuring local optimality of the datapoint with respect to the current model. In contrast, the high-dimensional direction of the gradient can carry significant information, as the angle between two data points quantifies the change in prediction at one datapoint when taking a gradient step towards another [60, 148]. As such we propose to use a cost function based on angles, i.e. cosine similarity, $l(x, y) = \langle x, y\rangle / (||x||||y||)$. In comparison to (5.3), the objective is not to find images with a gradient that best fits the observed gradient, but to find images that lead to a similar change in model prediction as the (unobserved!) ground truth. This is equivalent to minimizing the euclidean cost function, if one additionally constrains both gradient vectors to be normalized to a magnitude of 1.

We further constrain our search space to images within $[0, 1]$ and add only total variation [245] as a simple image prior to the overall problem, cf. [294]:

$$\arg \min_{x \in [0,1]^n} 1 - \frac{\langle \nabla_\theta \mathcal{L}_\theta(x, y), \nabla_\theta \mathcal{L}_\theta(x^*, y)\rangle}{||\nabla_\theta \mathcal{L}_\theta(x, y)||||\nabla_\theta \mathcal{L}_\theta(x^*, y)||} + \alpha \, \mathrm{TV}(x). \tag{5.4}$$

Table 5.1: PSNR mean and standard deviation for 100 experiments on the first images of the CIFAR-10 validation data set over two different networks with trained an untrained parameters.

Architecture	LeNet (Zhu)		ResNet20-4	
Trained	False	True	False	True
Eucl. Loss + L-BFGS	$\mathbf{46.25 \pm 12.66}$	13.24 ± 5.44	10.29 ± 5.38	6.90 ± 2.80
Proposed	18.00 ± 3.33	$\mathbf{18.08 \pm 4.27}$	$\mathbf{19.83 \pm 2.96}$	$\mathbf{13.95 \pm 3.38}$

Secondly, we note that our goal of finding some inputs x in a given interval by minimizing a quantity that depends (indirectly, via their gradients) on the outputs of intermediate layers, is related to the task of finding adversarial perturbations for neural networks [275, 184, 10]. As such, we minimize (5.4) only based on the sign of its gradient, which we optimize with Adam [145] with step size decay. Note though that signed gradients only affect the first and second order momentum for Adam, with the actual update step still being unsigned based on accumulated momentum, so that an image can still be accurately recovered.

Applying these techniques leads to the reconstruction observed in Figure 5.1. Further ablation of the proposed mechanism can be found in the appendix. We provide a `pytorch` implementation at https://github.com/JonasGeiping/invertinggradients.

Remark 5.2 (Optimizing label information). While we could also consider the label y as unknown in (5.4) and optimize jointly for (x, y) as in [314], we follow [311] who find that label information can be reconstructed analytically for classification tasks. Thus, we consider label information to be known.

5.5 Single Image Reconstruction from a Single Gradient

Similar to previous works on breaking privacy in a federated learning setting, we first focus in the reconstruction of a single input image $x \in \mathbb{R}^n$ from the gradient $\nabla_\theta \mathcal{L}_\theta(x, y) \in \mathbb{R}^p$. This setting serves as a proof of concept as well as an upper bound on the reconstruction quality for the multi-image distributed learning settings we consider in section 5.6. While previous works have already shown that a break of privacy is possible for single images, their experiments have been limited to rather shallow, smooth, and untrained networks. In the following, we compare our proposed approach to prior works, and conduct detailed experiments on the effect that architectural- as well as training-related choices have on the reconstruction. All hyperparameter settings and more visual results for each experiment are provided in the supp. material.

Comparison to previous approaches. We first validate our approach by comparison to the Euclidean loss (5.3) optimized via L-BFGS considered in [294, 314, 311]. This approach can often fail due to a bad initialization, so we allow a generous setting of 16 restarts of the L-BFGS solver. For a quantitative comparison we measure the mean PSNR of the reconstruction of 32×32 CIFAR-10 images over the first 100 images from the validation set using the same shallow and smooth CNN as in [314], which we denote as "LeNet (Zhu)" as well as a ResNet architecture, both with trained and untrained parameters. Table 5.1 compares the reconstruction quality of euclidean loss (5.3) with L-BFGS optimization (as in [294, 314, 311]) with the proposed approach. The former works extremely well for the untrained, smooth, shallow architecture, but completely fails on the trained ResNet. We note that [294] applied a GAN to enhance image quality from the LBFGS reconstruction, which, however, fails, when the representative is too distorted to be enhanced. Our approach provides recognizable images and works particularly well on the realistic setting of a trained ResNet as we can see in Figure 5.2. Interestingly, the reconstructions on the trained ResNet have a better visual quality than those of the untrained ResNet, despite their lower PSNR values according to Table 5.1. Let us study the effect of trained network parameters in an even more realistic setting, i.e., for reconstructing ImageNet images from a ResNet-152.

Trained vs. untrained networks. If a network is trained and has sufficient capacity for the gradient of the loss function $\mathcal{L}_\theta$ to be zero for different inputs, it is obvious that they can never be distinguished from their gradient. In practical settings, however, owing to stochastic gradient descent, data augmentation

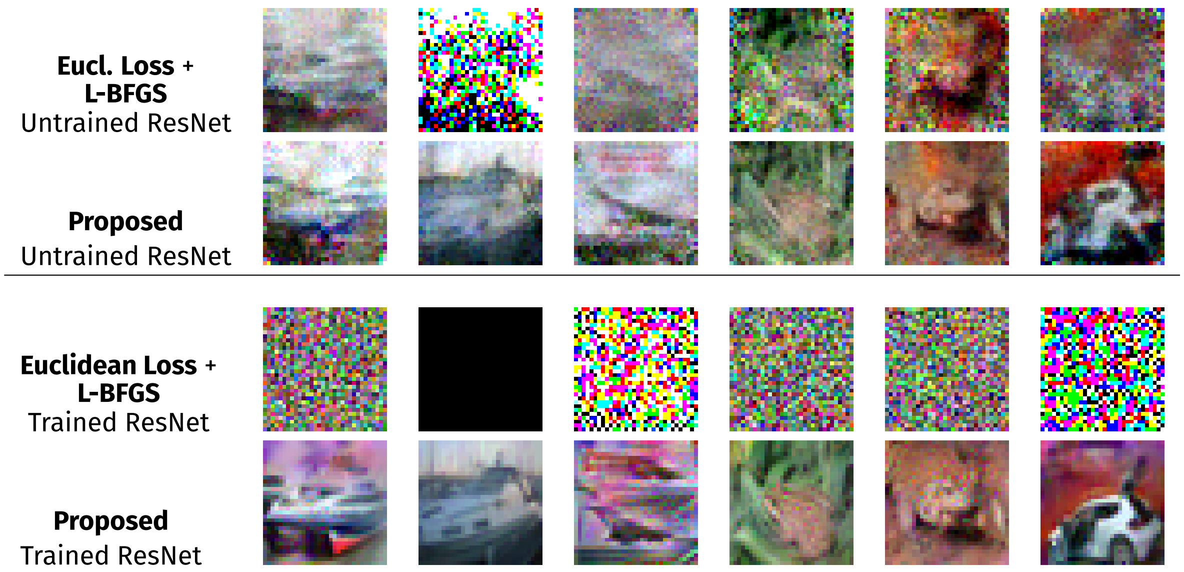


Figure 5.2: Baseline comparison for the network architectures shown in [294, 314].We show the first 6 images from the CIFAR-10 validation set.

and a finite number of training epochs, the gradient of images is rarely entirely zero. While we do observe that image gradients have a much smaller magnitude in a trained network than in an untrained one, our magnitude-oblivious approach of (5.4) still recovers important visual information from the direction of the trained gradients.

We observe two general effects on trained networks that we illustrate with our ImageNet reconstructions in Figure 5.3: First, reconstructions seem to become *implicitly biased* to typical features of the same class in the training data, e.g., the more blueish feathers of the capercaillie in the 5th image, or the large eyes of the owl in the inset figure. Thus, although the overall privacy of most images is clearly breached, this effect at least obstructs the recovery of fine scale details or the image's background.

Second, we find that the data augmentation used during the training of neural networks leads to trained networks that make the localization of objects more difficult: Notice how few of the objects in Figure 5.3 retain their original position and how the snake and gecko duplicate. Thus, although image reconstruction with networks trained with data augmentation still succeeds, some location information is lost.

Translational invariant convolutions. Let us study the ability to obscure the location of objects in more detail by testing how a conventional convolutional neural network, that uses convolutions with zero-padding, compares to a provably translationally invariant CNN, that uses convolutions with circular padding. As shown in Figure 5.4, while the conventional CNN allows for recovery of a rather high quality image (left), the translationally invariant network makes the localization of objects impossible (right) as the original object is separated. As such we identify the common zero-padding as a source of privacy risk.

Network Depth and Width. For classification accuracy, the depth and number of channels of each layer of a CNN are very important parameters, which is why we study their influence on our reconstruction. Figure 5.5 shows that the reconstruction quality measurably increase with the number of channels. Yet, the larger network width is also accompanied with an increasing variance of experimental success. However with multiple restarts of the experiment, better reconstructions can be produced for wider networks, resulting in PSNR values that increases from 19 to almost 23 for when increasing the number of channels from 16 to 128. As such, greater network width increases the computational effort of the attacker, but does not provide greater security.

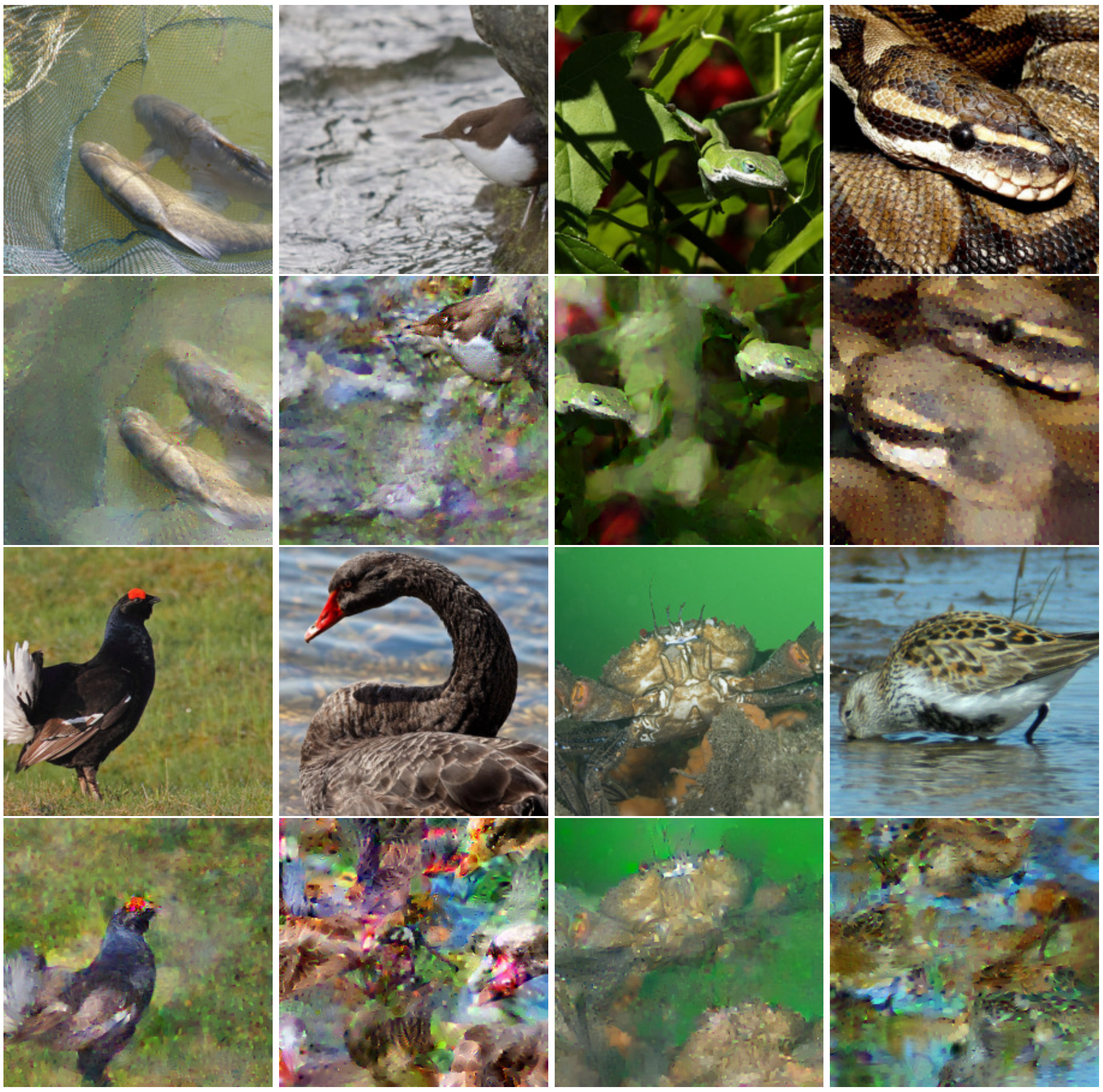


Figure 5.3: Single-Image Reconstruction from the parameter gradients of trained ResNet-152. 1st and 3rd row: Ground Truth. 2nd and 4th row: Reconstruction. We check every 1000th image of the ILSVRC2012 validation set. The amount of information leaked per image is highly dependent on image content - while some examples like the two tenches are highly compromised, the black swan leaks almost no usable information.

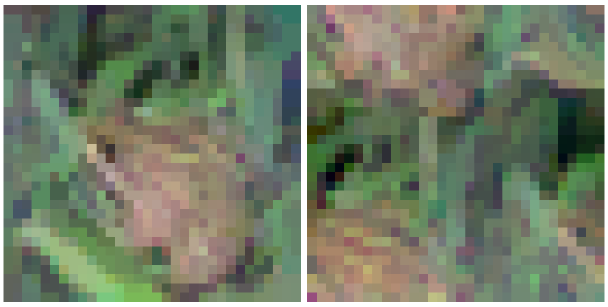


Figure 5.4: Reconstruction for a network with zero-padded convolutions (left) and a network with circular padded convolutions (right). The zero-padded convolutions leak positional information, compared to the circular convolution.

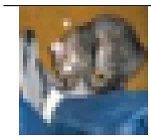
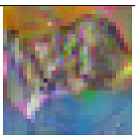
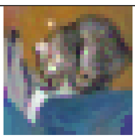
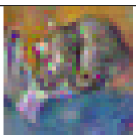
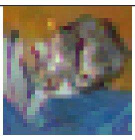
Original	**ResNet-18** with base width:			**ResNet-34**	**ResNet-50**
	16	64	128		
					
PSNR	17.24	17.37	25.25	18.62	21.36
Avg. PSNR	19.02	22.04	22.94	21.59	20.98
Std.	2.84	5.89	6.83	4.49	5.57

Figure 5.5: Reconstructions of the original image (left) for multiple ResNet architectures. The PSNR value refers to the displayed image while the avg. PSNR is calculated over the first 10 CIFAR-10 images. The standard deviation is the average standard deviation of one experiment under a given architecture. The ResNet-18 architecture is displayed for three different widths.

Looking at the reconstruction results we obtain from ResNets with different depths, the proposed attack degrades very little with an increased depth of the network. In particular - as illustrated in Fig. 5.3, even faithful ImageNet reconstructions through a ResNet-152 are possible.

5.6 Distributed Learning with Federated Averaging and Multiple Images

So far we have only considered recovery of a single image from its gradient and discussed limitations and possibilities in this setting. We now turn to strictly more difficult generalized setting of *Federated Averaging* [191, 192, 236] and *multi-image* reconstruction, to show that the proposed improvements translate to this more practical case as well, discussing possibilities and limits in this application.

Instead of only calculating the gradient of a network's parameters based on local data, federated averaging performs multiple update steps on local data before sending the updated parameters back to the server. Following the notation of [191], we let the local data on the user's side consist of n images. For a number E of local epochs the user performs $\frac{n}{B}$ stochastic gradient update steps per epoch, where B denotes the local mini-batch size, resulting in a total number of $E\frac{n}{B}$ local update steps. Each user i then sends the locally updated parameters $\tilde{\theta}_i^{k+1}$ back to the server, which in turn updates the global parameters θ^{k+1} by averaging over all users.

We empirically show that even the setting of federated averaging with $n \geq 1$ images is potentially amenable for attacks. To do so we try to reconstruct the local batch of n images by the knowledge of the local update $\tilde{\theta}_i^{k+1} - \theta^k$. In the following we evaluate the quality of the reconstructed images for different choices of n, E and B. We note that the setting studied in the previous sections corresponds to

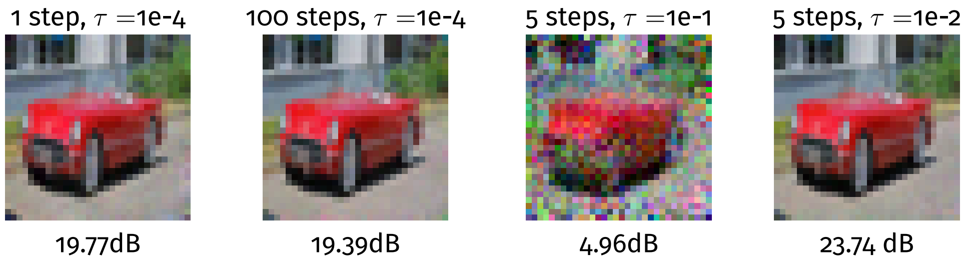


Figure 5.6: Illustrating the influence of the number of local update steps and the learning rate on the reconstruction: The left two images compare the influence of the number of gradient descent steps for a fixed learning rate of τ =1e-4. The two images on the right result from varying the learning rate for a fixed number of 5 gradient descent steps. PSNR values are shown below the images.

$n = 1$, $E = 1$, $B = 1$. For all our experiments we use an untrained ConvNet.

Multiple gradient descent steps, $B = n = 1$**,** $E > 1$**:**
Fig. 5.6 shows the reconstruction of $n = 1$ image for a varying number of local epochs E and different choices of learning rate τ. Even for a high number of 100 local gradient descent steps the reconstruction quality is unimpeded. The only failure case we were able to exemplify was induced by picking a high learning rate of 1e-1. This setup, however, corresponds to a step size that would lead to a divergent training update, and as such does not provide useful information for the federated learning.

Multi-Image Recovery, $B = n > 1$**,** $E = 1$**:**
So far we have considered the recovery of a single image only, and it seems reasonable to believe that averaging the gradients of multiple (local) images before sending an update to the server, restores the privacy of federated learning. While such a multi-image recovery has been considered in [314] for $B \leq 8$, we demonstrate that the proposed approach is capable of restoring some information from a batch of 100 averaged gradients: While most recovered images are unrecognizable (as shown in the supplementary material), Fig. 5.7 shows the 5 most recognizable images and illustrates that even averaging the gradient of 100 images does not entirely secure the private data. Most surprising is that the distortions arising from batching are *non-uniform*. One could have expected all images to be equally distorted and near-irrecoverable, yet some images are highly distorted and others only to an extend at which the pictured object can still be recognized easily, which demonstrates that privacy leaks are conceivable even for large batches of image data.

General case
We also consider the general case of multiple local update steps using a subset of the whole local data in each mini batch gradient step. An overview of all conducted experiments is provided in Table 5.2. For each setting we perform 100 experiments on the CIFAR-10 validation set. For multiple images in a mini batch we only use images of different labels avoiding permutation ambiguities of reconstructed images of the same label. As to be expected, the single image reconstruction turns out to be most amenable to attacks in terms of PSNRs values. Despite a lower performance in terms of PSNR, we still observe privacy leakage for all multi-image reconstruction tasks, including those in which gradients in random mini-batches are taken. Comparing the full-batch, 8 images examples for 1 and 5 epochs, we see that our previous observation that multiple epochs do not make the reconstruction problem more difficult, extends to multiple images. For a qualitative assessment of reconstructed images of all experimental settings of Table 5.2, we refer to the supplementary material.

5.7 Conclusions

Federated learning is a modern paradigm shift in distributed computing, yet its benefits to privacy are not as well understood yet. We shed light into possible avenues of attack, analyze the ability to reconstruct the input to any fully connected layer analytically, propose a general optimization-based attack based

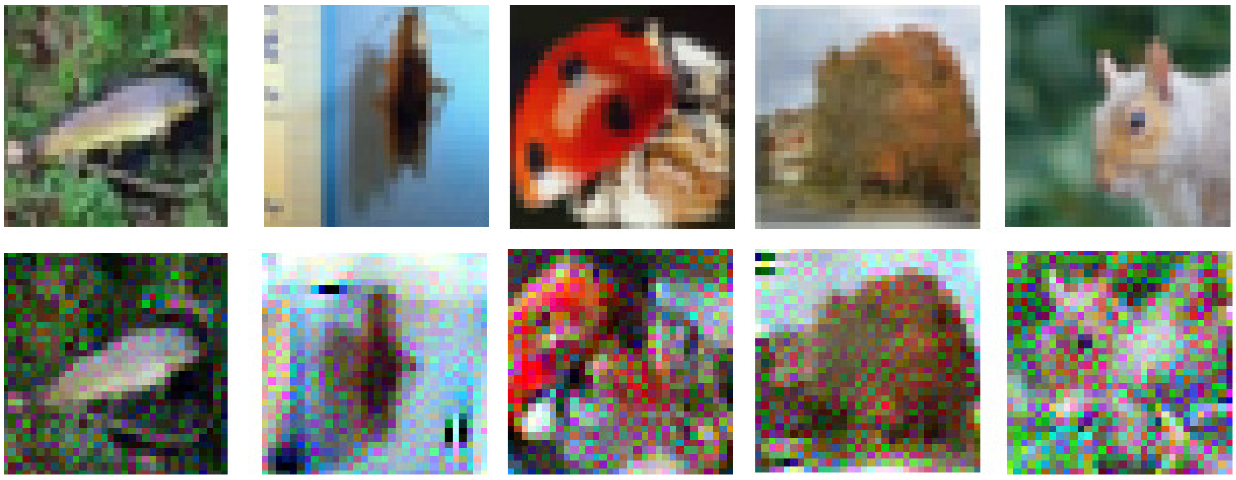


Figure 5.7: Information leakage for a batch of 100 images on CIFAR-100 for a ResNet32-10. Shown are the 5 *most* recognizable images from the whole batch. Although most images are unrecognizable, privacy is broken even in a large-batch setting. We refer to the supplementary material for all images.

Table 5.2: PSNR statistics for various federated averaging settings, averaged over experiments on the first 100 images of the CIFAR-10 validation data set.

1 epoch			5 epochs	
4 images	8 images		1 image	8 images
batchsize 2	batchsize 2	batchsize 8	batchsize 1	batchsize 8
16.92 ± 2.10	14.66 ± 1.12	16.49 ± 1.02	25.05 ± 3.28	16.58 ± 0.96

on cosine similarity of gradients, and discuss its effectiveness for different types of architectures and scenarios. In contrast to previous work we show that even *deep*, *nonsmooth* networks trained with ImageNet-sized data such as modern computer vision architectures like ResNet-152 are vulnerable to attacks - even when considering *trained* parameter vectors. Our experimental results clearly indicate that privacy is not an innate property of collaborative learning algorithms like federated learning, and that secure applications to be closely investigated on a case-by case basis for their potential of leaking private information. Provable differential privacy possibly remains the only way to guarantee security, possibly even for larger batches of data points.

5.8 Broader Impact - Federated Learning does not guarantee privacy

Recent works on privacy attacks in federated learning setups ([225, 224, 294, 314, 311]) have hinted at the fact that previous hopes that "Privacy is enhanced by the ephemeral and focused nature of the [Federated Learning] updates" [38] are not true in general. In this work, we demonstrated that improved optimization strategies such as a cosine similarity loss and a signed Adam optimizer allow for image recovery in a federated learning setup in industrially realistic settings for computer vision: Opposed to the idealized architectures of previous works we demonstrate that image recovery is possible in deep, non-smooth and trained architectures over multiple federated averaging steps of the optimizer and even in batches of 100 images.

We note that image classification is possibly especially vulnerable to these types of attacks, given the inherent structure of image data, the size of image classification networks, and the comparatively small number of images a single user might own, relative to other personal information. On the other hand, this attack is likely only a first step towards stronger attacks. Therefore, this work points out that the question how to protect the privacy of our data while collaboratively training highly accurate machine learning approaches remains largely unsolved: While differential privacy offers provable guarantees, it also reduces the accuracy of the resulting models significantly [136]. As such differential privacy and secure aggregation can be costly to implement so that there is some economic incentive for data companies to use only basic federated learning. For a more general discussion, see [293]. There is strong interest in further research on privacy preserving learning techniques that render the attacks proposed in this paper ineffective. This might happen via defensive mechanisms or via computable guarantees that allow practitioners to verify whether their specific application is vulnerable to such an attack and within which bounds.

Acknowledgements:

This research was directly supported by the University of Siegen. HB and MM further received support from the German Research Foundation (DFG) under grant MO 2962/2-1.

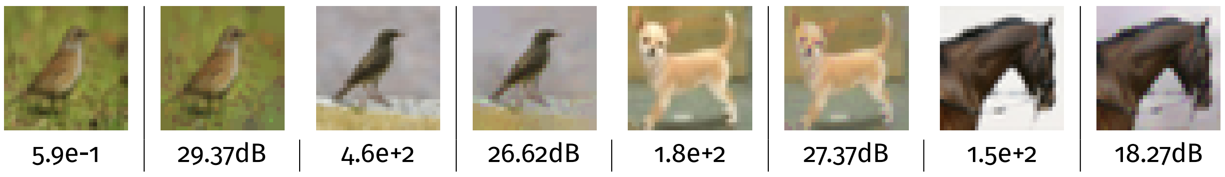


Figure 5.8: Label flipping. Images can be easily reconstructed when two rows in the parameters of the final classification layer are permuted. Below each input image is given the gradient magnitude, below each output image its PSNR. Compare these results to the additional examples in Figure 5.10

This appendix reprints the appendix of Geiping et al. [2020][101].

5.A Variations of the threat model

In this work we consider a *honest-but-curious* threat model as discussed in the introduction. Straying from this scenario could be done primarily in two ways: First by changing the architecture, and second by keeping the architecture non-malicious, but changing the global parameters sent to the user.

5.A.1 Dishonest Architectures

So far we assumed that the server operates under an *honest-but-curious* model, and as such would not modify the model maliciously to make reconstruction easier. If we instead allow for this, then reconstruction becomes nearly trivial: Several mechanisms could be used: Following Proposition 5.1, the server could, for example, place a fully-connected layer in the first layer, or even directly connect the input to the end of the network by concatenation. Slightly less obvious, the model could be modified to contain reversible blocks [59, 134]. These blocks allow the recovery of input from their outputs. From Proposition 5.1 we know that we can reconstruct the input to the classification layer, so this allows for immediate access to the input image. If the server maliciously introduces separate weights or sub-models for each batch example, then this also allows for a recovery of an arbitrarily large batch of data. Operating in a setting, where such behavior is possible would require the user (or a provider trusted by the user) to vet any incoming model either manually or programmatically.

5.A.2 Dishonest Parameter Vectors

However, even with a fixed *honest* architecture, a malicious choice of global parameters can significantly influence reconstruction quality. For example, considering the network architecture in [314] which does not contain strides and flattens convolutional features, the dishonest server could set all convolution layers to represent the identity [109], moving the input through the network unchanged up to the classification layer, from which the input can be analytically computed as in Proposition 5.1. Likewise for an architecture that contains strides to a recognizable lower resolution [294], the input can be recovered immediately albeit in a smaller resolution when the right parameter vector is sent to the user.

Such a specific choice of parameters is however likely detectable. A subtler approach, as least possible in theory, would be to optimize the network parameters themselves that are sent to the user so that reconstruction quality from these parameters is maximized. While such an attack is likely to be difficult to detect on the user-side, it would also be very computationally intensive.

Label flipping. There is even a cheaper alternative. According to section 5.5, very small gradient vectors may contain less information. A simple way for a dishonest server to boost these gradients is to permute two rows in the weight matrix and bias of the classification layer, effectively flipping the semantic meaning of a label. This attack is difficult to detect for the user (as long as the gradient magnitude stays within usual bounds), but effectively tricks him into differentiating his network w.r.t to the wrong label. Fig. 5.8 shows that this mechanism can allow for a reliable reconstruction with boosted PSNR scores, as the effect of the trained model is negated.

5.B Experimental Details

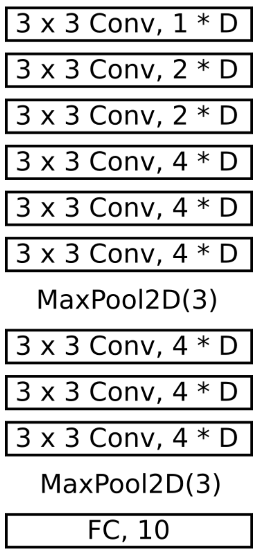


Figure 5.9: Network architecture *ConvNet*, consisting of 8 convolution layers, specified with corresponding number of output channels. Each convolution layer is followed by a batch normalization layer and a ReLU layer. D scales the number of output channels and is set to $D = 64$ by default.

5.B.1 Federated Averaging

The extension of (5.4) to the case of federated averaging (in which multiple local update steps are taken and sent back to the server) is straightforward. Notice first, that given old parameters θ^k, local updates θ^{k+l}, learning rate τ, and knowledge about the number of update steps[2], the update can be rewritten as the average of updated gradients.

$$\theta^{k+l} = \theta^k - \tau \sum_{m=1}^{l} \nabla_{\theta^{k+m}} \mathcal{L}_{\theta^{k+m}}(x, y) \tag{5.5}$$

Subtracting θ^k from θ^{k+l}, we simply apply the proposed approach to the resulting average of updates:

$$\arg\min_{x \in [0,1]^n} 1 - \frac{\langle \sum_{m=1}^{l} \nabla_{\theta^{k+m}} \mathcal{L}_{\theta^{k+m}}(x, y), \sum_{m=1}^{l} \nabla_{\theta^{k+m}} \mathcal{L}_{\theta^{k+m}}(x^*, y) \rangle}{|| \sum_{m=1}^{l} \nabla_{\theta^{k+m}} \mathcal{L}_{\theta^{k+m}}(x, y) |||| \sum_{m=1}^{l} \nabla_{\theta^{k+m}} \mathcal{L}_{\theta^{k+m}}(x^*, y) ||} + \alpha \, \mathrm{TV}(x). \tag{5.6}$$

Using automatic differentiation, we backpropagate the gradient w.r.t to x from the average of update steps.

5.B.2 ConvNet

We use a ConvNet architecture as a baseline for our experiments as it is relatively fast to optimize, reaches above 90% accuracy on CIFAR-10 and includes two max-pooling layers. It is a rough analogue to AlexNet [155]. The architecture is described in Figure 5.9.

5.B.3 Ablation Study

We provide an ablation for proposed choices in Table 5.3. We note that two things are central, the Adam optimizer and the similarity loss. Total variation is a small benefit, and using signed gradients is a minor benefit.

[2]We assume that the number of local updates is known to the server, yet this could also be found by brute-force, given that l is a small integer.

Table 5.3: Ablation Study for the proposed approach for a trained ResNet-18 architecture, trained on CIFAR-10. Reconstruction PSNR scores are averaged over the first 10 images of the CIFAR-10 validation set (Standard Error in parentheses).

Basic Setup	20.12 dB (± 1.02)
L2 Loss instead of cosine similarity	15.13 dB (± 0.70)
Without total variation	19.96 dB (± 0.75)
With L-BFGS instead of Adam	5.13 dB (± 0.50)

5.C Hyperparameter Settings

In our experiments we reconstruct the network's input using Adam based on signed gradients as optimization algorithm and cosine similarity as cost function as described in section 5.4. It is important to note that the optimal hyperparameters for the attack depend on the specific attack scenario - that the attack fails with default parameters is no guarantee for security. We always initialize our reconstructions from a Gaussian distribution with mean 0 and variance 1 (Note that the input data is normalized as usual for all considered datasets) and set the step size of the optimization algorithm within $[0.01, 1]$. We use a smaller step sizes of 0.1, for the wider and deeper networks and a larger step sizes of 1 for the federated averaging experiments, with 0.1 being the default choice. The optimization runs for up to 24000 iterations. The step size decay is always fixed, occuring after $\frac{3}{8}$, $\frac{5}{8}$ and $\frac{7}{8}$ of iterations and reducing the learning rate by a factor of 0.1 each time. The number of iterations is a generally conservative estimate, privacy can often be broken much earlier.

We tweak the total variation parameter depending on the specific attack scenario, however note that its effect on avg. PSNR is mostly minor as seen in Table 5.3. When not otherwise noted we default to a value of 0.01.

Remark 5.3 (Restarts). Generally, multiple restarts of the attack from different random initializations can improve the attack success moderately. However they also increase the computational requirements significantly. To allow for quantitative experimental evaluations of multiple images, we do not consider restarts in this work (aside from Sec. 5 where we apply them to improve results of the competing LBFGS solver) - but stress that an attacker with enough ressources could further improve his attack by running it with multiple restarts.

5.C.1 Settings for the experiments in Sec. 5.5

Comparison to previous approaches

For comparison with baselines, we re-implement the network from [314], which we dub LeNet (Zhu) in the following, and additionally run all experiments for the ResNet20-4 architecture. We base both the network and the approach on code from the authors of [314], [3]. For the LBFGS-L2 optimization we use a learning rate of $1e-4$ and 300 iterations. For the ResNet experiments we use the generous amount of 8 restarts and for the faster to optimize LeNet (Zhu) architecture we use the even higher number of 16 restarts. All experiment conducted with the proposed approach only use one restart, 4800 iterations, a learning rate of 0.1 and TV regularization parameters as detailed in Table 5.4. Note that in the described settings the proposed method took significantly less time to optimize than the LBFGS optimization.

Spatial Information

The experiments on spatial information are performed on the ConvNet architecture with $D = 64$ channels.

5.C.2 Setting for experiments in Sec. 5.6

For the five cases consider in Table 2 we consider an untrained ConvNet, a learning rate of 1, 4800 iterations, one restart and the TV regularization parameters as given in Table 5.5. Each of the 100 experiments uses

[3]https://github.com/mit-han-lab/dlg

Architecture	LeNet (Zhu)		ResNet20-4	
Trained	False	True	False	True
TV	10^{-2}	10^{-3}	0	10^{-2}

Table 5.4: TV regularization values used for the proposed approach in the baseline experiments of section 5.5

Number of epochs E	1	1	1	5	5
Number of local images n	4	8	8	1	8
Mini-batch size B	2	2	8	1	8
TV	10^{-6}	10^{-6}	10^{-4}	10^{-4}	10^{-4}

Table 5.5: Total variation weights for the reconstruction of network input in the experiments in Sec. 4.2

different images, i.e. each experiments uses the images of the CIFAR-10 validation set following the ones used in the previous experiment. As multiple images of the same label in one mini-batch cause an ambiguity in the ordering of images w.r.t. that label, we do not consider that case. If an image with an already encountered label is about to be added to the respective mini-batch we skip that image and use the next image of the validation set with a different label.

5.D Proofs for section 5.2

In the following we give a more detailed proof of Prop 3.1, which is follows directly from the two propositions below:

Proposition 5.1. *Let a neural network contain a biased fully-connected layer at some point, i.e. for the layer's input $x_l \in \mathbb{R}^{n_l}$ its output $x_{l+1} \in \mathbb{R}^{n_{l+1}}$ is calculated as $x_{l+1} = \max\{y_l, 0\}$ for*

$$y_l = A_l x_l + b_l, \tag{5.7}$$

for $A_l \in \mathbb{R}^{n_{l+1} \times n_l}$ and $b_l \in \mathbb{R}^{n}_{l+1}$. Then the input x_l can be reconstructed from $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}A_l}$ and $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}b_l}$, if there exists an index i s.t. $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(b_l)_i} \neq 0$.

Proof. It holds that $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(b_l)_i} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(y_l)_i}$ and $\frac{\mathrm{d}y_i}{\mathrm{d}(A_l)_{i,:}} = x^T$. Therefore

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\left(A_l\right)_{i,:}} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\left(y_l\right)_i} \cdot \frac{\mathrm{d}\left(y_l\right)_i}{\mathrm{d}\left(A_l\right)_{i,:}} \tag{5.8}$$

$$= \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\left(b_l\right)_i} \cdot x_l^T \tag{5.9}$$

for $(A_l)_{i,:}$ denoting the i^{th} row of A_l. Hence x_l can can be uniquely determined as soon as $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(b_l)_i} \neq 0$. ■

Proposition 5.2. *Consider a fully-connected layer (not necessarily including a bias) followed by a ReLU activation function, i.e. for an input $x_l \in \mathbb{R}^{n_l}$ the output $x_{l+1} \in \mathbb{R}^{n_{l+1}}$ is calculated as $x_{l+1} = \max\{y_l, 0\}$ for*

$$y_l = A_l x_l, \tag{5.10}$$

where the maximum is computed element-wise. Now assume we have the additional knowledge of the derivative w.r.t. to the output $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}x_{l+1}}$. Furthermore assume there exists an index i s.t. $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(x_{l+1})_i} \neq 0$. Then the input v can be derived from the knowledge of $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}A_l}$.

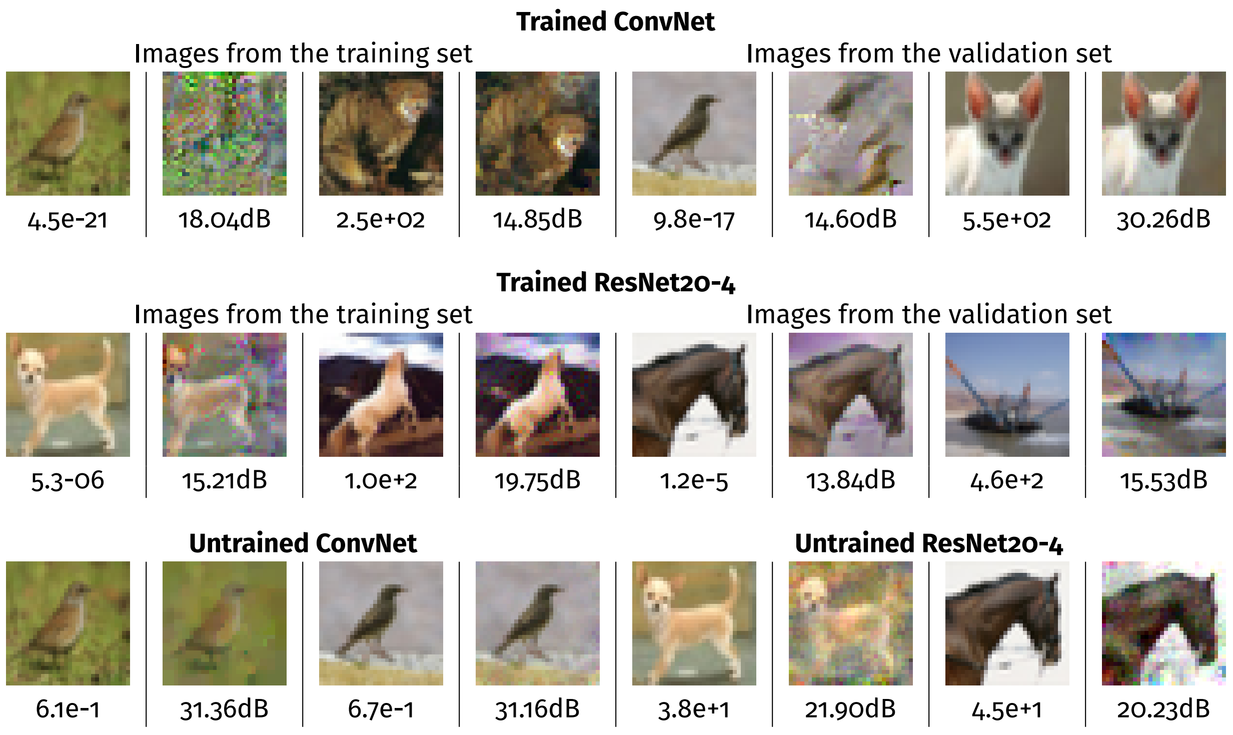


Figure 5.10: Reconstruction of images for the *trained* ConvNet model (Top) and ResNet20-4 (middle). We show reconstructions of the **worst-case** image and **best case** image from CIFAR-10, based on gradient magnitude for both the training and the validation set. Below each input image is given the gradient magnitude, below each output image its PSNR. The bottom row shows reconstructions for the worst-case examples for untrained models.

Proof. As $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(x_{l+1})_i} \neq 0$ it holds that $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(y_l)_i} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}(x_{l+1})_i}$ and it follows that

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\,(A_l)_{i,:}} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\,(y_l)_i} \cdot \frac{\mathrm{d}\,(y_l)_i}{\mathrm{d}\,(A_l)_{i,:}} \tag{5.11}$$

$$= \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\,(x_{l+1})_i} \cdot x_l^T . \tag{5.12}$$

■

5.E Additional Examples

5.E.1 Additional CIFAR-10 examples

Figure 5.10 shows additional "extreme" examples for CIFAR-10, reconstructing the image with lowest and the image with largest gradient magnitude for the training and validation set of CIFAR-10 for trained and untrained ConvNet and ResNet20-4 models.

5.E.2 Visualization of experiments in Sec. 5.5

Network Width

The reconstructions for the first six CIFAR images for different width ResNet-18 architectures are given in Figure 5.11.

Network Depth

The experiments concerning the network depth are performed for different deep ResNet architectures. Multiple reconstruction results for different deep networks are shown in Figure 5.12.

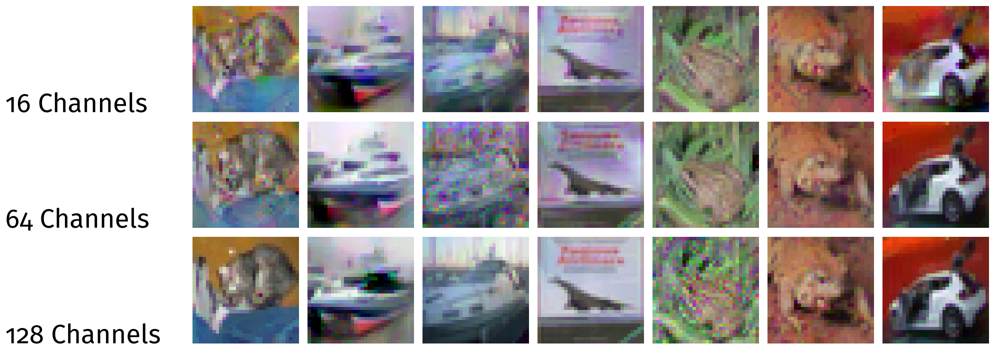


Figure 5.11: Reconstructions using ResNet-18 architectures with different widths.

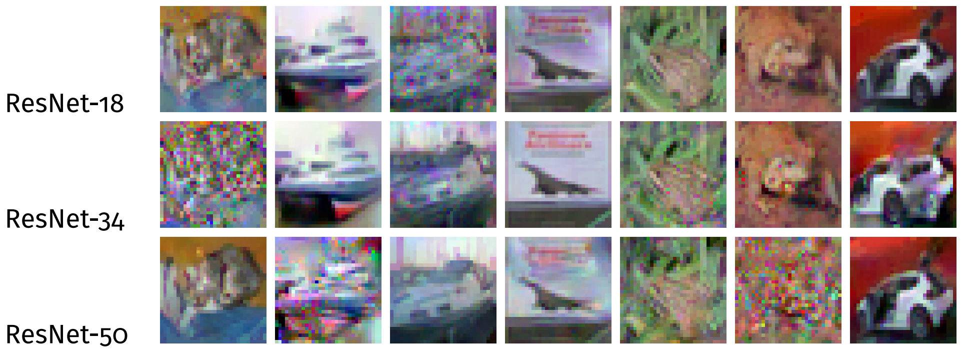


Figure 5.12: Reconstructions using different deep ResNet architectures.

5.E.3 More ImageNet examples for Sec. 5.5

Figure 5.13 shows further instructive examples of reconstructions for ImageNet validation images for a trained ResNet-18 (the same setup as Figure 5.3 in the main paper). We show a very good reconstruction (German shepherd), a good, but translated reconstruction (giant panda) and two failure cases (ambulance and flower). For the ambulance, for example, the actual writing on the ambulance car is still hidden. For the flower, the exact number of petals is hidden. Also, note how the reconstruction of the giant panda is much clearer than that of the tree stump co-occurring in the image, which we consider an indicator of the self-regularizing effect described in section 5.5.

Figure 5.14 and Figure 5.15 show more examples. We note that the examples in these figures and in Figure 5.3 are not handpicked, but chosen neutrally according to their ID in the ILSVRC2012, ImageNet, validation set. The ID for each image is obtained by sorting the synset that make up the dataset in increasing order according to their synset ID and sorting the images within each synset according to their synset ID in increasing order. This is the default order in `torchvision`.

5.E.4 Multi-Image Recovery of Sec. 5.6

For multi-image recovery, we show the full set of 100 images in Figure 5.21, we recommend to zoom in to a digital version of the figure. The success rate for separate images is semi-random, depending on the initialization.

5.E.5 General case of Sec. 5.6

We show the results for the first ten experiments in Figure 5.16, Figure 5.17, Figure 5.18, Figure 5.19, Figure 5.20. In Figure Figure 5.16 we even show all 100 experiments as there only one image is used per experiment.

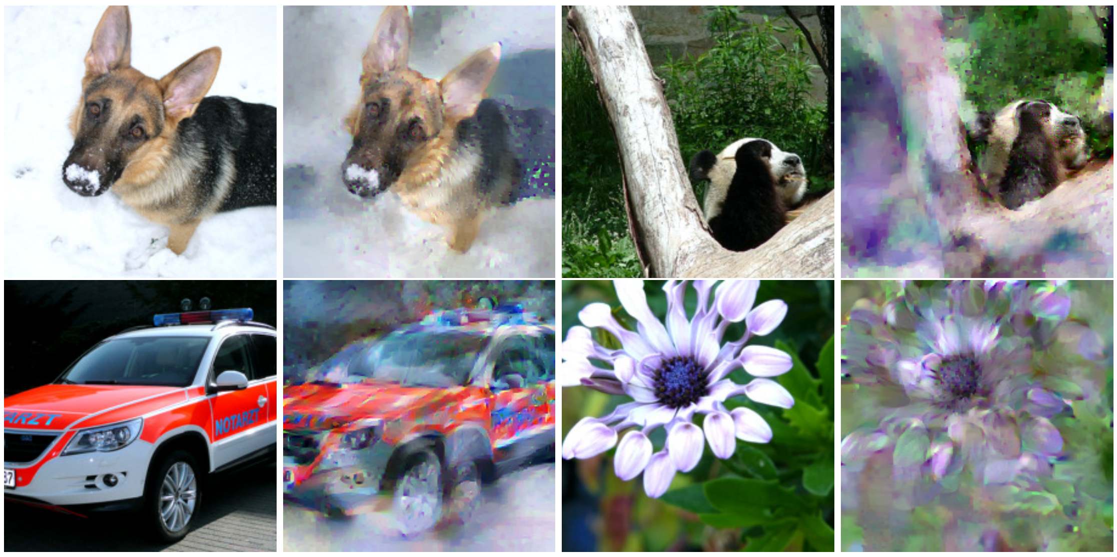


Figure 5.13: Additional qualitative ImageNet examples, failure cases and positive cases for a trained ResNet-18. Images taken from the ILSVRC2012 validation set.

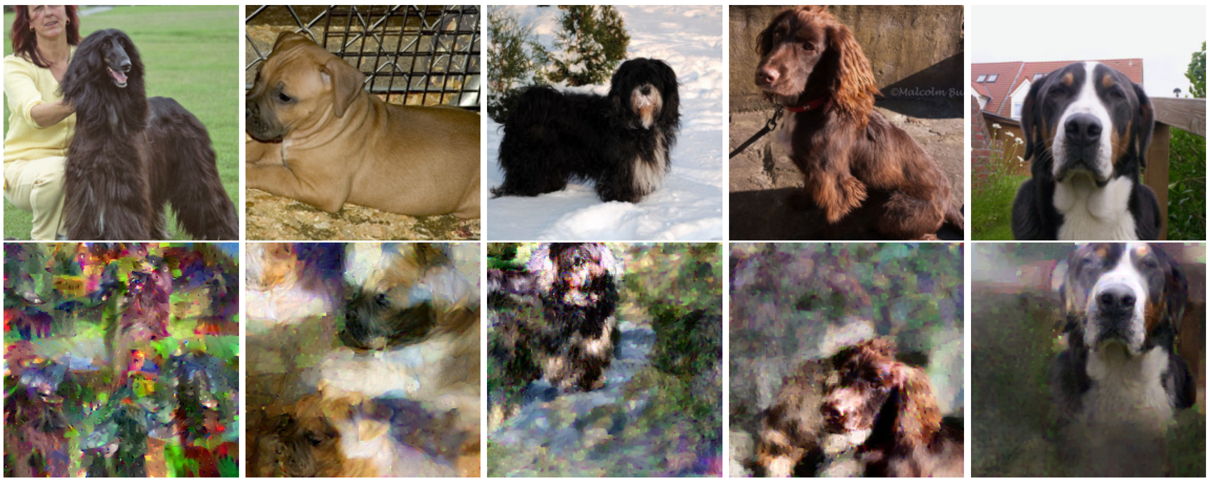


Figure 5.14: Additional single-image reconstruction from the parameter gradients of trained ResNet-152. Top row: Ground Truth. Bottom row: Reconstruction. The paper showed images 0000, 1000, 2000, 3000, 4000, 5000, 6000, 7000 from the ILSVRC2012 validation set. These are images 8000-12000.

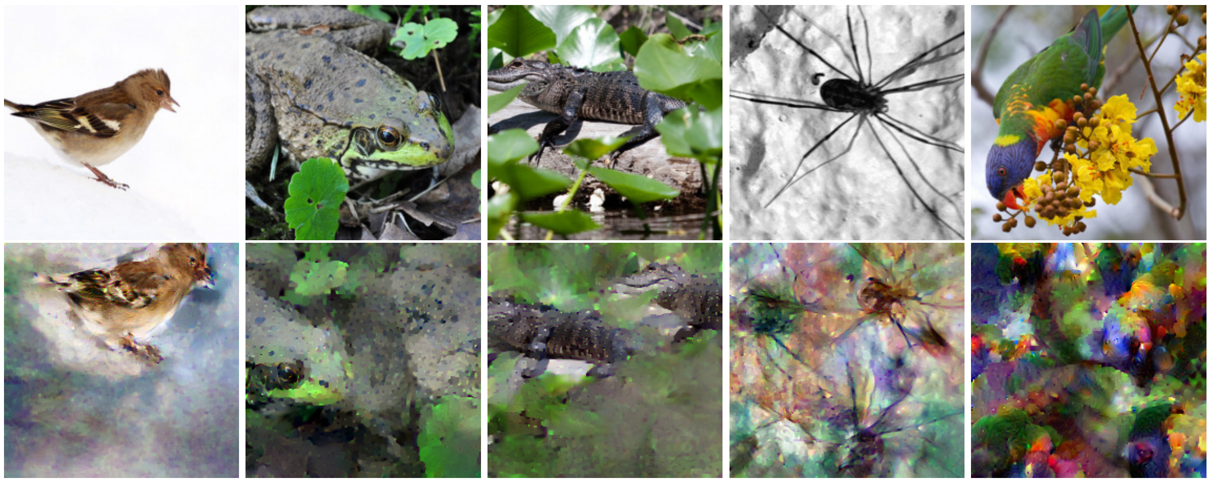


Figure 5.15: Additional single-image reconstruction from the parameter gradients of trained ResNet-152. Top row: Ground Truth. Bottom row: Reconstruction. These are images 500, 1500, 2500, 3500, 4500.

Additional images are following on the next pages.

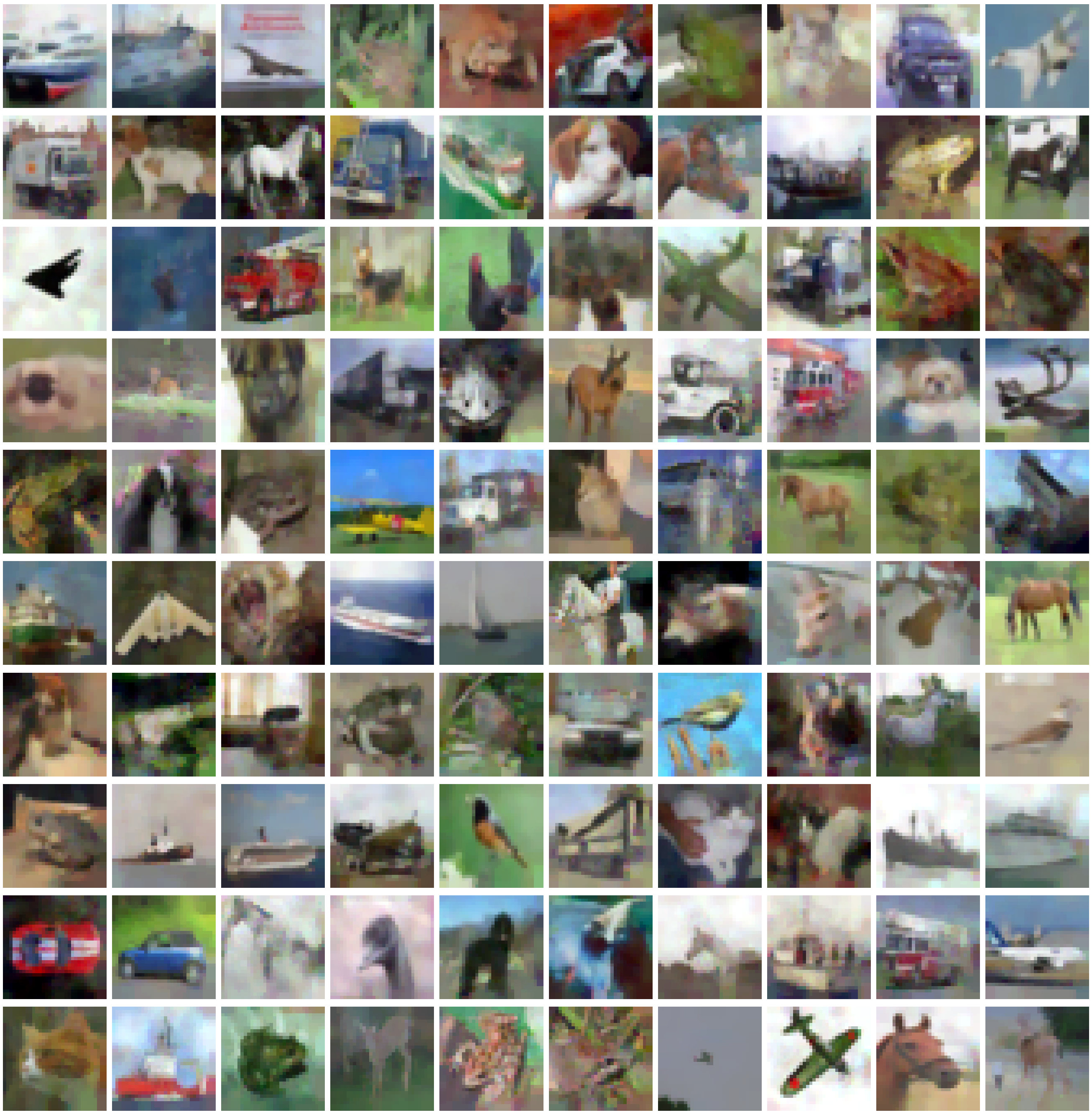


Figure 5.16: Results of the first 100 experiments for $E = 5$, $n = 1$, $B = 1$.

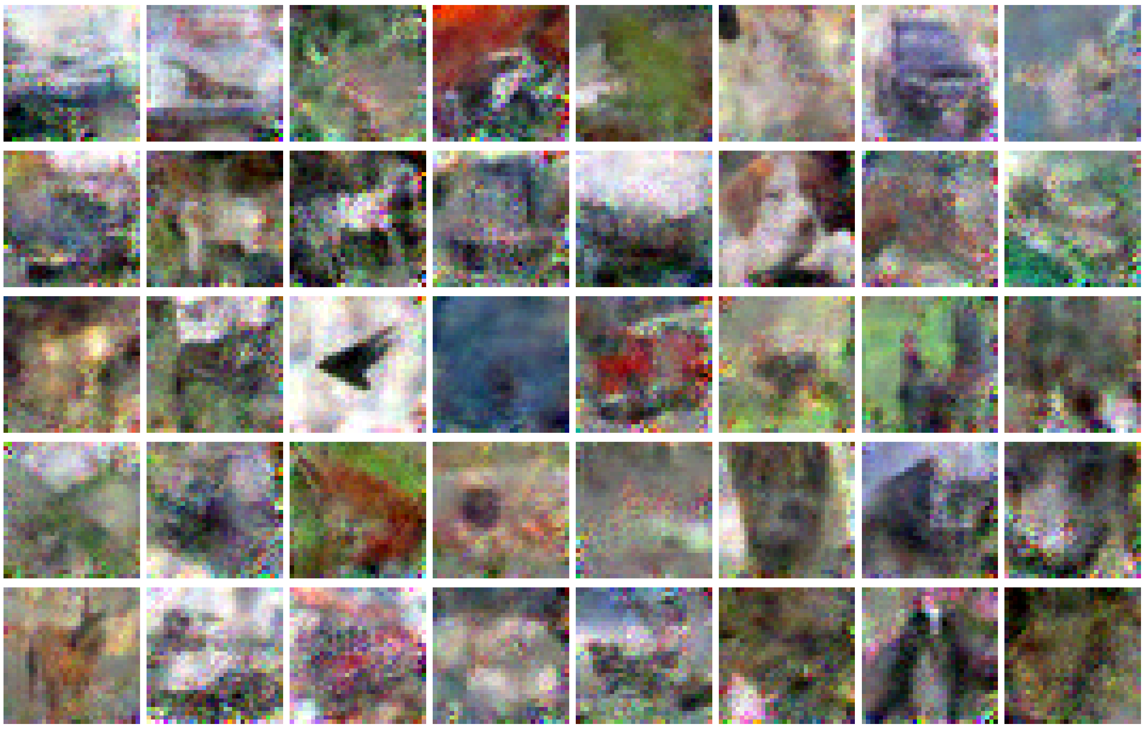


Figure 5.17: Results of the first ten experiments for $E = 1$, $n = 4$, $B = 2$.

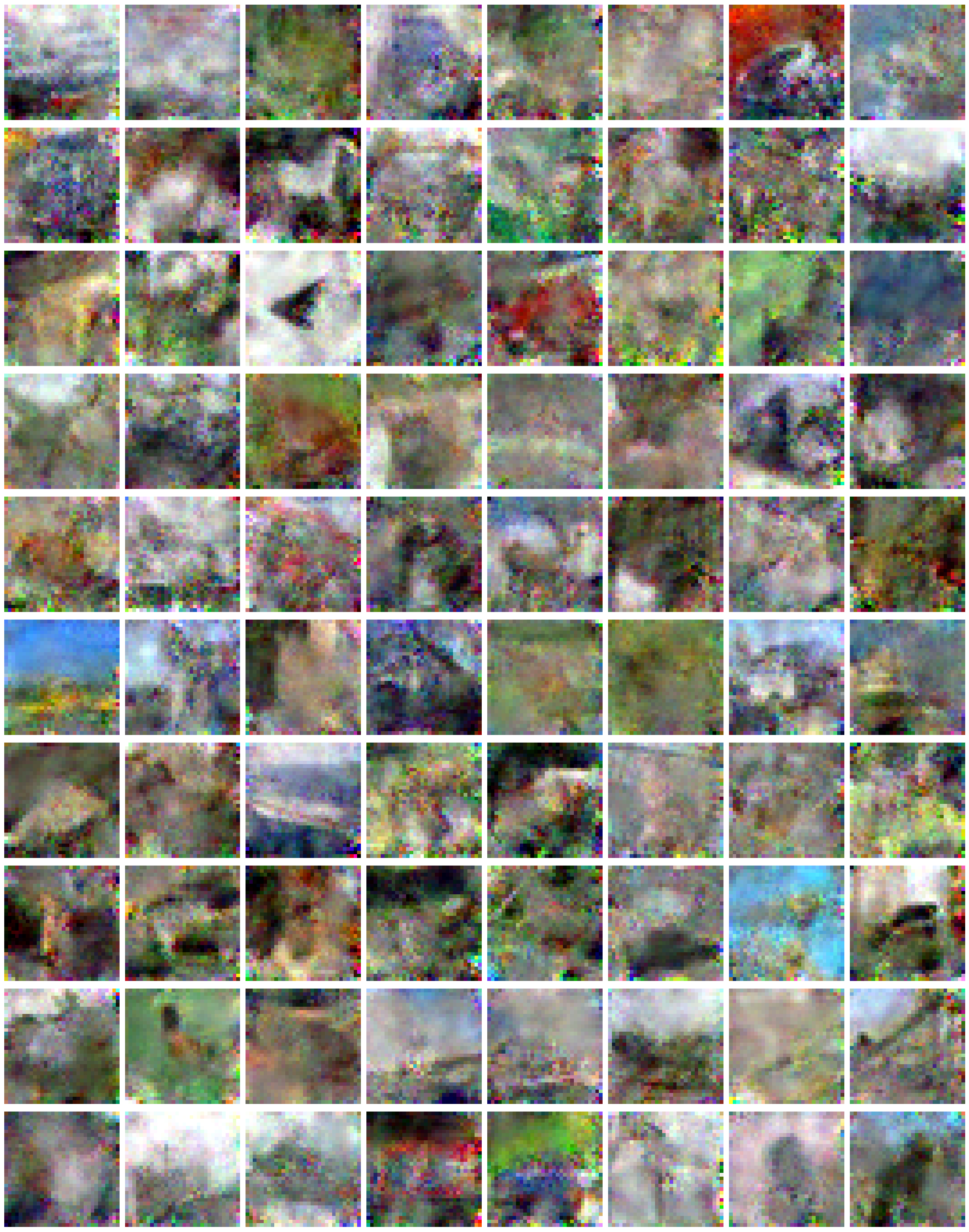


Figure 5.18: Results of the first ten experiments for $E = 1$, $n = 8$, $B = 2$.

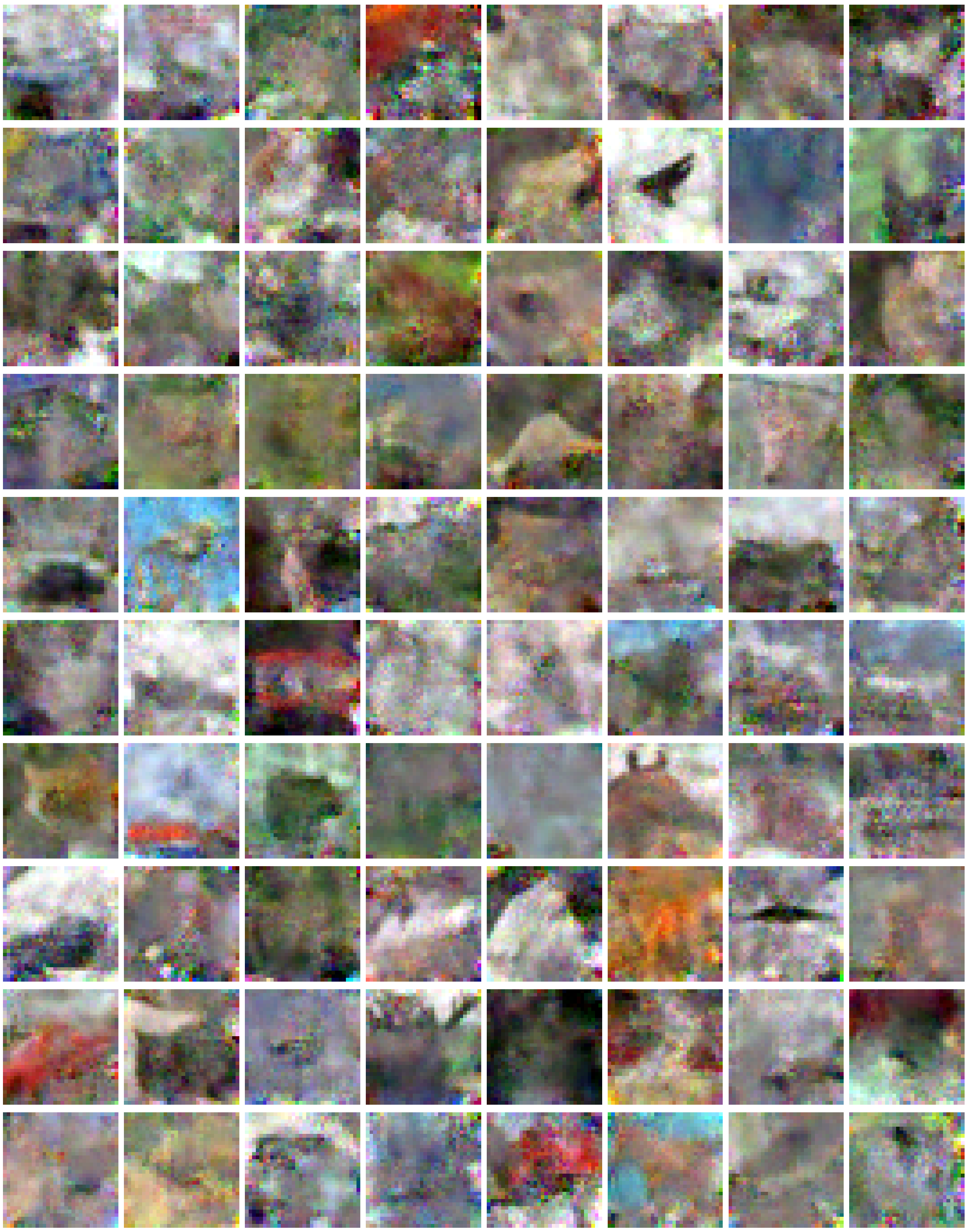


Figure 5.19: Results of the first ten experiments for $E = 1$, $n = 8$, $B = 8$.

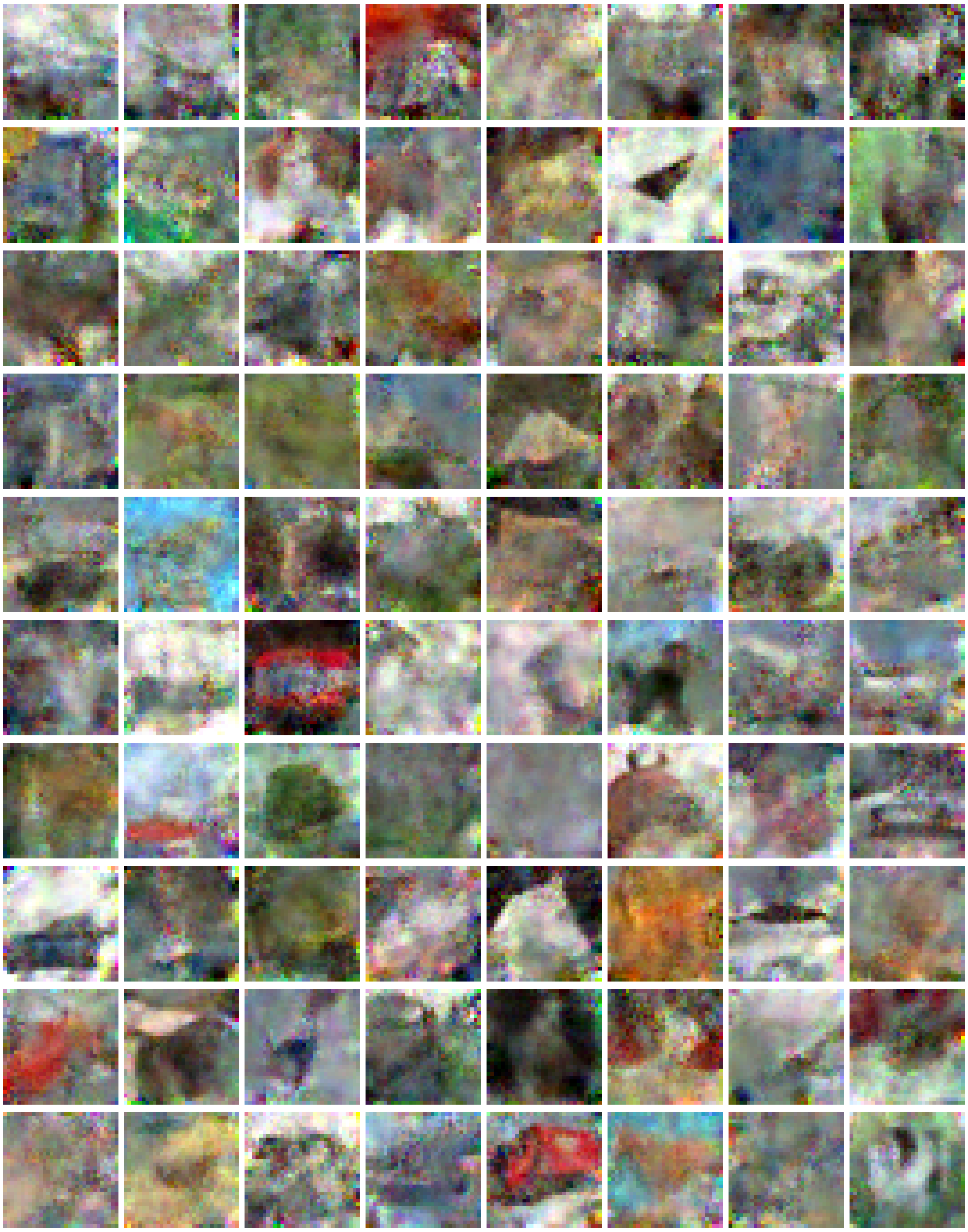


Figure 5.20: Results of the first ten experiments for $E = 5$, $n = 8$, $B = 8$.

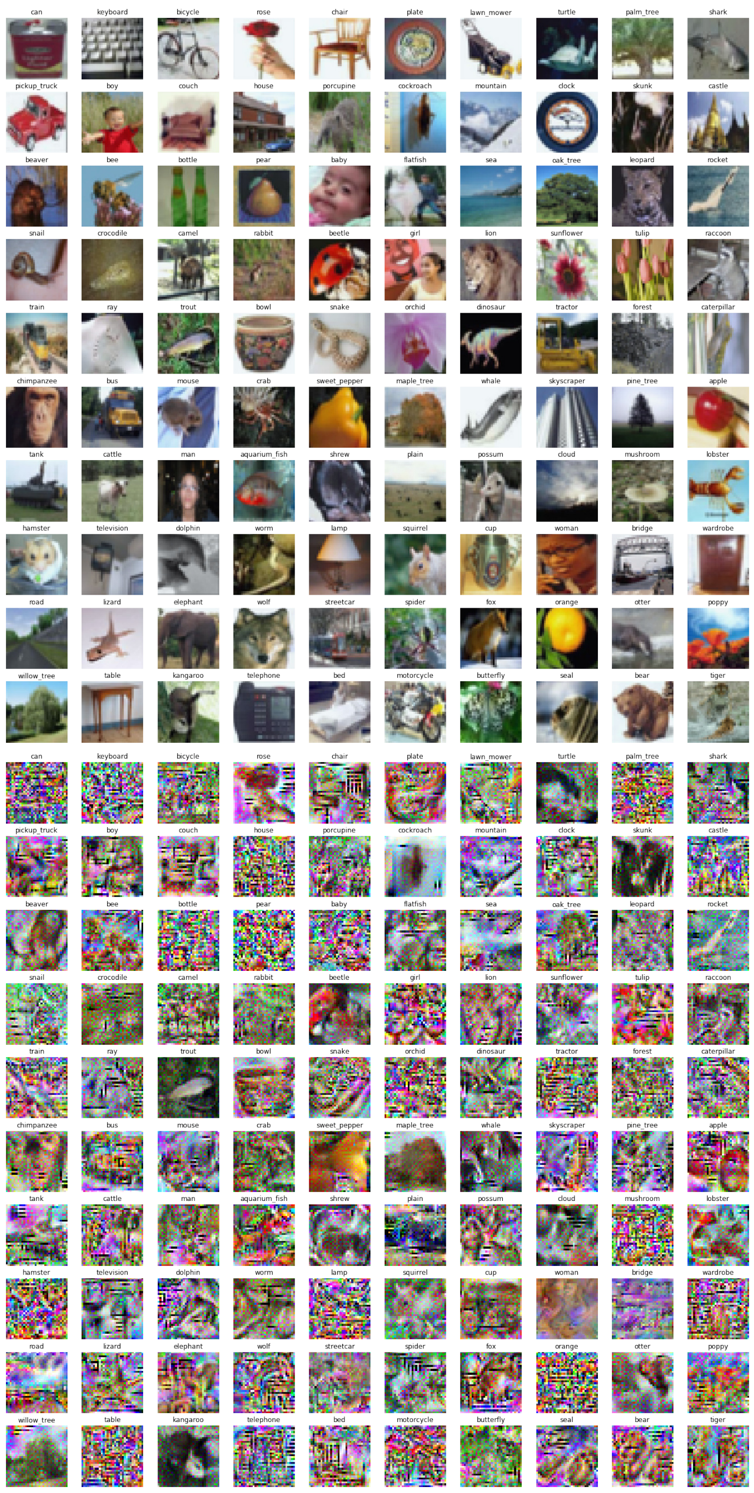


Figure 5.21: Full results for the batch of CIFAR-100 images. Same experiment as in Figure 5.7 of the paper.

CHAPTER 6

Witches' Brew: Industrial Scale Data Poisoning via Gradient Matching

Contextualization

This chapter reprints the publication "Witches' Brew: Industrial Scale Data Poisoning via Gradient Matching", accepted for publication at the International Conference on Learning Representations 2021 (ICLR 2021) in with co-authors Liam Fowl, W. Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller and Tom Goldstein (Geiping et al. [2021][102]).

This work is a second example of an application of optimization techniques in the field of machine learning security. Here, the focus lies on data poisoning: The training data of a machine learning is modified imperceptibly to encode a "backdoor" into a model later trained on this modified data. In this case the backdoor is created to allow for the arbitrary re-classification of specific target images, while keeping the model performance otherwise inconspicuous. Threat models like this can again be formalized as a bilevel objective, as described in (1.7), which can be represented in the notation of this chapter as

$$\min_{x_p \in \mathcal{C}} \mathcal{L}_{\text{adv}}\left(x_t, \theta(x_p)\right) \qquad \text{s.t. } \theta(x_p) = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}_{\text{train}}(x_p^i, y_p^i, \theta).$$

Here, an arbitrary malicious goal is encoded via the upper-level objective $\mathcal{L}_{\text{adv}}$, dependent on the target images x_t. However, this goal can only be realized by altering the parameters θ of the model, which themselves depend on N data points x_p with labels y_p that can be poisoned within constraints dictated by $\mathcal{C}$.

Solving this bilevel problem is especially challenging, if the machine learning model that is attacked is a modern convolutional neural network. After previous work by the authors, where unrolling strategies were used to optimize the objective (Huang et al. [2020]), here we consider a surrogate objective based on gradient matching. Interestingly, the results of the previous chapter can be seen as a special case sanity check of this attack: If only a single image x_p is poisoned and only a single target x_t is considered and $\mathcal{C}$ describes arbitrary changes to x_p, then the attack proposed here reduces to a gradient recovery problem, and as seen in the last chapter, indeed recovers the optimal solution $x_p = x_t$. In practice however, the attack is required to be inconspicuous, and as such $\mathcal{C}$ is a major constraining factor - which is mitigated by increasing the number of poisoned images x_p, each of which is modified in a minor way.

This strategy was proposed jointly by Liam Fowl and Jonas Geiping. Jonas Geiping implemented the attack and related poisoning attacks and proposed the modifications in subsection 6.3.4, that allow for the attack to succeed especially in practical scenarios. Liam Fowl and Michael Moeller derived Proposition 6.1 on adversarial descent in discussion with Jonas Geiping and Wojciech Czaja. Tom Goldstein and Gavin Taylor supported and conducted large-scale experiments with this method on the ImageNet dataset and comparisons to previous work. Jonas Geiping conducted the main experimental evaluation on CIFAR-10 and defenses via differential privacy, Liam Fowl conducted the main experimental evaluation on ImageNet and defenses via feature collision. W. Ronny Huang contributed implementations, domain knowledge and support for the black-box attacks against Google's Cloud AutoML model, as well as insight and considerations into previous work such as Huang et al. [2020] and related work. Tom Goldstein and Michael Moeller supervised the project and with Wojciech Czaja contributed to the writing and style of this work, which was drafted by Jonas Geiping and Liam Fowl. Liam Fowl further contributed the illustrations in Figure 6.1, details on filtering defenses, ablation studies with reduced data and transfer as well as multi-target experiments. Jonas Geiping further contributed the gradient alignment measurements, experimental setup, conducted the AutoML experiments, full-scale MetaPoison comparison, ablation studies and benchmark results on the poisoning benchmark of [255]. A framework of code for evaluation of this attack is publicly available at https://github.com/JonasGeiping/poisoning-gradient-matching.

Abstract

Data Poisoning attacks modify training data to maliciously control a model trained on such data. Previous poisoning attacks against deep neural networks have been limited in scope and success, working only in simplified settings or being prohibitively expensive for large datasets. In this work, we focus on a particularly malicious poisoning attack that is both "from scratch" and "clean label", meaning we analyze an attack that successfully works against new, randomly initialized models, and is nearly imperceptible to humans, all while perturbing only a small fraction of the training data. The central mechanism of this attack is matching the gradient direction of malicious examples. We analyze why this works, supplement with practical considerations. and show its threat to real-world practitioners, finding that it is the first poisoning method to cause targeted misclassification in modern deep networks trained from scratch on a full-sized, poisoned ImageNet dataset. Finally we demonstrate the limitations of existing defensive strategies against such an attack, concluding that data poisoning is a credible threat, even for large-scale deep learning systems.

6.1 Introduction

Machine learning models have quickly become the backbone of many applications from photo processing on mobile devices and ad placement to security and surveillance [166]. These applications often rely on large training datasets that aggregate samples of unknown origins, and the security implications of this are not yet fully understood [217]. Data is often sourced in a way that lets malicious outsiders contribute to the dataset, such as scraping images from the web, farming data from website users, or using large academic datasets scraped from social media [277]. *Data Poisoning* is a security threat in which an attacker makes imperceptible changes to data that can then be disseminated through social media, user devices, or public datasets without being caught by human supervision. The goal of a poisoning attack is to modify the final model to achieve a malicious goal. Poisoning research has focuses on attacks that achieve mis-classification of some predetermined target data in [272, 256], i.e. implementing a backdoor - but other potential goals of the attacker include denial-of-service [268, 259], concealment of users [258], and introduction of fingerprint information [182]. These attacks are applied in scenarios such as social recommendation [124], content management [172, 93], algorithmic fairness [265] and biometric recognition [179]. Accordingly, industry practitioners ranked data poisoning as the most serious attack on ML systems in a recent survey of corporations [156].

In this work we show that efficient poisoned data can be created even in the setting of deep neural networks trained on large image classification tasks, such as ImageNet [246]. Previous work on data

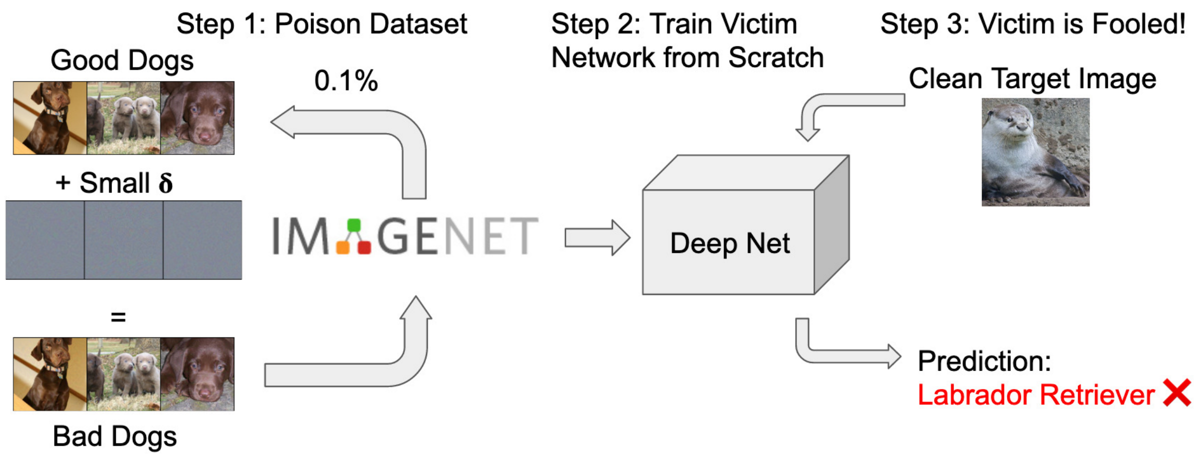


Figure 6.1: The poisoning pipeline. Poisoned images (labrador retriever class) are inserted into a dataset and cause a newly trained victim model to mis-classify a target (otter) image. We show successful poisons for a threat model where 0.1% of training data is changed within an ℓ_∞ bound of $\varepsilon = 8$. Further visualizations of poisoned data can be found in the appendix.

poisoning has often focused on either linear classification tasks [31, 299, 149] or poisoning of transfer learning and fine tuning [256, 148] rather than a full end-to-end training pipeline. Poison attacks on deep neural networks (and especially on ones trained from scratch) have proven difficult in [199] and [256]. Only recently were attacks against neural networks retrained from scratch shown to be possible in [126] for CIFAR-10 - however with costs that render scaling to larger datasets, like the ImageNet, prohibitively expensive.

We formulate data poisoning as the problem of solving a *gradient matching* problem and analyze the resulting novel attack algorithm that scales to unprecedented dataset size and effectiveness. Crucially, the new poisoning objective is orders-of-magnitude more efficient than a previous formulation based on on meta learning [126] and succeeds more often. We conduct an experimental evaluation, showing that poisoned datasets created by this method are robust and significantly outperform other attacks on CIFAR-10. We then demonstrate reliably successful attacks on common ImageNet models in realistic training scenarios. For example, the attack successfully compromises a ResNet-34 by manipulating only 0.1% of the data points with perturbations less than 8 pixel values in ℓ_∞-norm. We close by discussing previous defense strategies and how strong differential privacy [1] is the only existing defense that can partially mitigate the effects of the attack.

6.2 Related Work

The task of data poisoning is closely related to the problem of adversarial attacks at test time, also referred to as evasion attacks [275, 184], where the attacker alters a target test image to fool an already-trained model. This attack is applicable in scenarios where the attacker has control over the target image, but not over the training data. An intermediary between data poisoning and adversarial attacks are backdoor trigger attacks [288, 248]. These attacks involve inserting a trigger – often an image patch – into training data, which is later activated by also applying the trigger to test images. Backdoor attacks require perturbations to both training and test-time data – a more permissive threat model than either poisoning or evasion.

In contrast to evasion and backdoor attacks, data poisoning attacks consider a setting where the attacker can modify training data, but does not have access to test data. Within this setting we focus on *targeted attacks* – attacks that aim to cause a specific target test image (or set of target test images) to be mis-classified. For example, an attack may cause a certain target image of a otter to be classified as a dog by victim models at test time. This attack is difficult to detect, because it does not noticeably degrade either training or validation accuracy [256, 126].

Two basic schemes for targeted poisoning are label flipping [17, 221], and watermarking [272, 256]. In label flipping attacks, an attacker is allowed to change the label of examples, whereas in a watermarking attack, the attacker perturbs the training image, not label, by superimposing a target image onto training images. These attacks can be successful, yet they are easily detected by supervision such as [218]. This is in contrast to *clean-label* attacks which maintain the semantic labels of data.

Mathematically speaking, data poisoning is a *bilevel* optimization problem [15, 31]; the attacker optimizes image pixels to enforce (malicious) criteria on the resulting network parameters, which are themselves the solution to an "inner" optimization problem that minimizes the training objective. Direct solutions to the bilevel problem have been proposed where feasible, for example, SVMs in [31] or logistic regression in [81]. However, direct optimization of the poisoning objective is intractable for deep neural networks because it requires backpropagating through the entire SGD training procedure, see [199]. As such, the bilevel objective has to be approximated. Recently, MetaPoison [126] proposed to approximately solve the bi-level problem based on methods from the meta-learning community [96]. The bilevel gradient is approximated by backpropagation through several unrolled gradient descent steps. This is the first attack to succeed against deep networks on CIFAR-10 as well as providing transferability to other models. Yet, [126] uses a complex loss function averaged over a wide range of models trained to different epochs and a single unrolling step necessarily involves both clean and poisoned data, making it roughly as costly as one epoch of standard training. With an ensemble of 24 models, [126] requires 3 (2 unrolling steps + 1 clean update step) x 2 (backpropagation through unrolled steps) x 60 (first-order optimization steps) x 24 (ensemble of models) equivalent epochs of normal training to attack, as well as ($\sum_{k=0}^{23} k = 253$) epochs of pretraining. All in all, this equates to 8893 training epochs.

In contrast to bilevel approaches stand heuristics for data poisoning of neural networks. The most prominent heuristic is *feature collision*, as in Poison Frogs [256], which seeks to cause a target test image to be misclassified by perturbing training data to collide with the target image in feature space. Modifications surround the target image in feature space with a convex polytope [313] or collection of poisons [3]. These methods are efficient, but designed to attack fine-tuning scenarios where the feature extractor is nearly fixed. When applied to deep networks trained from scratch, their performance drops significantly.

6.3 Efficient Poison Brewing

In this section, we will discuss an intriguing weakness of neural network training based on first-order optimization and derive an attack against it. This attack modifies training images that so they produce a *malicious gradient signal* during training, even while appearing inconspicuous. This is done by matching the gradient of the target images within ℓ^∞ bounds. Because neural networks are trained by gradient descent, even minor modifications of the gradients can be incorporated into the final model.

This attack compounds the strengths of previous schemes, allowing for data poisoning as efficiently as in Poison Frogs [256], requiring only a single pretrained model and a time budget on the order of one epoch of training for optimization - but still capable of poisoning the from-scratch setting considered in [126]. This combination allow an attacker to "brew" poisons that successfully attack realistic models on ImageNet.

6.3.1 Threat Model

We define two parties, the *attacker*, which has limited control over the training data, and the *victim*, which trains a model based on this data. We first consider a gray-box setting, where the attacker has knowledge of the model architecture used by its victim. The attacker is permitted to poison a fraction of the training dataset (usually less than 1%) by changing images within an ℓ_∞-norm ε-bound (e.g. with $\varepsilon \leq 16$). This constraint enforces *clean-label* attacks, meaning that the semantic label of a poisoned image is still unchanged. The attacker has no knowledge of the training procedure - neither about the initialization of the victim's model, nor about the (randomized) mini-batching and data augmentation that is standard in the training of deep learning models.

We formalize this threat model as bilevel problem for a machine learning model $F(x, \theta)$ with inputs $x \in \mathbb{R}^n$ and parameters $\theta \in \mathbb{R}^p$, and loss function $\mathcal{L}$. We denote the N training samples by $(x_i, y_i)_{i=1}^N$, from which a subset of P samples are poisoned. For notation simplicity we assume the first P training images are poisoned by adding a perturbation Δ_i to the i^{th} training image. The perturbation is constrained to be smaller than ε in the ℓ_∞-norm. The task is to optimize Δ so that a set of T target samples $(x_i^t, y_i^t)_{i=1}^T$ is reclassified with the new adversarial labels y_i^{adv}:

$$\min_{\Delta \in \mathcal{C}} \sum_{i=1}^T \mathcal{L}\left(F(x_i^t, \theta(\Delta)), y_i^{\text{adv}}\right) \quad \text{s.t. } \theta(\Delta) \in \arg\min_\theta \frac{1}{N} \sum_{i=1}^N \mathcal{L}(F(x_i + \Delta_i, \theta), y_i). \tag{6.1}$$

We subsume the constraints in the set $\mathcal{C} = \{\Delta \in \mathbb{R}^{N \times n} : ||\Delta||_\infty \leq \varepsilon, \Delta_i = 0 \, \forall i > P\}$. We call the main objective on the left the *adversarial loss*, and the objective that appears in the constraint on the right is the *training loss*. For the remainder, we consider a single target image ($T = 1$) as in [256], but stress that this is not a general limitation as shown in the appendix.

6.3.2 Motivation

What is the optimal alteration of the training set that causes a victim neural network $F(x, \theta)$ to mis-classify a specific target image x^t? We know that the expressivity of deep networks allows them to fit arbitrary training data [306]. Thus, if an attacker was unconstrained, a straightforward way to cause targeted mis-classification of an image is to insert the target image, with the incorrect label y^{adv}, into the victim network's training set. Then, when the victim minimizes the training loss they simultaneously minimize the adversarial loss, based on the gradient information about the target image. In our threat model however, the attacker is not able to insert the mis-labeled target. They can, however, still mimic the gradient of the target by creating poisoned data whose training gradient correlates with the adversarial target gradient. If the attacker can enforce

$$\nabla_\theta \mathcal{L}(F(x^t, \theta), y^{\text{adv}}) \approx \frac{1}{P} \sum_{i=1}^P \nabla_\theta \mathcal{L}(F(x_i + \Delta_i, \theta), y_i) \tag{6.2}$$

to hold for any θ encountered during training, then the victim's gradient steps that minimize the training loss on the poisoned data (right hand side) will also minimize the attackers adversarial loss on the targeted data (left side).

6.3.3 The Central Mechanism: Gradient Alignment

Gradient magnitudes vary dramatically across different stages of training, and so finding poisoned images that satisfy (6.2) for all θ encountered during training is infeasible. Instead we *align* the target and poison gradients in the same direction, that is we minimize their negative cosine similarity. We do this by taking a clean model F with parameters θ, keeping θ fixed, and then optimizing

$$\mathcal{B}(\Delta, \theta) = 1 - \frac{\left\langle \nabla_\theta \mathcal{L}(F(x^t, \theta), y^{\text{adv}}), \sum_{i=1}^P \nabla_\theta \mathcal{L}(F(x_i + \Delta_i, \theta), y_i) \right\rangle}{\|\nabla_\theta \mathcal{L}(F(x^t, \theta), y^{\text{adv}})\| \cdot \|\sum_{i=1}^P \nabla_\theta \mathcal{L}(F(x_i + \Delta_i, \theta), y_i)\|}. \tag{6.3}$$

We optimize $\mathcal{B}(\Delta)$ using signed Adam updates with decaying step size, projecting onto $\mathcal{C}$ after every step. This produces an alignment between the averaged poison gradients and the target gradient. In contrast to Poison Frogs, all layers of the network are included (via their parameters) in this objective, not just the last feature layer.

Each optimization step of this attack requires only a *single* differentiation of the parameter gradient w.r.t to its input, instead of differentiating through several unrolled steps as in MetaPoison. Furthermore, as in Poison Frogs we differentiate through a loss that only involves the (small) subset of poisoned data instead of involving the entire dataset, such that the attack is especially fast if the budget is small. Finally, the method is able to create poisons using only a single parameter vector, θ (like Poison Frogs in fine-tuning setting, but not the case for MetaPoison) and does not require updates of this parameter vector after each poison optimization step.

6.3.4 Making attacks that transfer and succeed "in the wild"

A practical and robust attack must be able to poison different random initializations of network parameters and a variety of architectures. To this end, we employ several techniques:
Differentiable Data Augmentation and Resampling: Data augmentation is a standard tool in deep learning, and transferable image perturbations must survive this process. At each step minimizing (6.3), we randomly draw a translation, crop, and possibly a horizontal flip for each poisoned image, then use bilinear interpolation to resample to the original resolution. When updating Δ, we differentiate through this grid sampling operation as in [135]. This creates an attack which is robust to data augmentation and leads to increased transferability.
Restarts: The efficiency we gained in subsection 6.3.3 allows us to incorporate restarts, a common technique in the creation of evasion attacks [232, 198]. We minimize (6.3) several times from random starting perturbations, and select the set of poisons that give us the lowest alignment loss $\mathcal{B}(\Delta)$. This allows us to trade off reliability with computational effort.
Model Ensembles: A known approach to improving transferability is to attack an ensemble of model instances trained from different initializations [177, 313, 126]. However, ensembles are highly expensive, increasing the pre-training cost for only a modest, but stable, increase in performance.

We show the effects of these techniques via CIFAR-10 experiments (see Table 6.1 and subsection 6.5.1). To keep the attack within practical reach, we do not consider ensembles for our experiments on ImageNet data, opting for the cheaper techniques of restarts and data augmentation. A summarizing description of the attack can be found in Algorithm 6.1. Lines 8 and 9 of Algorithm 6.1 are done in a stochastic (mini-batch) setting (which we omitted in Algorithm 6.1 for notation simplicity).

6.4 Theoretical Analysis

Can gradient alignment cause network parameters to converge to a model with low adversarial loss? To simplify presentation, we denote the adversarial loss and normal training loss of (6.1) as $\mathcal{L}_{\text{adv}}(\theta) =: \mathcal{L}(F((x^t, \theta), y^{\text{adv}})$ and $\mathcal{L}(\theta) =: \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(x_i, y_i, \theta)$, respectively. Also, recall that $1 - \mathcal{B}\left(\Delta, \theta^k\right)$, defined in (6.3), measures the cosine similarity between the gradient of the adversarial loss and the gradient of normal training loss. We adapt a classical result of Zoutendijk [208, Thm. 3.2] to shed light on why data poisoning can work even though the victim only performs standard training on a poisoned dataset:

Algorithm 6.1: Poison Brewing via the discussed approach.

```
1:  **Require** Pretrained clean network $\{F(\cdot, \theta)\}$, a training set of images and labels $(x_i, y_i)_{i=1}^N$, a target
    $(x^t, y^{\text{adv}})$, $P < N$ poison budget, perturbation bound $\varepsilon$, restarts $R$, optimization steps $M$
2:  **Begin**
3:  Select $P$ training images with label $y^{\text{adv}}$
4:  **For** $r = 1, \dots, R$ restarts:
5:      Randomly initialize perturbations $\Delta^r \in \mathcal{C}$
6:      **For** $j = 1, \dots, M$ optimization steps:
7:          Apply data augmentation to all poisoned samples $(x_i + \Delta_i^r)_{i=1}^P$
8:          Compute the average costs, $\mathcal{B}(\Delta^r, \theta)$ as in (6.3), over all poisoned samples
9:          Update $\Delta^r$ with a step of signed Adam and project onto $||\Delta^r||_\infty \leq \varepsilon$
10: Choose the optimal $\Delta^*$ as $\Delta^r$ with minimal value in $\mathcal{B}(\Delta^r, \theta)$
11: **Return** Poisoned dataset $(x_i + \Delta_i^*, y_i)_{i=1}^N$
```

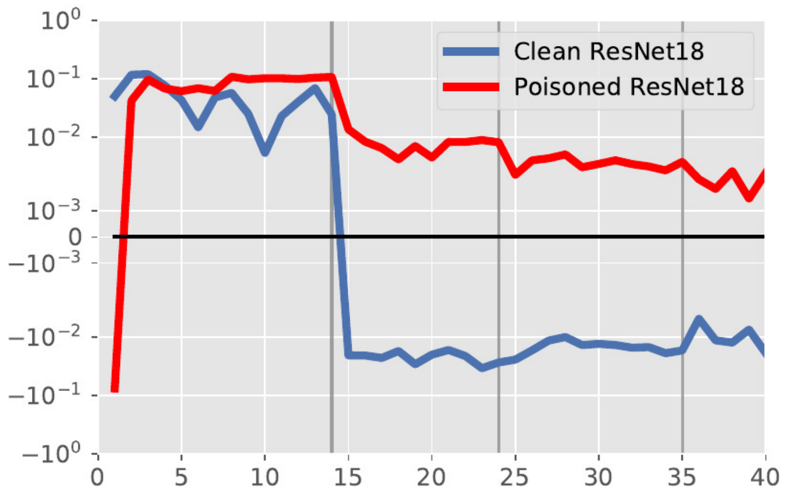


Figure 6.2: Average batch cosine similarity, per epoch, between the adversarial gradient $\nabla\mathcal{L}_{\text{adv}}(\theta)$ and the gradient of each mini-batch $\nabla\mathcal{L}(\theta)$ for a poisoned and a clean ResNet-18. Crucially, the gradient alignment is strictly positive.

Proposition 6.1 (Adversarial Descent). *Let $\mathcal{L}_{\text{adv}}(\theta)$ be bounded below and have a Lipschitz continuous gradient with constant $L > 0$ and assume that the victim model is trained by gradient descent with step sizes α_k, i.e. $\theta^{k+1} = \theta^k - \alpha_k \nabla\mathcal{L}(\theta^k)$. If the gradient descent steps $\alpha_k > 0$ satisfy*

$$\alpha_k L < \beta \left(1 - \mathcal{B}(\Delta, \theta^k)\right) \frac{||\nabla\mathcal{L}(\theta^k)||}{||\nabla\mathcal{L}_{\text{adv}}(\theta^k)||} \tag{6.4}$$

for some fixed $\beta < 1$, then $\mathcal{L}_{adv}(\theta^{k+1}) < \mathcal{L}_{adv}(\theta^k)$. If in addition $\exists \varepsilon > 0,\ k_0$ so that $\forall k \geq k_0$, $\mathcal{B}(\Delta, \theta^k) < 1 - \varepsilon$, then

$$\lim_{k\to\infty} ||\nabla\mathcal{L}_{\text{adv}}(\theta^k)|| \to 0. \tag{6.5}$$

Proof. See supp. material. ∎

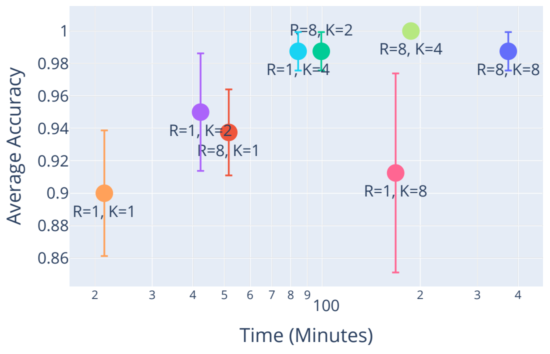
Put simply, our poisoning method aligns the gradients of training loss and adversarial loss. This enforces that the gradient of the main objective is a descent direction for the adversarial objective, which, when combined with conditions on the step sizes, causes a victim to unwittingly converge to a stationary point of the adversarial loss, i.e. optimize *the original bilevel objective* locally.

The strongest assumption in Proposition 6.1 is that gradients are almost always aligned, $\mathcal{B}(\Delta, \theta^k) < 1 - \epsilon, k \geq k_0$. We directly maximize alignment during creation of the poisoned data, but only for a selected θ^*, and not for all θ^k encountered during gradient descent from any possible initialization. However, poison perturbations made from one parameter vector, θ, can transfer to other parameter vectors encountered during training. For example, if one allows larger perturbations, and in the limiting case, unbounded perturbations, our objective is minimal if the poison data is identical to the target image, which aligns training and adversarial gradients at every θ encountered. Empirically, we see that the proposed "poison brewing" attack does indeed increase gradient alignment. In Figure 6.2, we see that in the first phase of training all alignments are positive, but only the poisoned model maintains a positive similarity for the adversarial target-label gradient throughout training. The clean model consistently shows that these angles are negatively aligned - i.e. normal training on a clean dataset will increase adversarial loss. However, after the inclusion of poisoned data, the gradient alignment is modified enough to change the prediction for the target.

6.5 Experimental Evaluation

We evaluate poisoning approaches in each experiment by sampling 10 random poison-target cases. We compute poisons for each and evaluate them on 8 newly initialized victim models (see supp. material Sec. A.1 for details of our methodology). We use the following hyperparameters for all our experiments: $\tau = 0.1$, $R = 8$, $M = 250$. We train victim models in a realistic setting, considering data augmentation, SGD with momentum, weight decay and learning rate drops.

6.5.1 Evaluations on CIFAR-10



As a baseline on CIFAR-10, we compare the number of restarts R and the number of ensembled models K, showing that the proposed method is successful in creating poisons even with just a single model (instead of an ensemble). The inset figure shows poison success versus time necessary to compute the poisoned dataset for a budget of 1%, $\varepsilon = 16$ on CIFAR-10 for a ResNet-18. We find that as the number of ensemble models, K, increases, it is beneficial to increase the number of restarts as well, but increasing the number of restarts independently also improves performance. We validate the differentiable data augmentation discussed in subsection 6.3.4 in Table 6.1, finding it crucial for scalable data poisoning, being as efficient as a large model ensemble in facilitating robustness.

Next, to test different poisoning methods, we fix our "brewing" framework of efficient data poisoning, with only a single network and diff. data augmentation. We evaluate the discussed gradient matching cost function, replacing it with either the feature-collision objective of Poison Frogs or the bullseye objective of [3], thereby effectively replicating their methods, but in our context of from-scratch training.

The results of this comparison are collated in Table 6.2. While Poison Frogs and Bullseye Polytope succeeded in finetuning settings, we find that their feature collision objectives are only successful in the shallower network in the from-scratch setting. Gradient matching further outperforms MetaPoison on CIFAR-10, while faster (see appendix), in particular as $K = 24$ for MetaPoison.

Benchmark results on CIFAR-10: To evaluate our results against a wider range of poison attacks, we

Table 6.1: CIFAR-10 ablation. $\varepsilon = 16$, budget is 1%. Differentiable data augmentation is able to replace a large 8-model ensemble, without increasing computational effort.

Ensemble	**Diff. Data Aug.**	**Victim does data aug.**	**Poison Accuracy** ($\%(\pm$SE))
1	X	X	100.00% (± 0.00)
1	X	✓	32.50% (± 12.27)
8	X	X	78.75% (± 11.77)
1	✓	✓	91.25% (± 6.14)

Table 6.2: CIFAR-10 Comparison to other poisoning objectives with a budget of 1% within our framework (columns 1 to 3), for a 6-layer ConvNet and an 18-layer ResNet. MetaPoison* denotes the full framework of [126]. Each cell shows the avg. poison success and its standard error.

	Proposed	Bullseye	Poison Frogs	MetaPoison*
ConvNet ($\varepsilon = 32$)	*86.25%* (± 9.43)	78.75% (± 7.66)	52.50% (± 12.85)	35.00% (± 11.01)
ResNet-18 ($\varepsilon = 16$)	*90.00%* (± 3.87)	3.75% (± 3.56)	1.25% (± 1.19)	42.50 % (± 8.33)

Table 6.3: Results on the benchmark of [255]. Avg. accuracy of poisoned CIFAR-10 (budget 1%, $\varepsilon = 8$) over 100 trials is shown. (*) denotes rows replicated from [255]. Poisons are created with a ResNet-18 except for the last row, where the ensemble consists of two models of each architecture.

Attack	**ResNet-18**	**MobileNet-V2**	**VGG11**	**Average**
Poison Frogs* [256]	0%	1%	3%	1.33%
Convex Polytopes* [313]	0%	1%	1%	0.67%
Clean-Label Backd.* [288]	0%	1%	2%	1.00%
Hidden-Trigger Backd.* [248]	0%	4%	1%	2.67%
Proposed Attack ($K = 1$)	45%	36%	8%	29.67%
Proposed Attack ($K = 4$)	$\mathbf{55\%}$	37%	7%	33.00%
Proposed Attack ($K = 6$, Heterogeneous)	49%	$\mathbf{38\%}$	$\mathbf{35\%}$	$\mathbf{40.67\%}$

consider the recent benchmark proposed in [255] in Table 6.3. In the category "Training From Scratch", this benchmark evaluates poisoned CIFAR-10 datasets with a budget of 1% and $\varepsilon = 8$ against various model architectures, averaged over 100 fixed scenarios. We find that the discussed gradient matching attack, even for $K = 1$ is significantly more potent in the more difficult benchmark setting. An additional feature of the benchmark is *transferability*. Poisons are created using a ResNet-18 model, but evaluated also on two other architectures. We find that the proposed attack transfers to the similar MobileNet-V2 architecture, but not as well to VGG11. However, we also show that this advantage can be easily circumvented by using an ensemble of different models as in [313]. If we use an ensemble of $K = 6$, consisting of 2 ResNet-18, 2 MobileNet-V2 and 2 VGG11 models (last row), then the same poisoned dataset can compromise all models and generalize across architectures.

6.5.2 Poisoning ImageNet models

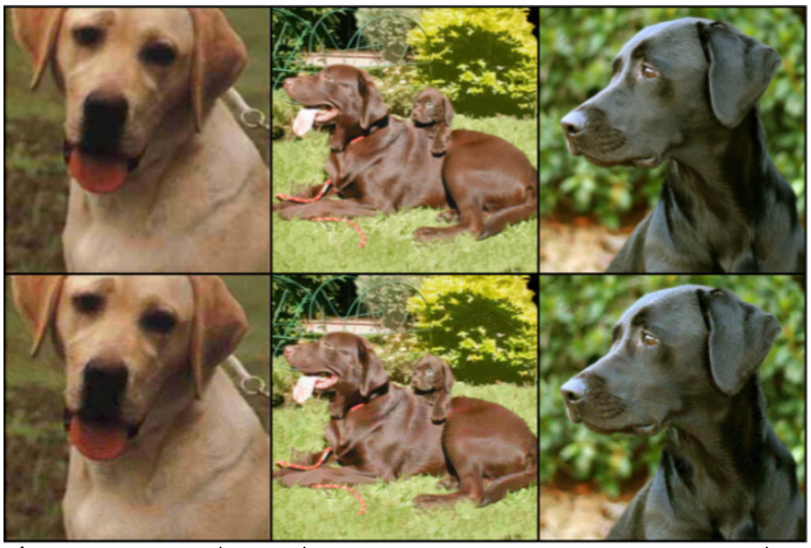
The ILSVRC2012 challenge, "ImageNet", consists of over 1 million training examples, making it infeasible for most actors to train large model ensembles or run extensive hyperparameter optimizations. However, as the new gradient matching attack requires only a single sample of pretrained parameters θ, and operates only on the poisoned subset, it can poison ImageNet images using publicly available pretrained models without ever training an ImageNet classifier. Poisoning ImageNet with previous methods would be infeasible. For example, following the calculations in section 6.2, it would take over 500 GPU days (relative to our hardware) to create a poisoned ImageNet for a ResNet-18 via MetaPoison. In contrast, the new attack can poison ImageNet in less than four GPU hours.

Figure 6.3 shows that a standard ImageNet models trained from scratch on a poisoned dataset "brewed" with the discussed attack, are reliably compromised - with examples of successful poisons shown (top). We first study the effect of varying poison budgets, and ε-bounds (bottom left). Even at a budget of 0.05% and ε-bound of 8, the attack poisons a randomly initialized ResNet-18 80% of the time. These results extend to other popular models, such as MobileNet-v2 and ResNet50 (bottom right).

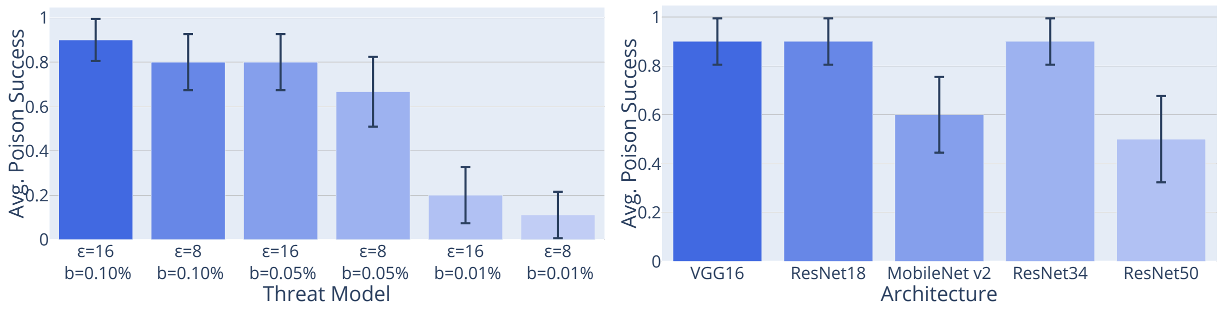
Poisoning Cloud AutoML: To verify that the discussed attack can compromise models in practically relevant *black-box setting*, we test against Google's Cloud AutoML. This is a cloud framework that provides access to black-box ML models based on an uploaded dataset. In [126] Cloud AutoML was shown to be vulnerable for CIFAR-10. We upload a poisoned ImageNet dataset (base: ResNet18, budget 0.1%, $\varepsilon = 32$) for our first poison-target test case and upload the dataset. Even in this scenario, the attack is measurably effective, moving the adversarial label into the top-5 predictions of the model in 5 out of 5 runs, and the top-1 prediction in 1 out of 5 runs.

6.5.3 Deficiencies of Defense Strategies

Previous defenses against data poisoning [268, 220, 223] have relied mainly on data sanitization, i.e. trying to find and remove poisons by outlier detection (often in feature space). We demonstrate why sanitization methods fail in the face of the attack discussed in this work in Figure 6.4a. Poisoned data points are distributed like clean data points, reducing filtering based methods to almost-random guessing (see



a) Clean images (above), with their poisoned counterparts (below) from a successful poisoning of a randomly initialized ResNet-18 trained on ImageNet for a poison budget of 0.1% and an ℓ_∞ bound of $\varepsilon = 8$.



b) Left: ResNet-18 results for different budgets and varying ε-bounds. **Right**: More architectures [261, 117, 250] with a budget of 0.1% and $\varepsilon = 16$.

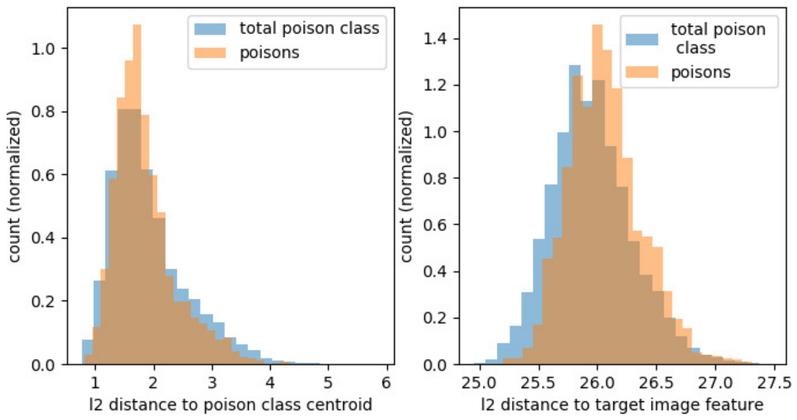
Figure 6.3: Poisoning ImageNet.

supp. material, table 6).

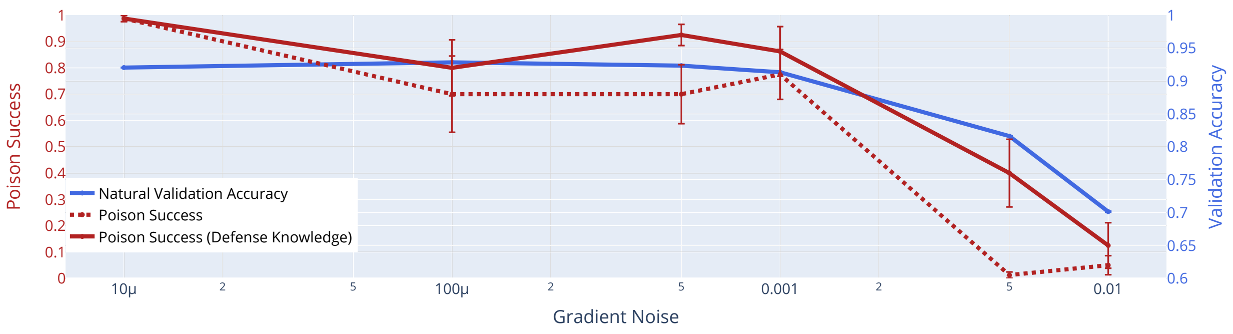
Differentially private training is a different defense. It diminishes the impact of individual training samples, in turn making poisoned data less effective [183, 122]. However, this come at a significant cost. Figure 6.4b shows that to push the Poison Success below 15$\%$, one has to sacrifice over 20% validation accuracy, even on CIFAR-10. Training a diff. private ImageNet model is even more challenging. From this aspect, differentially private training can be compared to adversarial training [184] against evasion attacks. Both methods can mitigate the effectiveness of an adversarial attack, but only by significantly impeding natural accuracy.

6.6 Conclusion

We investigate data poisoning via gradient matching and discover that this mechanism allows for data poisoning attacks against fully retrained models that are unprecedented in scale and effectiveness. We motivate the attack theoretically and empirically, discuss additional mechanisms like differentiable data augmentation and experimentally investigate modern deep neural networks in realistic training scenarios,



a) Feature space distance to base class centroid, and target image feature, for victim model on CIFAR-10. 4.0% budget, $\varepsilon = 16$, showing sanitization defenses failing and no feature collision as in Poison Frogs.



b) Defending through differential privacy. CIFAR-10, 1% budget, $\varepsilon = 16$, ResNet-18. Differential privacy is only able to limit the success of poisoning via trade-off with significant drops in accuracy.

Figure 6.4: Defense strategies against poisoning.

showing that gradient matching attacks compromise even models trained on ImageNet. We close with discussing the limitations of current defense strategies.

The following sections reprint the Witches' Brew appendix.

6.A Remarks

Remark 6.2 (Validating the approach in a special case). Inner-product loss functions like (6.3) work well in other contexts. In [101], cosine similarity between image gradients was minimized to uncover training images used in federated learning. If we disable our constraints, setting $\varepsilon = 255$, and consider a single poison image and a single target, then we minimize the problem of recovering image data from a normalized gradient as a special case. In [101], it was shown that minimizing this problem can recover the target image. This means that we can indeed return to the motivating case in the unconstrained setting - the optimal choice of poison data is insertion of the target image in an unconstrained setting for one image.

Remark 6.3 (Transfer of gradient alignment). An analysis of how gradient alignment often transfers between different parameters and even between architectures has been conducted, e.g. in [60, 148] and [81]. It was shown in [81] that the performance loss when transferring an evasion attack to another model is governed by the gradient alignment of both models. In the same vein, optimizing alignment appears to be a useful metric in the case of data poisoning. Furthermore [122] note that previous poisoning algorithms might already cause gradient alignment as a side effect, even without explicitly optimizing for it.

Remark 6.4 (Poisoning is a Credible Threat to Deep Neural Networks). It is important to understand the security impacts of using unverified data sources for deep network training. Data poisoning attacks up to this point have been limited in scope. Such attacks focus on limited settings such as poisoning SVMs, attacking transfer learning models, or attacking toy architectures [31, 200, 256]. We demonstrate that data poisoning poses a threat to large-scale systems as well. The approach discussed in this work pertains only to the classification scenario, as a guinea pig for data poisoning, but applications to a variety of scenarios of practical interest have been considered in the literature, for example spam detectors mis-classifying a spam email as benign, or poisoning a face unlock based mobile security systems.

The central message of the data poisoning literature can be described as follows: From a security perspective, the data that is used to train a machine learning model should be under the same scrutiny as the model itself. These models can only be secure if the entire data processing pipeline is secure. This issue further cannot easily be solved by human supervision (due to the existence of clean-label attacks) or outlier detection (see Figure 6.4a). Furthermore, targeted poisoning is difficult to detect as validation accuracy is unaffected. As such, data poisoning is best mitigated by fully securing the data pipeline.

So far we have considered data poisoning from the industrial side. From the perspective of a user, or individual under surveillance, however, data poisoning can be a means of securing personal data shared on the internet, making it unusable for automated ML systems. For this setting, we especially refer to an interesting application study in [258] in the context of facial recognition.

6.B Experimental Setup

This appendix section details our experimental setup for replication purposes. A central question in the context of evaluating data poisoning methods is how to judge and evaluate "average" performance. Poisoning is in general volatile with respect to poison-target class pair, and to the specific target example, with some combinations and target images being in general easier to poison than others. However, evaluating all possible combinations is infeasible for all but the simplest datasets, given that poisoned data has to created for each example and then a neural network has to be trained from scratch every time. Previous works [256, 313] have considered select target pairs, e.g. "birds-dogs" and "airplanes-frogs", but this runs the risk of mis-estimating the overall success rates. Another source of variability arises, especially in the from-scratch setting: Due to both the randomness of the initialization of the neural network, the randomness of the order in which images are drawn during mini-batch SGD, and the randomness of data augmentations, a fixed poisoned dataset might only be effective some of the time, when evaluating it multiple times.

In light of this discussion, we adopt the following methodology: For every experiment we randomly select n (usually 10 in our case) settings consisting of a random target class, random poison class, a random target and random images to be poisoned. For each of these experiments we create a single poisoned dataset by the discussed or a comparing method within limits of the given threat model and then evaluate the poisoned datasets m times (8 for CIFAR-10 and 1 for ImageNet) on random re-initializations of the considered architecture. To reduce randomness for a fair comparison between different runs of this setup, we fix the random seeds governing the experiment and rerun different threat models or methods with the same random seeds. We have used CIFAR-10 with random seeds 1000000000-1111111111 hyperparameter tuning and now evaluate on random seeds 2000000000-2111111111 for CIFAR-10 experiments and 1000000000-1111111111 for ImageNet, with class pairs and target image IDs for reproduction given in Tables 6.4 and 6.5. For CIFAR-10, the target ID refers to the canonical order of all images in the dataset (as downloaded from https://www.cs.toronto.edu/~kriz/cifar.html); for ImageNet, the ID refers to an order of ImageNet images where the syn-sets are ordered by their increasing numerical value (as is the default in `torchvision`). However for future research we encourage the sampling of new target-poison pairs to prevent overfitting, ideally even in larger numbers given enough compute power.

For every measurement of *avg. poison success* in the paper, we measure in the following way: After retraining the given deep neural network to completion, we measure if the target image is successfully classified by the network as its adversarial class. We do not count mere misclassification of the original label (but note that this usually happens even before the target is incorrectly classified by the adversarial class). Over the m validation runs we repeat this measurement of target classification success and then compute the average success rate for a single example. We then aggregate this average over our 10 chosen random experiments and report the mean and standard error of these average success rates as *avg. poison success*. All error bars in the paper refer to standard error of these measurements.

Table 6.4: Target/poison class pairs generated from the initial random seeds for ImageNet experiments. Target ID relative to CIFAR-10 validation dataset.

Target Class	Poison Class	Target ID	Random Seed
dog	frog	8745	2000000000
frog	truck	1565	2100000000
frog	bird	2138	2110000000
airplane	dog	5036	2111000000
airplane	ship	1183	2111100000
cat	airplane	7352	2111110000
automobile	frog	3544	2111111000
truck	cat	3676	2111111100
automobile	ship	9882	2111111110
automobile	cat	3028	2111111111

6.B.1 Hardware

We use a heterogeneous mixture of hardware for our experiments. CIFAR-10, and a majority of the ImageNet experiments, were run on NVIDIA GEFORCE RTX 2080 Ti gpus. CIFAR-10 experiments were run on 1 gpu, while ImageNet experiments were run on 4 gpus. We also use NVIDIA Tesla P100 gpus for some ImageNet experiments. All timed experiments were run using 2080 Ti gpus.

6.B.2 Models

For our experiments on CIFAR-10 in section 5 we consider two models. In table 2, the "6-layer ConvNet", - in close association with similar models used in [96] or [155], we consider an architecture of 5 convolutional layers (with kernel size 3), followed by a linear layer. All convolutional layers are followed by a ReLU

Table 6.5: Target/poison class pairs generated from the initial random seeds for ImageNet experiments. Target Id relative to ILSVRC2012 validation dataset [246]

Target Class	Poison Class	Target ID	Random Seed
otter	Labrador retriever	18047	1000000000
warthog	bib	17181	1100000000
orange	radiator	37530	1110000000
theater curtain	maillot	42720	1111000000
hartebeest	capuchin	17580	1111100000
burrito	plunger	48273	1111110000
jackfruit	spider web	47776	1111111000
king snake	hyena	2810	1111111100
flat-coated retriever	alp	10281	1111111110
window screen	hard disc	45236	1111111111

activation. The last two convolutional layers are followed by max pooling with size 3. The output widths of these layers are given by $64, 128, 128, 256, 256, 2304$. In tables 1, 2, in the inset figure and Fig. 4 we consider a ResNet-18 model. We make the customary changes to the model architecture for CIFAR-10, replacing the stem of the original model (which requires ImageNet-sized images) by a convolutional layer of size 3, following by batch normalization and a ReLU. This is effectively equal to upsampling the CIFAR-10 images before feeding them into the model. For experiments on ImageNet, we consider ResNet-18, ResNet-34 [117], MobileNet-v2 [250] and VGG-16 [261] in standard configuration.

We train the ConvNet, MobileNet-v2 and VGG-16 with initial learning rate of 0.01 and the residual architectures with initial learning rate 0.1. We train for 40 epochs, dropping the learning rate by a factor of 10 at epochs 14, 24, 35. We train with stochastic mini-batch gradient descent with Nesterov momentum, with batch size 128 and momentum 0.9. Note that the dataset is shuffled in each epoch, so that where poisoned images appear in mini-batches is random and not known to the attacker. We add weight decay with parameter 5×10^{-4}. For CIFAR-10 we add data augmentations using horizontal flipping with probability 0.5 and random crops of size 32×32 with zero-padding of 4. For ImageNet we resize all images to 256×256 and crop to the central 224×224 pixels. We also consider horizontal flipping with probability 0.5, and data augmentation with random crops of size 224×224 with zero-padding of 28.

When evaluating ImageNet poisoning from-scratch we use the described procedure. To create our poisoned datasets as detailed in Alg. 1, we download the respective pretrained model from `torchvision`, see https://pytorch.org/docs/stable/torchvision/models.html.

6.B.3 Cloud AutoML Setup

For the experiment using Google's cloud autoML, we upload a poisoned ILSVRC2012 dataset into google storage, and then use https://cloud.google.com/vision/automl/ to train a classification model. Due to autoML limitations to 1 million images, we only upload up to 950 examples from each class (reaching a training set size slightly smaller than $950\,000$, which allows for an upload of the $50\,000$ validation images). We use a ResNet-18 model as surrogate for the black-box learning within autoML, pretrained on the full ILSVRC2012 as before. We create a `MULTICLASS` autoML dataset and specify the vision model to be `mobile-high-accuracy-1` which we train to $10\,000$ milli-node hours, five times. After training the model, we evaluate its performance on the validation set and target image. The trained models all reach a 69% clean top-1 accuracy on the ILSVRC2012 validation set.

6.C Proof of Proposition 6.1

Proof of Prop. 6.1. Consider the gradient descent update

$$\theta^{k+1} = \theta^k - \alpha_k \nabla \mathcal{L}(\theta^k)$$

Firstly, due to Lipschitz smoothness of the gradient of the adversarial loss $\mathcal{L}_{\text{adv}}$ we can estimate the value at θ^{k+1} by the descent lemma

$$\mathcal{L}_{\text{adv}}(\theta^{k+1}) \leq \mathcal{L}_{\text{adv}}(\theta^k) - \langle \alpha_k \nabla \mathcal{L}_{\text{adv}}(\theta^k), \nabla \mathcal{L}(\theta^k) \rangle + \alpha_k^2 L ||\nabla \mathcal{L}(\theta^k)||^2$$

If we further use the cosine identity:

$$\langle \nabla \mathcal{L}_{\text{adv}}(\theta^k), \nabla \mathcal{L}(\theta^k) \rangle = ||\nabla \mathcal{L}(\theta^k)|| ||\nabla \mathcal{L}_{\text{adv}}(\theta^k)|| \cos(\gamma^k),$$

denoting the angle between both vectors by γ^k, we find that

$$\begin{aligned} \mathcal{L}_{\text{adv}}(\theta^{k+1}) &\leq \mathcal{L}_{\text{adv}}(\theta^k) - ||\nabla \mathcal{L}(\theta^k)|| ||\nabla \mathcal{L}_{\text{adv}}(\theta^k)|| \cos(\gamma^k) + \alpha_k^2 L ||\nabla \mathcal{L}(\theta^k)||^2 \\ &= \mathcal{L}_{\text{adv}}(\theta^k) - \left(\alpha_k \frac{||\nabla \mathcal{L}_{\text{adv}}(\theta^k)||}{||\nabla \mathcal{L}(\theta^k)||} \cos(\gamma^k) - \alpha_k^2 L \right) ||\nabla \mathcal{L}(\theta^k)||^2 \end{aligned}$$

As such, the adversarial loss decreases for nonzero step sizes if

$$\frac{||\nabla \mathcal{L}_{\text{adv}}(\theta^k)||}{||\nabla \mathcal{L}(\theta^k)||} \cos(\gamma^k) > \alpha_k L$$

i.e.

$$\alpha_k L \leq \frac{||\nabla \mathcal{L}_{\text{adv}}(\theta^k)||}{||\nabla \mathcal{L}(\theta^k)||} \frac{\cos(\gamma^k)}{c}$$

for some $1 < c < \infty$. This follows from our assumption on the parameter β in the statement of the proposition. Reinserting this estimate into the descent inequality reveals that

$$\mathcal{L}_{\text{adv}}(\theta^{k+1}) < \mathcal{L}_{\text{adv}}(\theta^k) - ||\nabla \mathcal{L}_{\text{adv}}||^2 \frac{\cos(\gamma^k)}{c' L},$$

for $\frac{1}{c'} = \frac{1}{c} - \frac{1}{c^2}$. Due to monotonicity we may sum over all descent inequalities, yielding

$$\mathcal{L}_{\text{adv}}(\theta^0) - \mathcal{L}_{\text{adv}}(\theta^{k+1}) \geq \frac{1}{c' L} \sum_{j=0}^{k} ||\nabla \mathcal{L}_{\text{adv}}(\theta^j)||^2 \cos(\gamma^j)$$

As $\mathcal{L}_{\text{adv}}$ is bounded below, we may consider the limit of $k \to \infty$ to find

$$\sum_{j=0}^{\infty} ||\nabla \mathcal{L}_{\text{adv}}(\theta^j)||^2 \cos(\gamma^j) < \infty.$$

If for all, except finitely many iterates the angle between adversarial and training gradient is less than $90°$, i.e. $\cos(\gamma^k)$ is bounded below by some fixed $\epsilon > 0$, as assumed, then the convergence to a stationary point follows:

■

$$\lim_{k \to \infty} ||\nabla \mathcal{L}_{\text{adv}}(\theta^k)|| \to 0$$

In Figure 6.5 we visualize measurements of the computed bound from an actual poisoned training. The classical gradient descent converges only if $\alpha_k L < 1$, so we can find an upper bound to this value by 1, even if the actual Lipschitz constant of the neural network training objective is not known to us.

6.D Poisoned Datasets

We provide access to poisoned datasets as part of the supplementary material, allowing for a replication of the attack. To save space however, we provide only the subset of poisoned images and not the full

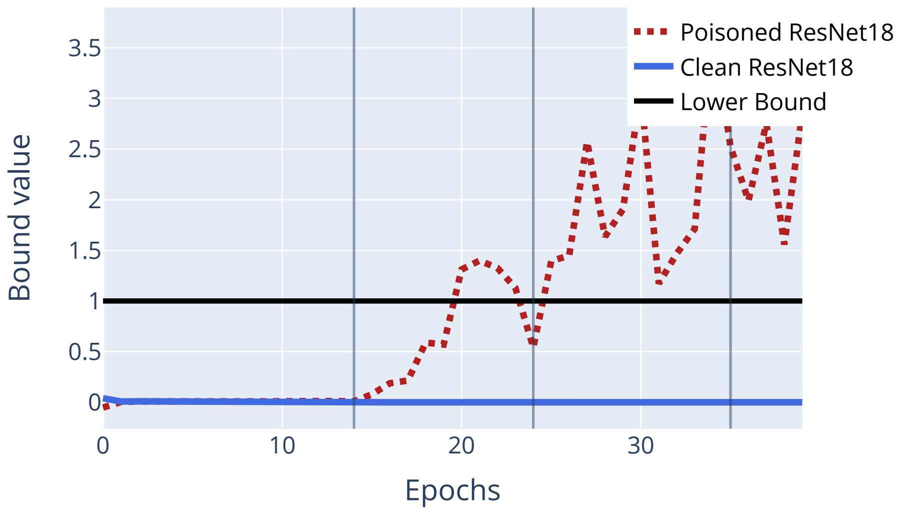


Figure 6.5: The bound considered in Proposition 6.1, evaluated during training of a poisoned and a clean model, using a practical estimation of the lower bound via $\alpha_k L \approx 1$. This is an upper bound of $\alpha_k L$ as $\alpha_k < \frac{1}{L}$ is necessary for the convergence of (clean) gradient descent.

dataset. We hope that this separation also aids in the development of defensive strategies. To train a model using these poisoned data points, you can use our code (using `-save full`) our your own to export either CIFAR-10 or ImageNet into an image folder structure, where the clean images can then be replaced by poisoned images according to their ID. Note that the given IDs refer to the dataset ordering as discussed above.

6.E Visualizations

We visualize poisoned sample from our ImageNet runs in Figures 6.6 and 6.7, noting especially the "clean label" effect. Poisoned data is only barely distinguishable from clean data, even in the given setting where the clean data is shown to the observer. In a realistic setting, this is significantly harder. A subset of poisoned images used to poison Cloud autoML with $\varepsilon = 32$ can be found in Figure 6.8.

We concentrate only on small ℓ^∞ perturbations to the training data as this is the most common setting for adversarial attacks. However, there exist other choices for attacks in practical settings. Previous works have already considered additional color transformations [126] or watermarks [256]. Most techniques that create adversarial attacks at test time within various constraints [92, 307, 118, 142] are likely to transfer into the data poisoning setting. Likewise, we do not consider hiding poisoned images further by minimizing perceptual scores and relate to the large literature of adversarial attacks that evade detection [51].

In Figure 6.9 we visualize how the adversarial loss and accuracy behave during an exemplary training run, comparing the adversarial label with the original label of the target image.

6.F Additional Experiments

This section contains additional experiments.

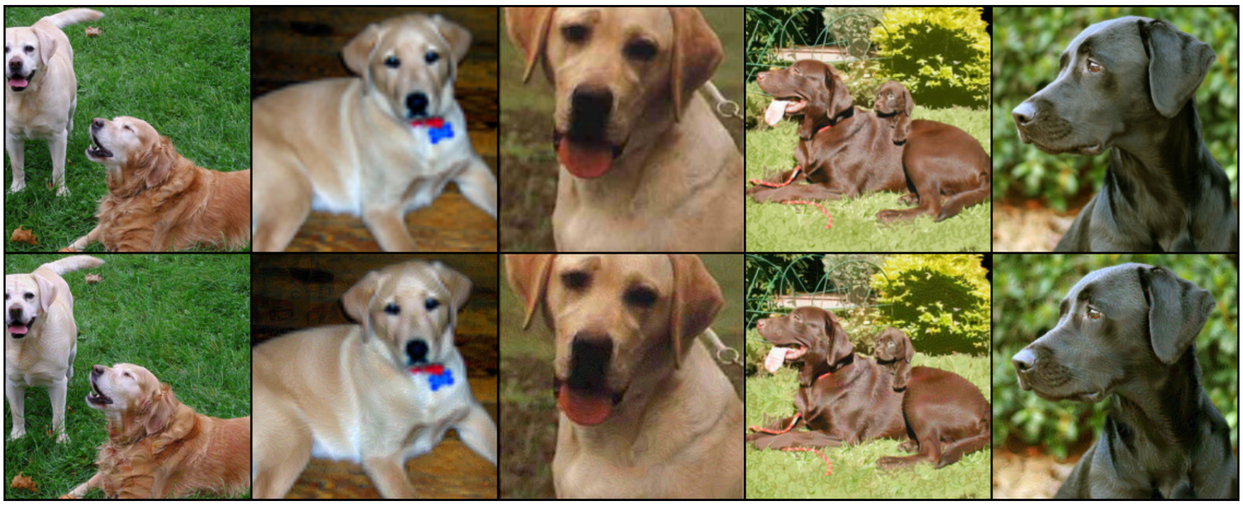


Figure 6.6: Clean images (above), with their poisoned counterparts (below) from a successful poisoning of a Google Cloud AutoML model trained on ImageNet. The poisoned images (taken from the Labrador Retriever class) successfully caused mis-classification of a target (otter) image under a threat model given by a budget 0.1% and an ℓ_∞ bound of $\varepsilon = 32$.

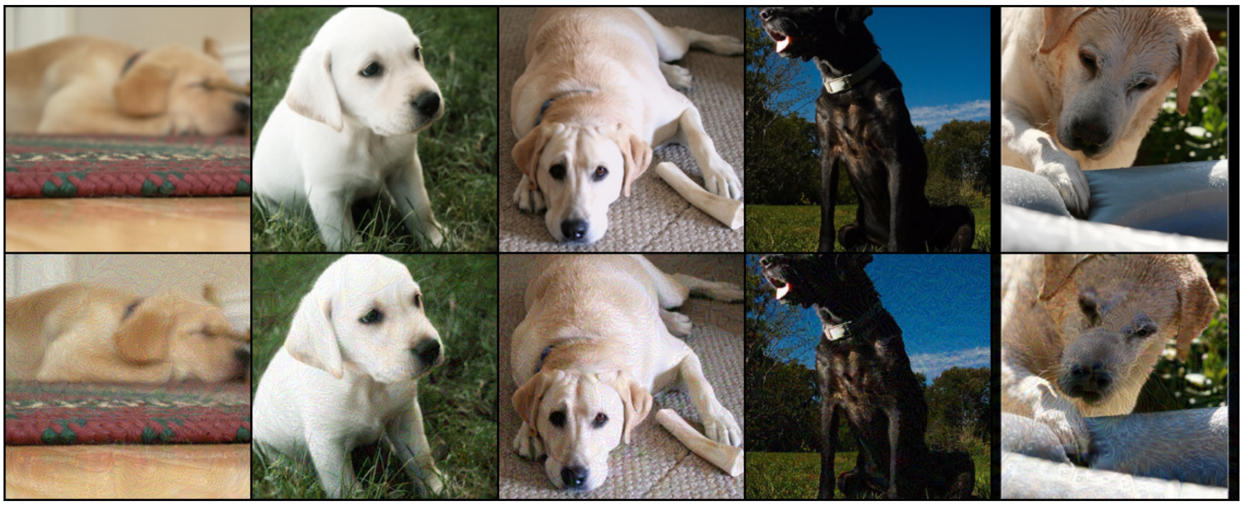


Figure 6.7: Clean images (above), with their poisoned counterparts (below) from a successful poisoning of a randomly initialized ResNet-18 trained on ImageNet. The poisoned images (taken from the Labrador Retriever class) successfully caused mis-classification of a target (otter) image under a threat model given by a budget of 0.1% and an ℓ_∞ bound of $\epsilon = 16$.

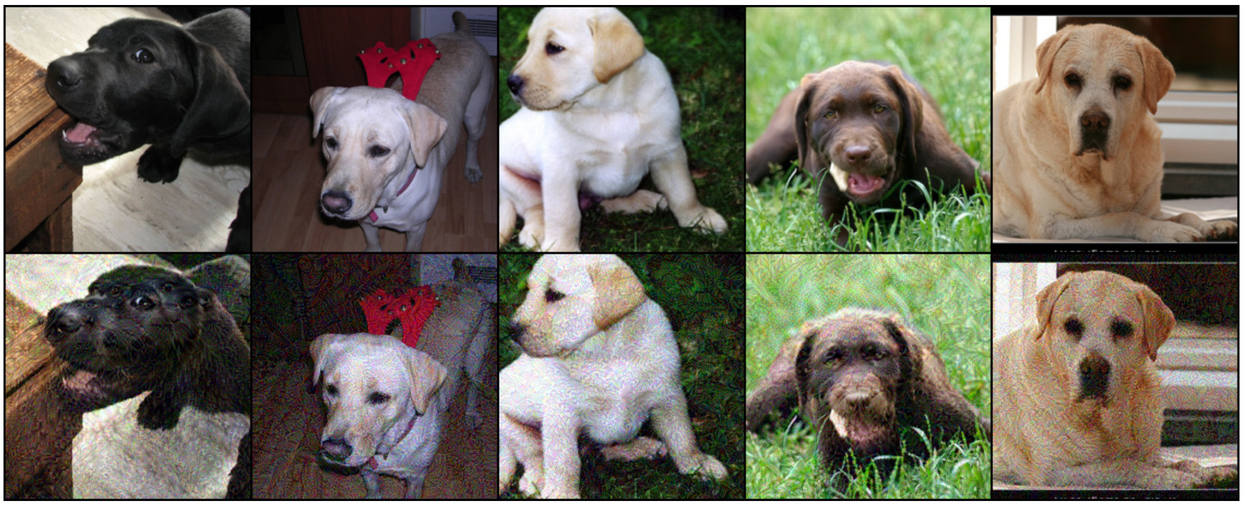


Figure 6.8: Clean images (above), with their poisoned counterparts (below) from a successful poisoning of a Google Cloud AutoML model trained on ImageNet. The poisoned images (taken from the Labrador Retriever class) successfully caused mis-classification of a target (otter) image. This is accomplished with a poison budget of 0.1% and an ℓ_∞ bound of $\varepsilon = 32$.

6.F.1 Full-scale MetaPoison Comparisons on CIFAR-10

Removing all constraints for time and memory, we visualize time/accuracy of our approach against other poisoning approaches in Figure 6.10. Note that attacks, like MetaPoison, which succeed on CIFAR-10 only after removing these constraints, cannot be used on ImageNet-sized datasets due to the significant computational effort required. For MetaPoison, we use the original implementation of [126], but add our larger models. We find that with the larger architectures and different threat model (original MetaPoison considers a color perturbation in addition to the ℓ^∞ bound), our gradient matching technique still significantly outperforms MetaPoison. Note that for the ConvNet experiment on MetaPoison in Table 6.2, we found that MetaPoison seems to overfit with $\varepsilon = 32$, and as such we show numbers running the MetaPoison code with $\varepsilon = 16$ in that column, which are about 8% better than $\varepsilon = 16$. This is possibly a hyperparameter question for MetaPoison, which was optimized for $\varepsilon = 8$ and a color perturbation.

6.F.2 Deficiencies of Filtering Defenses

Defenses aim to sanitize training data of poisons by detecting outliers (often in feature space), and removing or relabeling these points [268, 220, 223]. In some cases, these defenses are in the setting of general performance degrading attacks, while others deal with targeted attacks. By in large, poison defenses up to this point are limited in scope. For example, many defenses that have been proposed are specific to simple models like linear classifiers and SVM, or the defenses are tailored to weaker attacks such as collision based attacks where feature space is well understood [268, 220, 223]. However, data sanitization defenses break when faced with stronger attacks. Table 6.6 shows a defense by anomaly filtering. averaged over 6 randomly seeded poisoning runs on CIFAR-10 (4% budget w/ $\varepsilon = 16$), we find that outlier detection is only marginally more successful than random guessing.

Table 6.6: Outlier detection is close to random-guessing for poison detection on CIFAR-10.

	10% filtering	20% filtering
Expected poisons removed (outlier method)	248	467
Expected clean removed (outlier method)	252	533
Expected poisons removed (random guessing)	200	400
Expected clean removed (random guessing)	300	600

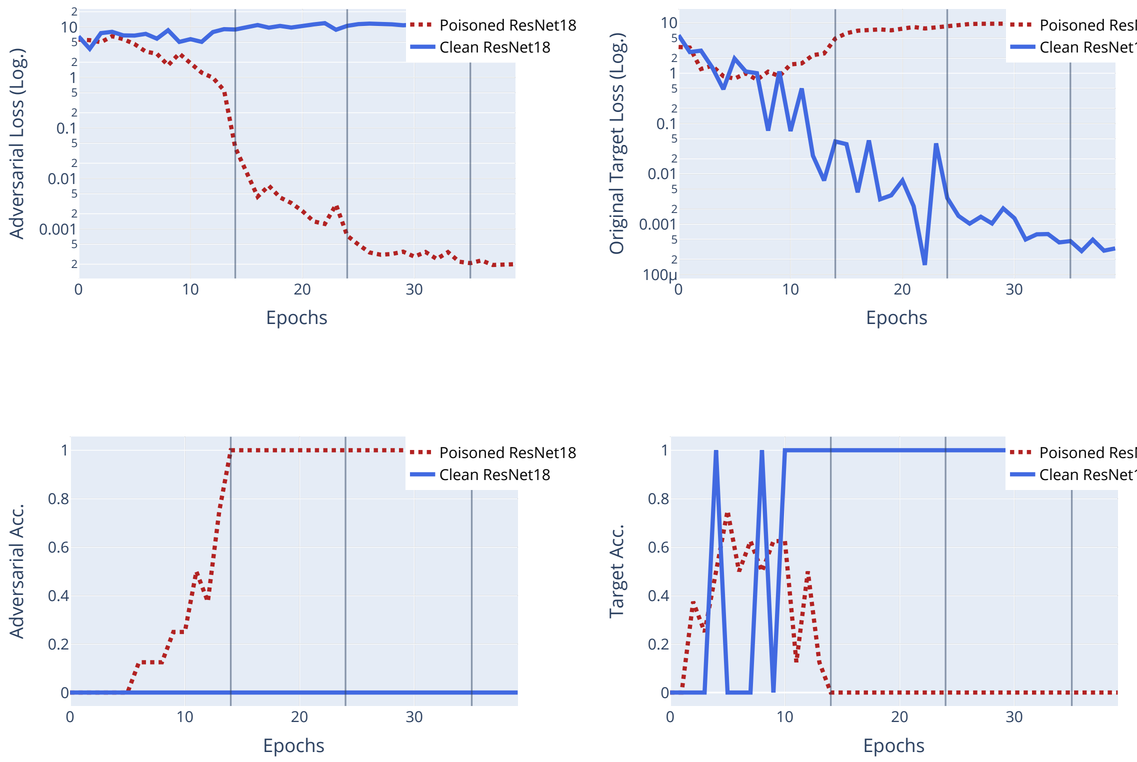


Figure 6.9: Cross entropy loss (Top) and accuracy (Bottom) for a given target with its adversarial label (left), and with its original label (right) shown for a poisoned and a clean ResNet-18. The clean model is used as victim for the poisoned model. The loss is averaged 8 times for the poisoned model. Learning rate drops are marked with gray horizontal bars.

6.F.3 Details: Defense by Differential Privacy

In Figure 6.4b we consider a defense by differential privacy. According to [122], gradient noise is the key factor that makes differentially private SGD [1] useful as a defense. As such we keep the gradient clipping fixed to a value of 1 and only increase the gradient noise in Figure 6.4b. To scale differentially private SGD, we only consider this gradient clipping on the mini-batch level, not the example level. This is reflected in the red, dashed line. A trivial counter-measure against this defense is shown as the solid red line. If the level of gradient noise is known to the attacker, then the attacker can brew poisoned data by the approach shown in Algorithm 6.1, but also add gradient noise and gradient clipping to the poison gradient. We use a naive strategy of redrawing the added noise every time the matching objective $\mathcal{B}(\Delta, \theta)$ is evaluated. It turns out that this yields a good baseline counter-attack against the defense through differential privacy.

6.F.4 Details: Gradient Alignment Visualization

Figure 6.11 visualizes additional details regarding Figure 6.2. Figure 6.11a replicates Figure 6.2 with linear scaling, whereas Figure 6.11b shows the behavior after epoch 14, which is the first learning rate drop. Note that in all figures each measurement is averaged over an epoch and the learning rate drops are marked with gray vertical bars. Figure 6.11c shows the opposite metric, that is the alignment of the original (non-adversarial) gradient. It is important to note for these figures, that the positive alignment is the crucial, whereas the magnitude of alignment is not as important. As this is the gradient averaged over

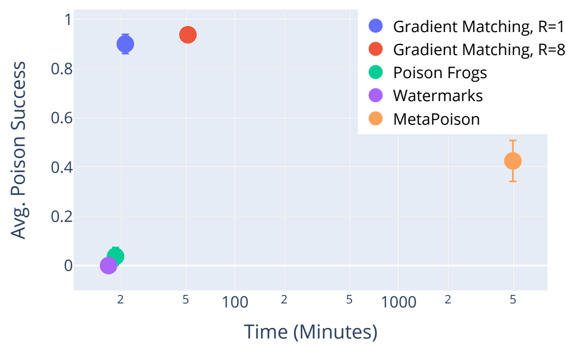
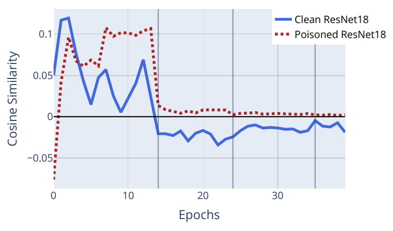
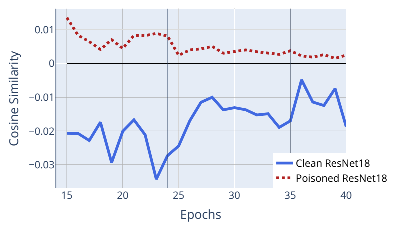
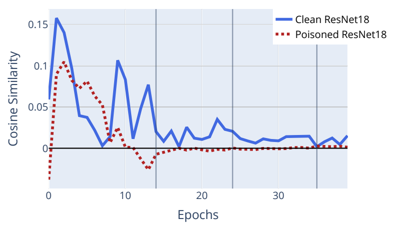


Figure 6.10: CIFAR-10 comparison without time and memory constraints for a ResNet18 with realistic training. Budget 1%, $\varepsilon = 16$. Note that the x-axis is logarithmic.







a) Alignment of $\nabla\mathcal{L}_{\text{adv}}(\theta)$ and $\nabla\mathcal{L}(\theta)$

b) Zoom: Alignment of $\nabla\mathcal{L}_{\text{adv}}(\theta)$ and $\nabla\mathcal{L}(\theta)$ from epoch 14.

c) Alignment of $\nabla\mathcal{L}_{\text{t}}(\theta)$ (orig. label) and $\nabla\mathcal{L}(\theta)$

Figure 6.11: Average batch cosine similarity, per epoch, between the adversarial gradient and the gradient of each mini-batch (left), and with its clean counterpart $\nabla\mathcal{L}_{\text{t}}(\theta) := \nabla_\theta\mathcal{L}(x^t, y^t)$ (right) for a poisoned and a clean ResNet-18. Each measurement is averaged over an epoch. Learning rate drops are marked with gray vertical bars.

the entire epoch, the contributions are from mini-batches can contain none or only a single poisoned example.

6.F.5 Ablation Studies - Reduced Brewing/Victim Training Data

In order to further test the strength and possible limitations of the discussed poisoning method, we perform several ablation studies, where we reduce either the training set known to the attacker or the set of poisons used by the victim, or both.

In many real world poisoning situations, it is not reasonable to assume that the victim will unwittingly add all poison examples to their training set, or that the attacker knows the full victim training set to begin with. For example, if the attacker puts 1000 poisoned images on social media, the victim might only scrape 300 of these. We test how dependent the method is on the victim training set by randomly removing a proportion of data (clean + poisoned) from the victim's training set. We then train the victim on the ablated poisoned dataset, and evaluate the target image to see if it is misclassified by the victim as the attacker's intended class. Then, we add another assumption - the brewing network does not have access to all victim training data when creating the poisons (see Table 6.7). We see that the attacker can still successfully poison the victim, even after a large portion of the victim's training data is removed, or

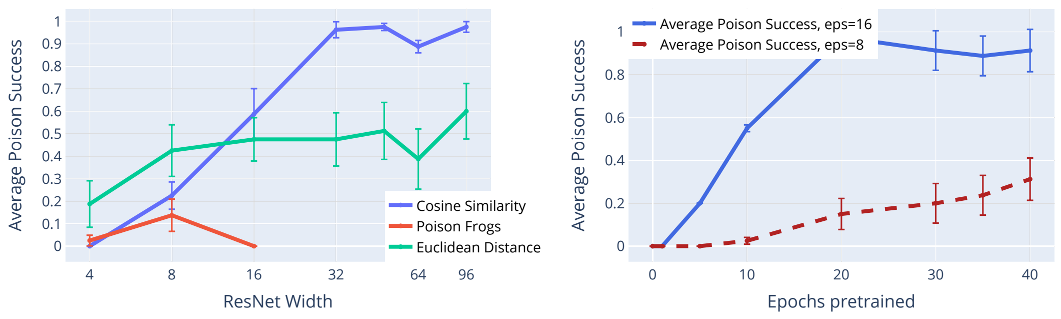


Figure 6.12: Ablation Studies. Left: avg. poison success for Euclidean Loss, cosine similarity and the Poison Frogs objective [256] for thin ResNet-18 variants. Right: Avg. poison success vs number of pretraining epochs.

the attacker does not have access to the full victim training set.

Table 6.7: Average poisoning success under victim training data ablation. In the first regime, victim ablation, a proportion of the victim's training data (clean + poisoned) is selected randomly and then the victim trains on this subset. In the second regime, pretrained + victim ablation, the pretrained network is trained on a randomly selected proportion of the data, and then the victim chose a new random subset of clean + poisoned data on which to train. All results averaged over 5 runs on ImageNet.

	70% data removed	50% data removed
victim ablation	60%	100%
pretrained + victim ablation	60%	80%

6.F.6 Ablation Studies - Method

Table 6.8 shows different variations of the proposed method. While using the Carlini-Wagner loss as a surrogate for cross entropy helped in [126], it does not help in our setting. We further find that running the proposed method for only 50 steps (instead of 250 as everywhere else in the paper) leads to a significant loss in avg. poison success. Lastly we investigate whether using euclidean loss instead of cosine similarity would be beneficial. This would basically imply trying to match (6.2) directly. Euclidean loss amounts to removing the invariance to gradient magnitude, in comparison to cosine similarity, which is invariant. We find that this is not beneficial in our experiments, and that the invariance with respect to gradient magnitude does allow for the construction of stronger poisoned datasets. Interestingly the discrepancy between both loss functions is related to the width of the network. In Figure 6.12 on the left, we visualize avg. poison success for modified ResNet-18s. The usual base width of 64 is replaced by the width value shown on the x-axis. For widths smaller than 16, the Euclidean loss dominates, but its effectiveness does not increase with width. In contrast the cosine similarity is superior for larger widths and seems to be able to make use of the greater representative power of the wider networks to find vulnerabilities. Figure 6.12 on the right examines the impact of the pretrained model that is supplied to Algorithm 6.1. We compare avg. poison success against the number of pretraining epochs for a budget of 1%, first with $\varepsilon = 16$ and then with $\varepsilon = 8$. It turns out that for the easier threat model of $\varepsilon = 8$, even pretraining to only 20 epochs can be enough for the algorithm to work well, whereas in the more difficult scenario of $\varepsilon = 8$, performance increases with pretraining effort.

Table 6.8: CIFAR-10 ablation runs. $\varepsilon = 16$, budget is 1%. All values are computed for ResNet-18 models.

Setup	Avg. Poison Success $\%(\pm\text{SE})$	Validation Acc.%
Baseline (full data aug., $R = 8$, $M = 250$	91.25% (± 6.14)	92.20%
Carlini-Wagner loss instead of $\mathcal{L}$	77.50% (± 9.32)	92.08%
Fewer Opt. Steps ($M = 50$)	40.00% (± 10.87)	92.05%
Euclidean Loss instead of cosine sim.	61.25% (± 9.75)	92.09%

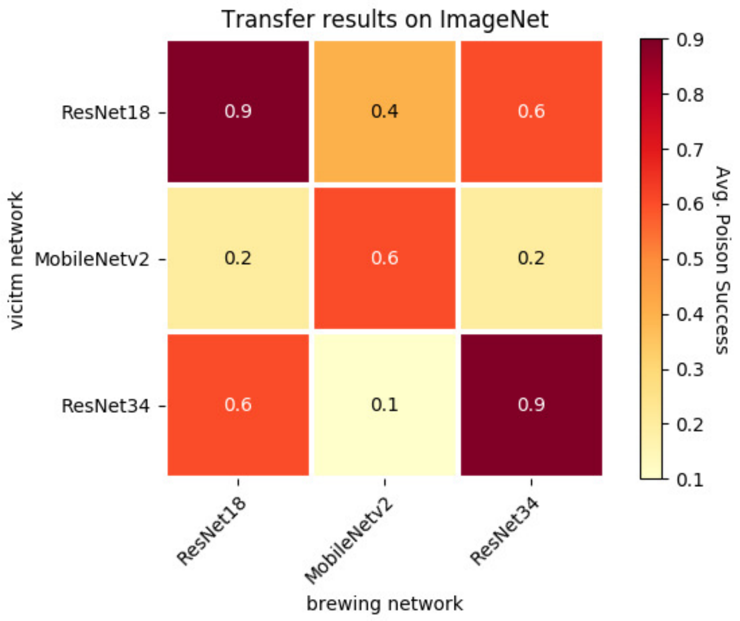


Figure 6.13: Direct transfer results on common architectures. Averaged over 10 runs with budget of 0.1% and ε-bound of 16. Note that for these transfer experiments, the model was *only* trained on the "brewing" network, without knowledge of the victim. This shows a transferability to unknown architectures.

Table 6.9: CIFAR-10 ablation runs. $\varepsilon = 16$, budget is 1%. All values are computed for ResNet-18 models. Averaged over 5 runs.

	Avg. Poison Success	
Setup	($\%$ of total targets poisoned successfully)	Effective Budget / Target
1 target (Baseline)	90.00%	1%
5 targets	32.00%	0.2%
10 targets	14.00%	0.1%

6.F.7 Transfer Experiments

In addition to the fully black-box pipeline of the AutoML experiments in section 6.B, we test the transferability of our poisoning method against other commonly used architectures. Transfer results on CIFAR-10 can be found in Table 6.3. On Imagenet, we brew poisons with a variety of networks, and test against other networks. We find that poisons crafted with one architecture can transfer and cause targeted mis-classification in other networks (see Figure 6.13).

6.F.8 Multi-Target Experiments

We also perform limited tests on poisoning multiple targets simultaneously. We find that while keeping the small poison budget of 1% fixed, we are able to successfully poison more than one target while optimizing poisons simultaneously, see Table 6.9. Effectively, however, every target image gradient has to be matched with an increasingly smaller budget. As the target images are drawn at random and not semantically similar (aside from their shared class), their synergy is limited - for example the 5 targets experiment reaches an accuracy of 32%, which is only 14% better than the naive baseline of $\frac{90\%}{5}$ one might expect. One encouraging result from the standpoint of poisoning though is that the multiple targets do not compete against each other, canceling out their respective different alignments.

CHAPTER 7

Conclusions

This dissertation developed and showcased a variety of applications of optimization techniques in computer vision, which aid in the formalization and facilitation of a variety of goals, from learning of energy models to the analysis of security aspects.

Each chapter shows an immediate application in computer vision:

- Many model-based approaches in computer vision lead to non-convex optimization problems, a subset of which can be solved accurately by functional lifting techniques, although this comes with significant computational costs. Chapter 2 improves the efficiency of these lifting applications with new insights into graph-based discretizations with examples in image segmentation and stereo reconstruction, while Chapter 3 extends the applicability of functional lifting techniques to operator-based imaging models, which appear, for example, in Time-of-Flight applications.
- The applicability of such model-based approaches is limited by the property that all parts of the model have to be hand-crafted and designed manually. Chapter 4 presents efficient learning strategies for such energy models, that can be used to learn additional parameters from data, while keeping the overall model structure. We discuss examples where this technique can be used to learn image regularization for computed tomography reconstruction as well as natural image denoising.
- Machine learning models based on deep neural networks are more and more relied on in real-world scenarios, also in computer vision. However, because of their inherent complexity it can be difficult to investigate security implications and vulnerabilities of these models. We show in Chapter 5 that the security of models trained by federated learning algorithms can be breached practically, even for large batches of data; and we investigate in Chapter 6 that training data can be modified maliciously to encode backdoors into machine learning models, even for large, industrial-sized models. We show that this applies in a very real scenario by uploading a modified ImageNet dataset to a cloud service provider, Google AutoML, and finding that the backdoor is still present in the cloud model.

These applications are based on new theoretical insights that are developed and analyzed over the course of these works:

- Chapter 2 constructs discretization of generalized minimal partition problems on a graph, generalizing the discretization to arbitrary grid shapes.

- Chapter 3 analyzes the convergence of majorization strategies for very general composite optimization problems, allowing for non-convexity of the majorizing function.
- Composite optimization is revisited in Chapter 4 in the special case of bilevel optimization problems. We develop approximate solution strategies for these bilevel problems based on optimizing a tractable upper bound, i.e. a majorizer, in the case of convex, but possibly non-smooth lower-level problems. We then apply these approximations iteratively based on developments from Chapter 3.
- In Chapter 5 and Chapter 6 we investigate bilevel problems in machine learning security and develop approximations based on gradient matching to solve them even for deep neural networks, extending considerations of gradient penalties discussed in Chapter 4.
- We further provide theoretical insights into the invertibility of neural network layers in Chapter 5 and in Chapter 6 we analyze which conditions are necessary for the bilevel approximation to succeed.

We hope that both practical and theoretical implications of this work will aid researchers and practitioners in developing computer vision that is precise and secure.

In more general terms, we find in many applications that the devil is really in the details. Not only the formalization of the considered computer vision applications into optimization problems, but also the tractable approximation of these formalized problems with surrogate problems are crucial fields of study. In many cases, only considering both details of the application and formalized knowledge about the problem class lead to progress. While future work will naturally continue to consider a multitude of direct applications of optimization in computer vision, it is the formalization of approximation tools and strategies used in applications into general methods for optimization that will be more and more substantial, as the overall complexity of the field increases.

References

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, *Deep Learning with Differential Privacy*, in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16, Vienna, Austria, Oct. 2016, Association for Computing Machinery, pp. 308–318, https://doi.org/10.1145/2976749.2978318. [Cited on pages 113, 129, 109, and 124.]

[2] R. Achanta, A. S. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, *SLIC Superpixels Compared to State-of-the-Art Superpixel Methods*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 34 (2012), pp. 2274–2282, https://doi.org/10.1109/TPAMI.2012.120. [Cited on pages 7, 17, 6, and 16.]

[3] H. Aghakhani, D. Meng, Y.-X. Wang, C. Kruegel, and G. Vigna, *Bullseye Polytope: A Scalable Clean-Label Poisoning Attack with Improved Transferability*, arXiv:2005.00191 [cs, stat], (2020), https://arxiv.org/abs/2005.00191. [Cited on pages 114, 118, and 110.]

[4] G. Alberti, G. Bouchitté, and G. Dal Maso, *The calibration method for the Mumford-Shah functional and free-discontinuity problems*, Calculus of Variations and Partial Differential Equations, 16 (2003), pp. 299–333, https://doi.org/10.1007/s005260100152. [Cited on pages 41 and 39.]

[5] Y. Altun, I. Tsochantaridis, and T. Hofmann, *Hidden Markov Support Vector Machines*, in Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML'03, AAAI Press, 2003, pp. 3–10. [Cited on pages 58 and 53.]

[6] L. Ambrosio, N. Fusco, and D. Pallara, *Functions of Bounded Variation and Free Discontinuity Problems*, Oxford university press, Oxford, 2000. [Cited on pages 8 and 7.]

[7] B. Amos and J. Z. Kolter, *OptNet: Differentiable Optimization as a Layer in Neural Networks*, in International Conference on Machine Learning, July 2017, pp. 136–145. [Cited on pages 57 and 52.]

[8] V. Antun, F. Renna, C. Poon, B. Adcock, and A. C. Hansen, *On instabilities of deep learning in image reconstruction - Does AI come at a cost?*, arXiv:1902.05300 [cs], (2019), https://arxiv.org/abs/1902.05300. [Cited on pages 56, 65, 51, and 60.]

[9] M. Artina, M. Fornasier, and F. Solombrino, *Linearly Constrained Nonsmooth and Nonconvex Minimization*, SIAM J. Optim., 23 (2013), pp. 1904–1937, https://doi.org/10.1137/120869079. [Cited on pages 40, 44, 38, and 41.]

[10] A. Athalye, N. Carlini, and D. Wagner, *Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples*, arXiv:1802.00420 [cs], (2018), https://arxiv.org/abs/1802.00420. [Cited on pages 90 and 86.]

[11] H. Attouch, J. Bolte, and B. F. Svaiter, *Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods*, Math. Program., 137 (2013), pp. 91–129, https://doi.org/10.1007/s10107-011-0484-9. [Cited on pages 24, 36, 37, 22, 33, and 34.]

[12] H. Attouch, G. Buttazzo, and G. Michaille, *Variational Analysis in Sobolev and BV Spaces: Applications to PDEs and Optimization*, MPS-SIAM Series on Optimization, Society for Industrial and Applied Mathematics : Mathematical Programming Society, Philadelphia, 2006. [Cited on pages 8 and 7.]

[13] C. Audet and J. Dennis, *Analysis of Generalized Pattern Searches*, SIAM J. Optim., 13 (2002), pp. 889–903, https://doi.org/10.1137/S1052623400378742. [Cited on pages 46 and 44.]

[14] A. Barbu, *Learning real-time MRF inference for image denoising*, 2009 IEEE Conference on Computer Vision and Pattern Recognition, (2009), pp. 1574–1581, https://doi.org/10.1109/CVPRW.2009.5206811. [Cited on pages 58, 76, 53, and 71.]

[15] J. F. Bard and J. E. Falk, *An explicit solution to the multi-level programming problem*, Computers & Operations Research, 9 (1982), pp. 77–100, https://doi.org/10.1016/0305-0548(82)90007-7. [Cited on pages 75, 114, 70, and 110.]

[16] C. Bardaro, J. Musielak, and G. Vinti, *Nonlinear Integral Operators and Applications*, Walter de Gruyter, Jan. 2003. [Cited on pages 39 and 36.]

[17] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, *The security of machine learning*, Mach. Learn., 81 (2010), pp. 121–148, https://doi.org/10.1007/s10994-010-5188-5. [Cited on pages 114 and 109.]

[18] H. H. Bauschke, J. Bolte, and M. Teboulle, *A Descent Lemma Beyond Lipschitz Gradient Continuity: First-Order Methods Revisited and Applications*, Mathematics of Operations Research, 42 (2017), pp. 330–348, https://doi.org/10.1287/moor.2016.0817. [Cited on pages 24, 31, 41, 44, 84, 22, 28, 29, 38, and 79.]

[19] H. H. Bauschke and J. J. Borwein, *Legendre Functions and the Method of Random Bregman Projections*, Journal of Convex Analysis, 4 (1997), pp. 27–67. [Cited on pages 41, 69, 38, and 64.]

[20] H. H. Bauschke, J. M. Borwein, and P. L. Combettes, *Essential Smoothness, Essential Strict Convexity, and Legendre Functions in Banach Spaces*, Communications in Contemporary Mathematics, 03 (2001), pp. 615–647, https://doi.org/10.1142/S0219199701000524. [Cited on pages 28, 35, 26, and 33.]

[21] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, CMS Books in Mathematics, Springer New York, New York, NY, 2011. [Cited on pages 58, 71, 53, and 66.]

[22] H. H. Bauschke, P. L. Combettes, and D. Noll, *Joint minimization with alternating Bregman proximity operators*, Pacific Journal of Optimization, 2 (2006), pp. 401–424. [Cited on pages 84 and 79.]

[23] H. H. Bauschke, M. N. Dao, and S. B. Lindstrom, *Regularizing with Bregman-Moreau envelopes*, arXiv:1705.06019 [math], (2017), https://arxiv.org/abs/1705.06019. [Cited on pages 84 and 79.]

[24] A. Beck and M. Teboulle, *Mirror descent and nonlinear projected subgradient methods for convex optimization*, Operations Research Letters, 31 (2003), pp. 167–175, https://doi.org/10.1016/S0167-6377(02)00231-6. [Cited on pages 65, 74, 60, and 69.]

[25] A. Beck and M. Teboulle, *A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 183–202, https://doi.org/10.1137/080716542. [Cited on pages 24, 74, 22, and 69.]

[26] K. P. Bennett, G. Kunapuli, J. Hu, and J.-S. Pang, *Bilevel Optimization and Machine Learning*, in IEEE World Congress on Computational Intelligence, WCCI 2008, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 25–47, https://doi.org/10.1007/978-3-540-68860-0_2. [Cited on pages 57, 75, 52, and 70.]

[27] M. Benning, M. M. Betcke, M. J. Ehrhardt, and C. Schönlieb, *Gradient descent in a generalised Bregman distance framework*, in Joint Conference Geometric Numerical Integration and Its Applications, vol. 74, Melbourne, Mar. 2017, MI Lecture Note Kyushu University, pp. 40–45. [Cited on pages 24 and 22.]

[28] M. Benning and M. Burger, *Modern regularization methods for inverse problems*, Acta Numerica, 27 (2018), pp. 1–111, https://doi.org/10.1017/S0962492918000016. [Cited on pages 2, 61, 69, 88, 56, 64, and 84.]

[29] J. Bergstra, D. Yamins, and D. D. Cox, *Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures*, JMLR, 2013. [Cited on pages 75 and 70.]

[30] W. F. Bialas and M. H. Karwan, *Two-Level Linear Programming*, Manage. Sci., 30 (1984), pp. 1004–1020, https://doi.org/10.1287/mnsc.30.8.1004. [Cited on pages 75 and 70.]

[31] B. Biggio, B. Nelson, and P. Laskov, *Poisoning Attacks against Support Vector Machines*, arXiv:1206.6389 [cs, stat], (2012), https://arxiv.org/abs/1206.6389. [Cited on pages 113, 114, 122, 108, 110, and 118.]

[32] J. Bolte, A. Daniilidis, and A. Lewis, *The Łojasiewicz Inequality for Nonsmooth Subanalytic Functions with Applications to Subgradient Dynamical Systems*, SIAM J. Optim., 17 (2007), pp. 1205–1223, https://doi.org/10.1137/050644641. [Cited on pages 35 and 32.]

[33] J. Bolte, A. Daniilidis, A. Lewis, and M. Shiota, *Clarke Subgradients of Stratifiable Functions*, SIAM J. Optim., 18 (2007), pp. 556–572, https://doi.org/10.1137/060670080. [Cited on pages 24, 35, 22, and 32.]

[34] J. Bolte and E. Pauwels, *Majorization-Minimization Procedures and Convergence of SQP Methods for Semi-Algebraic and Tame Programs*, Mathematics of OR, 41 (2016), pp. 442–465, https://doi.org/10.1287/moor.2015.0735. [Cited on pages 25 and 22.]

[35] J. Bolte, S. Sabach, and M. Teboulle, *Proximal alternating linearized minimization for nonconvex and nonsmooth problems*, Math. Program., 146 (2014), pp. 459–494, https://doi.org/10.1007/s10107-013-0701-9. [Cited on pages 35 and 32.]

[36] J. Bolte, S. Sabach, and M. Teboulle, *Nonconvex Lagrangian-Based Optimization: Monitoring Schemes and Global Convergence*, arXiv:1801.03013 [math], (2018), https://arxiv.org/abs/1801.03013. [Cited on pages 25, 27, and 24.]

[37] J. Bolte, S. Sabach, M. Teboulle, and Y. Vaisbourd, *First Order Methods Beyond Convexity and Lipschitz Gradient Continuity with Applications to Quadratic Inverse Problems*, SIAM J. Optim., 28 (2018), pp. 2131–2151, https://doi.org/10.1137/17M1138558. [Cited on pages 24, 30, 31, 35, 36, 37, 38, 41, 22, 27, 28, 33, and 34.]

[38] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, *Towards Federated Learning at Scale: System Design*, arXiv:1902.01046 [cs, stat], (2019), https://arxiv.org/abs/1902.01046. [Cited on pages 87, 95, 83, and 91.]

[39] S. Bonettini, I. Loris, F. Porta, and M. Prato, *Variable Metric Inexact Line-Search-Based Methods for Nonsmooth Optimization*, SIAM Journal on Optimization, 26 (2016), pp. 891–921, https://doi.org/10.1137/15M1019325. [Cited on pages 30 and 27.]

[40] Y. Boykov and G. Funka-Lea, *Graph Cuts and Efficient N-D Image Segmentation*, International Journal of Computer Vision, 70 (2006), pp. 109–131, https://doi.org/10.1007/s11263-006-7934-5. [Cited on pages 6, 7, 17, 5, and 16.]

[41] Y. Boykov, O. Veksler, and R. Zabih, *Fast approximate energy minimization via graph cuts*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 23 (2001), pp. 1222–1239, https://doi.org/10.1109/34.969114. [Cited on pages 7, 12, 17, 27, 41, 6, 11, 16, 25, and 39.]

[42] X. Bresson and T. F. Chan, *Non-local Unsupervised Variational Image Segmentation Models*, UCLA CAM report, University of California, Los Angeles, 2008. [Cited on pages 10 and 9.]

[43] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, *High Accuracy Optical Flow Estimation Based on a Theory for Warping*, in Computer Vision - ECCV 2004, vol. 3024, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 25–36, https://doi.org/10.1007/978-3-540-24673-2_3. [Cited on pages 6 and 5.]

[44] M. Burger, *Bregman Distances in Inverse Problems and Partial Differential Equations*, in Advances in Mathematical Modeling, Optimization and Optimal Control, Springer Optimization and Its Applications, Springer International Publishing, Cham, 2016, pp. 3–33, https://doi.org/10.1007/978-3-319-30785-5_2. [Cited on pages 58, 69, 53, and 64.]

[45] M. Burger and S. Osher, *A Guide to the TV zoo*, in PDE Based Reconstruction Methods in Imaging, no. 2090 in Lecture Notes in Mathematics, Springer International Publishing, Switzerland, first ed., 2013. [Cited on pages 7, 16, 41, 6, 15, and 39.]

[46] D. Butnariu and G. Kassay, *A Proximal-Projection Method for Finding Zeros of Set-Valued Operators*, SIAM J. Control Optim., 47 (2008), pp. 2096–2136, https://doi.org/10.1137/070682071. [Cited on pages 61, 69, 56, and 64.]

[47] L. Calatroni, C. Cao, J. C. De Los Reyes, C.-B. Schönlieb, and T. Valkonen, *Bilevel approaches for learning of variational imaging models*, Variational Methods: In Imaging and Geometric Control, 18 (2017), p. 252. [Cited on pages 57 and 52.]

[48] L. Calatroni, C. Chung, J. C. D. L. Reyes, C.-B. Schönlieb, and T. Valkonen, *Bilevel approaches for learning of variational imaging models*, arXiv:1505.02120 [cs, math], (2015), https://arxiv.org/abs/1505.02120. [Cited on pages 75 and 70.]

[49] E. J. Candes, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Transactions on Information Theory, 52 (2006), pp. 489–509, https://doi.org/10.1109/TIT.2005.862083. [Cited on pages 88 and 84.]

[50] W. Candler and R. Townsley, *A linear two-level programming problem*, Computers & Operations Research, 9 (1982), pp. 59–76, https://doi.org/10.1016/0305-0548(82)90006-5. [Cited on pages 75 and 70.]

[51] N. Carlini and D. Wagner, *Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods*, in Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec '17, Dallas, Texas, USA, Nov. 2017, Association for Computing Machinery, pp. 3–14, https://doi.org/10.1145/3128572.3140444. [Cited on pages 126 and 121.]

[52] A. Chambolle, V. Caselles, D. Cremers, M. Novaga, and T. Pock, *An introduction to total variation for image analysis*, Theoretical foundations and numerical methods for sparse recovery, 9 (2010), p. 227. [Cited on pages 56 and 57.]

[53] A. Chambolle, D. Cremers, and T. Pock, *A Convex Approach to Minimal Partitions*, SIAM Journal on Imaging Sciences, 5 (2012), pp. 1113–1158, https://doi.org/10.1137/110856733. [Cited on pages 7, 8, 9, 27, 65, 6, 25, and 60.]

[54] A. Chambolle and T. Pock, *A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging*, J Math Imaging Vis, 40 (2011), pp. 120–145, https://doi.org/10.1007/s10851-010-0251-1. [Cited on pages 16, 18, 48, 74, 15, 17, 45, and 69.]

[55] A. Chambolle and T. Pock, *On the ergodic convergence rates of a first-order primal–dual algorithm*, Mathematical Programming, 159 (2016), pp. 253–287, https://doi.org/10.1007/s10107-015-0957-3. [Cited on pages 74 and 69.]

[56] T. F. Chan, S. Esedoglu, and M. Nikolova, *Algorithms for Finding Global Minimizers of Image Segmentation and Denoising Models*, SIAM J. Appl. Math., 66 (2006), pp. 1632–1648, https://doi.org/10.1137/040615286. [Cited on pages 22 and 20.]

[57] T. F. Chan and L. A. Vese, *Active contours without edges*, IEEE Transactions on image processing, 10 (2001), pp. 266–277. [Cited on pages 12, 56, 65, 11, 51, and 60.]

[58] S. Chandra and I. Kokkinos, *Fast, Exact and Multi-scale Inference for Semantic Image Segmentation with Deep Gaussian CRFs*, in Computer Vision – ECCV 2016, Lecture Notes in Computer Science, Springer International Publishing, 2016, pp. 402–418. [Cited on pages 58, 76, 53, and 71.]

[59] B. Chang, L. Meng, E. Haber, L. Ruthotto, D. Begert, and E. Holtham, *Reversible Architectures for Arbitrarily Deep Residual Neural Networks*, arXiv:1709.03698 [cs, stat], (2017), https://arxiv.org/abs/1709.03698. [Cited on pages 97 and 93.]

[60] G. Charpiat, N. Girard, L. Felardos, and Y. Tarabalka, *Input Similarity from the Neural Network Perspective*, in Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 5342–5351. [Cited on pages 89, 122, 85, and 117.]

[61] G. Chen and R. Rockafellar, *Convergence Rates in Forward–Backward Splitting*, SIAM J. Optim., 7 (1997), pp. 421–444, https://doi.org/10.1137/S1052623495290179. [Cited on pages 24 and 22.]

[62] G. Chen and M. Teboulle, *Convergence Analysis of a Proximal-Like Minimization Algorithm Using Bregman Functions*, SIAM J. Optim., 3 (1993), pp. 538–543, https://doi.org/10.1137/0803026. [Cited on pages 63 and 58.]

[63] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, *Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs*, in International Conference on Learning Representations (ICLR), 2015, https://arxiv.org/abs/1412.7062. [Cited on pages 56 and 51.]

[64] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*, arXiv:1606.00915 [cs], (2016), https://arxiv.org/abs/1606.00915. [Cited on pages 7, 58, 76, 6, 53, and 71.]

[65] Y. Chen and T. Pock, *Trainable Nonlinear Reaction Diffusion: A Flexible Framework for Fast and Effective Image Restoration*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 39 (2017), pp. 1256–1272, https://doi.org/10.1109/TPAMI.2016.2596743. [Cited on pages 58, 76, 53, and 71.]

[66] Y. Chen, T. Pock, and H. Bischof, *Learning l1-based analysis and synthesis sparsity priors using bi-level optimization*, in Neural Information Processing Systems Confercence (NIPS) 2012, 2012. [Cited on pages 57, 68, 75, 52, 63, and 70.]

[67] Y. Chen, R. Ranftl, and T. Pock, *A bi-level view of inpainting - based image compression*, in Computer Vision Winter Workshop, ., 2014. [Cited on pages 57, 74, 75, 52, 69, and 70.]

[68] Y. Chen, R. Ranftl, and T. Pock, *Insights Into Analysis Operator Learning: From Patch-Based Sparse Models to Higher Order MRFs*, IEEE Transactions on Image Processing, 23 (2014), pp. 1060–1072, https://doi.org/10.1109/TIP.2014.2299065. [Cited on pages 56, 57, 67, 68, 74, 75, 160, 51, 52, 62, 63, 69, 70, and 151.]

[69] Y. Chen, W. Yu, and T. Pock, *On Learning Optimized Reaction Diffusion Processes for Effective Image Restoration*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5261–5269. [Cited on pages 58, 76, 53, and 71.]

[70] T. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman, *Project Adam: Building an Efficient and Scalable Deep Learning Training System*, in 11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14), 2014, pp. 571–582. [Cited on pages 86 and 82.]

[71] E. Chouzenoux, J.-C. Pesquet, and A. Repetti, *Variable Metric Forward–Backward Algorithm for Minimizing the Sum of a Differentiable Function and a Convex Function*, J Optim Theory Appl, 162 (2014), pp. 107–132, https://doi.org/10.1007/s10957-013-0465-7. [Cited on pages 24, 41, 22, and 38.]

[72] M. Collins, *Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms*, in Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10, EMNLP '02, Stroudsburg, PA, USA, 2002, Association for Computational Linguistics, pp. 1–8, https://doi.org/10.3115/1118693.1118694. [Cited on pages 58, 77, 53, and 72.]

[73] A. Colovic, P. Knöbelreiter, A. Shekhovtsov, and T. Pock, *End-to-End Training of Hybrid CNN-CRF Models for Semantic Segmentation using Structured Learning*, in Computer Vision Winter Workshop, Feb. 2017. [Cited on pages 58, 76, 53, and 71.]

[74] B. Colson, P. Marcotte, and G. Savard, *An overview of bilevel optimization*, Annals of Operations Research, 153 (2007), pp. 235–256, https://doi.org/10.1007/s10479-007-0176-2. [Cited on pages 57, 75, 52, and 70.]

[75] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, *The Cityscapes Dataset for Semantic Urban Scene Understanding*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223. [Cited on pages 74 and 69.]

[76] K. Crammer and Y. Singer, *On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines*, Journal of Machine Learning Research, 2 (2001), pp. 265–292. [Cited on pages 76 and 71.]

[77] D. Cremers, T. Pock, K. Kolev, and A. Chambolle, *Convex Relaxation Techniques for Segmentation, Stereo and Multiview Reconstruction*, in Markov Random Fields for Vision and Image Processing, MIT Press, Boston, 2011. [Cited on pages 7, 56, 57, 65, 6, 51, 52, and 60.]

[78] H. Daumé III and D. Marcu, *Learning as Search Optimization: Approximate Large Margin Methods for Structured Prediction*, arXiv:0907.0809 [cs], (2009), https://arxiv.org/abs/0907.0809. [Cited on pages 76 and 71.]

[79] J. C. De Los Reyes, C.-B. Schönlieb, and T. Valkonen, *The structure of optimal parameters for image restoration problems*, Journal of Mathematical Analysis and Applications, 434 (2016), pp. 464–500, https://doi.org/10.1016/j.jmaa.2015.09.023. [Cited on pages 57, 75, 52, and 70.]

[80] J. C. De Los Reyes, C.-B. Schönlieb, and T. Valkonen, *Bilevel Parameter Learning for Higher-Order Total Variation Regularisation Models*, J Math Imaging Vis, 57 (2017), pp. 1–25, https://doi.org/10.1007/s10851-016-0662-8. [Cited on pages 57, 75, 52, and 70.]

[81] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, *Why Do Adversarial Attacks Transfer? Explaining Transferability of Evasion and Poisoning Attacks*, in 28th {USENIX} Security Symposium ({USENIX} Security 19), 2019, pp. 321–338. [Cited on pages 114, 122, 110, and 117.]

[82] S. Dempe, *Foundations of Bilevel Programming*, Nonconvex Optimization and Its Applications, Springer US, 2002. [Cited on pages 57, 75, 52, and 70.]

[83] S. Dempe and J. Dutta, *Is bilevel programming a special case of a mathematical program with complementarity constraints?*, Math. Program., 131 (2012), pp. 37–48, https://doi.org/10.1007/s10107-010-0342-1. [Cited on pages 57, 75, 52, and 70.]

[84] S. Dempe, V. Kalashnikov, G. A. Pérez-Valdés, and N. Kalashnykova, *Bilevel Programming Problems: Theory, Algorithms and Applications to Energy Networks*, Energy Systems, Springer-Verlag, Berlin Heidelberg, 2015. [Cited on pages 57, 75, 78, 52, 70, and 73.]

[85] A. Dosovitskiy and T. Brox, *Generating Images with Perceptual Similarity Metrics based on Deep Networks*, in Advances in Neural Information Processing Systems 29, Curran Associates, Inc., 2016, pp. 658–666. [Cited on pages 88, 89, 84, and 85.]

[86] A. Dosovitskiy and T. Brox, *Inverting Visual Representations With Convolutional Networks*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4829–4837. [Cited on pages 88, 89, 84, and 85.]

[87] D. Drusvyatskiy, *Slope and Geometry in Variational Mathematics*, PhD thesis, Cornell University, 2013. [Cited on pages 34, 35, and 32.]

[88] D. Drusvyatskiy, A. D. Ioffe, and A. S. Lewis, *Nonsmooth optimization using Taylor-like models: Error bounds, convergence, and termination criteria*, arXiv:1610.03446 [math], (2016), https://arxiv.org/abs/1610.03446. [Cited on pages 24, 25, 30, 77, 22, 23, 27, and 72.]

[89] D. Drusvyatskiy and C. Paquette, *Efficiency of minimizing compositions of convex functions and smooth maps*, arXiv:1605.00125 [math], (2016), https://arxiv.org/abs/1605.00125. [Cited on pages 25, 30, 23, and 27.]

[90] J. Eckstein and D. P. Bertsekas, *On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Mathematical Programming, 55 (1992), pp. 293–318, https://doi.org/10.1007/BF01581204. [Cited on pages 16 and 15.]

[91] A. Elmoataz, O. Lezoray, and S. Bougleux, *Nonlocal Discrete Regularization on Weighted Graphs: A framework for Image and Manifold Processing*, IEEE Transactions on Image Processing, 17 (2008), pp. 1047–1060. [Cited on pages 8 and 7.]

[92] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry, *A Rotation and a Translation Suffice: Fooling CNNs with Simple Transformations*, arXiv:1712.02779 [cs, stat], (2017), https://arxiv.org/abs/1712.02779. [Cited on pages 126 and 121.]

[93] M. Fang, G. Yang, N. Z. Gong, and J. Liu, *Poisoning Attacks to Graph-Based Recommender Systems*, in Proceedings of the 34th Annual Computer Security Applications Conference, ACSAC '18, San Juan, PR, USA, Dec. 2018, Association for Computing Machinery, pp. 381–392, https://doi.org/10.1145/3274694.3274706. [Cited on pages 112 and 108.]

[94] S. G. Finlayson, H. W. Chung, I. S. Kohane, and A. L. Beam, *Adversarial Attacks Against Medical Deep Learning Systems*, arXiv:1804.05296 [cs, stat], (2018), https://arxiv.org/abs/1804.05296. [Cited on pages 56 and 51.]

[95] T. Finley and T. Joachims, *Training structural SVMs when exact inference is intractable*, in Proceedings of the 25th International Conference on Machine Learning - ICML '08, Helsinki, Finland, 2008, ACM Press, pp. 304–311, https://doi.org/10.1145/1390156.1390195. [Cited on pages 76 and 71.]

[96] C. Finn, P. Abbeel, and S. Levine, *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*, arXiv:1703.03400 [cs], (2017), https://arxiv.org/abs/1703.03400. [Cited on pages 114, 123, 110, and 119.]

[97] M. Fredrikson, S. Jha, and T. Ristenpart, *Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures*, in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15, Denver, Colorado, USA, Oct. 2015, Association for Computing Machinery, pp. 1322–1333, https://doi.org/10.1145/2810103.2813677. [Cited on pages 88 and 84.]

[98] T. Frerix, T. Möllenhoff, M. Moeller, and D. Cremers, *Proximal Backpropagation*, in International Conference on Learning Representations (ICLR), 2018, https://arxiv.org/abs/1706.04638. [Cited on pages 27 and 24.]

[99] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Computers & Mathematics with Applications, 2 (1976), pp. 17–40, https://doi.org/10.1016/0898-1221(76)90003-1. [Cited on pages 27 and 24.]

[100] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, *Property Inference Attacks on Fully Connected Neural Networks using Permutation Invariant Representations*, in Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto Canada, Jan. 2018, ACM, pp. 619–633, https://doi.org/10.1145/3243734.3243834. [Cited on pages 88 and 84.]

[101] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, *Inverting Gradients - How easy is it to break privacy in federated learning?*, in Advances in Neural Information Processing Systems, vol. 33, 2020. [Cited on pages 85, 97, 122, 81, 93, and 117.]

[102] J. Geiping, L. Fowl, W. R. Huang, W. Czaja, G. Taylor, M. Moeller, and T. Goldstein, *Witches' Brew: Industrial Scale Data Poisoning via Gradient Matching*, arXiv:2009.02276 [cs], (2020), https://arxiv.org/abs/2009.02276. [Cited on pages 86, 111, 82, and 107.]

[103] J. Geiping, F. Gaede, H. Bauermeister, and M. Moeller, *Fast Convex Relaxations using Graph Discretizations*, in 31st British Machine Vision Conference (BMVC 2020, Oral Presentation), Virtual, Sept. 2020. [Cited on pages 5, 15, 4, and 14.]

[104] J. Geiping and M. Moeller, *Composite Optimization by Nonconvex Majorization-Minimization*, SIAM J. Imaging Sci., (2018), pp. 2494–2528, https://doi.org/10.1137/18M1171989. [Cited on pages 21, 52, 63, 77, 19, 48, 58, and 72.]

[105] J. Geiping and M. Moeller, *Parametric Majorization for Data-Driven Energy Minimization Methods*, in Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 10262–10273. [Cited on pages 69 and 64.]

[106] G. Gilboa and S. Osher, *Nonlocal Operators with Applications to Image Processing*, Multiscale Model. Simul., 7 (2008), pp. 1005–1028, https://doi.org/10.1137/070698592. [Cited on pages 10 and 9.]

[107] K. Gimpel and N. A. Smith, *Softmax-Margin CRFs: Training Log-Linear Models with Cost Functions*, in Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Los Angeles, California, June 2010, Association for Computational Linguistics, pp. 733–736. [Cited on pages 58 and 53.]

[108] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st ed., 1989. [Cited on pages 46 and 44.]

[109] M. Goldblum, J. Geiping, A. Schwarzschild, M. Moeller, and T. Goldstein, *Truth or backpropaganda? An empirical investigation of deep learning theory*, in Eighth International Conference on Learning Representations (ICLR 2020, Oral Presentation), Apr. 2020. [Cited on pages 97 and 93.]

[110] S. Gould, B. Fernando, A. Cherian, P. Anderson, R. S. Cruz, and E. Guo, *On Differentiating Parameterized Argmin and Argmax Problems with Application to Bi-level Optimization*, arXiv:1607.05447 [cs, math], (2016), https://arxiv.org/abs/1607.05447. [Cited on pages 57, 75, 52, and 70.]

[111] A. Griewank, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000. [Cited on pages 58, 76, 53, and 71.]

[112] S. Guinard, L. Landrieu, L. Caraffa, and B. Vallet, *Piecewise-Planar Approximation of Large 3D Data as Graph-Structured Optimization*, in ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. IV-2-W5, Copernicus GmbH, May 2019, pp. 365–372, https://doi.org/10.5194/isprs-annals-IV-2-W5-365-2019. [Cited on pages 8 and 7.]

[113] K. Hammernik, T. Klatzer, E. Kobler, M. P. Recht, D. K. Sodickson, T. Pock, and F. Knoll, *Learning a variational network for reconstruction of accelerated MRI data*, Magn Reson Med, 79 (2018), pp. 3055–3071, https://doi.org/10.1002/mrm.26977. [Cited on pages 58, 76, 53, and 71.]

[114] K. Hammernik, T. Würfl, T. Pock, and A. Maier, *A Deep Learning Architecture for Limited-Angle Computed Tomography Reconstruction*, in Bildverarbeitung für die Medizin 2017, Informatik aktuell, Springer Berlin Heidelberg, 2017, pp. 92–97. [Cited on pages 58, 76, 53, and 71.]

[115] E. R. Hansen and G. W. Walster, *Global Optimization Using Interval Analysis*, no. 264 in Monographs and Textbooks in Pure and Applied Mathematics, Marcel Dekker, New York, 2nd ed., revised and expanded ed., 2004. [Cited on pages 27 and 25.]

[116] F. Hanzely, P. Richtarik, and L. Xiao, *Accelerated Bregman Proximal Gradient Methods for Relatively Smooth Convex Optimization*, arXiv:1808.03045 [math], (2018), https://arxiv.org/abs/1808.03045. [Cited on pages 43 and 40.]

[117] K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, arXiv:1512.03385 [cs], (2015), https://arxiv.org/abs/1512.03385. [Cited on pages 76, 120, 124, 71, and 116.]

[118] W. Heng, S. Zhou, and T. Jiang, *Harmonic Adversarial Attack Method*, arXiv:1807.10590 [cs], (2018), https://arxiv.org/abs/1807.10590. [Cited on pages 126 and 121.]

[119] M. Hintermüller and C. N. Rautenberg, *Optimal Selection of the Regularization Function in a Weighted Total Variation Model. Part I: Modelling and Theory*, J Math Imaging Vis, 59 (2017), pp. 498–514, https://doi.org/10.1007/s10851-017-0744-2. [Cited on pages 57, 75, 52, and 70.]

[120] M. Hintermüller, C. N. Rautenberg, T. Wu, and A. Langer, *Optimal Selection of the Regularization Function in a Weighted Total Variation Model. Part II: Algorithm, Its Analysis and Numerical Tests*, J Math Imaging Vis, 59 (2017), pp. 515–533, https://doi.org/10.1007/s10851-017-0736-2. [Cited on pages 57, 75, 52, and 70.]

[121] G. E. Hinton, *Training Products of Experts by Minimizing Contrastive Divergence*, Neural Computation, 14 (2002), pp. 1771–1800, https://doi.org/10.1162/089976602760128018. [Cited on page 2.]

[122] S. Hong, V. Chandrasekaran, Y. Kaya, T. Dumitraş, and N. Papernot, *On the Effectiveness of Mitigating Data Poisoning Attacks with Gradient Shaping*, arXiv:2002.11497 [cs], (2020), https://arxiv.org/abs/2002.11497. [Cited on pages 120, 122, 129, 115, 118, and 124.]

[123] B. K. P. Horn and B. G. Schunck, *Determining optical flow*, Artificial Intelligence, 17 (1981), pp. 185–203, https://doi.org/10.1016/0004-3702(81)90024-2. [Cited on pages 6, 7, and 5.]

[124] R. Hu, Y. Guo, M. Pan, and Y. Gong, *Targeted Poisoning Attacks on Social Recommender Systems*, in 2019 IEEE Global Communications Conference (GLOBECOM), Dec. 2019, pp. 1–6, https://doi.org/10.1109/GLOBECOM38437.2019.9013539. [Cited on pages 112 and 108.]

[125] J. Huang and D. Mumford, *Statistics of natural images and models*, in Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), vol. 1, June 1999, pp. 541–547 Vol. 1, https://doi.org/10.1109/CVPR.1999.786990. [Cited on pages 74 and 69.]

[126] W. R. Huang, J. Geiping, L. Fowl, G. Taylor, and T. Goldstein, *MetaPoison: Practical General-purpose Clean-label Data Poisoning*, in Advances in Neural Information Processing Systems, vol. 33, Vancouver, Canada, Dec. 2020. [Cited on pages 113, 114, 116, 118, 119, 126, 128, 131, 160, 109, 110, 112, 115, 121, 122, 125, and 152.]

[127] D. R. Hunter and K. Lange, *A Tutorial on MM Algorithms*, The American Statistician, 58 (2004), pp. 30–37, https://doi.org/10.1198/0003130042836. [Cited on pages 24, 64, 22, and 59.]

[128] W. Huyer and A. Neumaier, *Global Optimization by Multilevel Coordinate Search*, Journal of Global Optimization, 14 (1999), pp. 331–355, https://doi.org/10.1023/A:1008382309369. [Cited on pages 44 and 41.]

[129] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, *FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks*, in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017, pp. 1647–1655, https://doi.org/10.1109/CVPR.2017.179. [Cited on pages 56 and 51.]

[130] L. Ingber, *Adaptive simulated annealing (ASA): Lessons learned*, Control Cybernetics, 25 (1996), pp. 33–54. [Cited on pages 46 and 44.]

[131] A. Ioffe, *An Invitation to Tame Optimization*, SIAM J. Optim., 19 (2009), pp. 1894–1917, https://doi.org/10.1137/080722059. [Cited on pages 24 and 22.]

[132] H. Ishikawa and D. Geiger, *Occlusions, discontinuities, and epipolar lines in stereo*, in Computer Vision — ECCV'98, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1998, pp. 232–248. [Cited on pages 6 and 5.]

[133] H. Ishikawa and D. Geiger, *Segmentation by Grouping Junctions*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR '98, Washington, DC, USA, 1998, IEEE Computer Society, pp. 125–. [Cited on pages 27, 41, 25, and 39.]

[134] J.-H. Jacobsen, A. Smeulders, and E. Oyallon, *I-RevNet: Deep Invertible Networks*, arXiv:1802.07088 [cs, stat], (2018), https://arxiv.org/abs/1802.07088. [Cited on pages 88, 97, 84, and 93.]

[135] M. Jaderberg, K. Simonyan, A. Zisserman, and k. kavukcuoglu, *Spatial Transformer Networks*, in Advances in Neural Information Processing Systems 28, Curran Associates, Inc., 2015, pp. 2017–2025. [Cited on pages 116 and 112.]

[136] B. Jayaraman and D. Evans, *Evaluating Differentially Private Machine Learning in Practice*, arXiv:1902.08874 [cs, stat], (2019), https://arxiv.org/abs/1902.08874. [Cited on pages 96 and 92.]

[137] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, *Deep Convolutional Neural Network for Inverse Problems in Imaging*, IEEE Transactions on Image Processing, 26 (2017), pp. 4509–4522, https://doi.org/10.1109/TIP.2017.2713099. [Cited on pages 65 and 60.]

[138] T. Joachims, T. Finley, and C.-N. J. Yu, *Cutting-plane training of structural SVMs*, Mach Learn, 77 (2009), pp. 27–59, https://doi.org/10.1007/s10994-009-5108-8. [Cited on pages 79 and 74.]

[139] A. Jochems, T. M. Deist, I. El Naqa, M. Kessler, C. Mayo, J. Reeves, S. Jolly, M. Matuszak, R. Ten Haken, J. van Soest, C. Oberije, C. Faivre-Finn, G. Price, D. de Ruysscher, P. Lambin, and A. Dekker, *Developing and Validating a Survival Prediction Model for NSCLC Patients Through Distributed Learning Across 3 Countries*, International Journal of Radiation Oncology*Biology*Physics, 99 (2017), pp. 344–352, https://doi.org/10.1016/j.ijrobp.2017.04.021. [Cited on pages 87 and 83.]

[140] A. Jochems, T. M. Deist, J. van Soest, M. Eble, P. Bulens, P. Coucke, W. Dries, P. Lambin, and A. Dekker, *Distributed learning: Developing a predictive model based on data from multiple hospitals without data leaving the hospital – A real life proof of concept*, Radiotherapy and Oncology, 121 (2016), pp. 459–467, https://doi.org/10.1016/j.radonc.2016.10.002. [Cited on pages 86 and 82.]

[141] E. Kang, J. Min, and J. C. Ye, *A deep convolutional neural network using directional wavelets for low-dose X-ray CT reconstruction*, Medical Physics, 44 (2017), pp. e360–e375, https://doi.org/10.1002/mp.12344. [Cited on pages 65 and 60.]

[142] D. Karmon, D. Zoran, and Y. Goldberg, *LaVAN: Localized and Visible Adversarial Noise*, arXiv:1801.02608 [cs], (2018), https://arxiv.org/abs/1801.02608. [Cited on pages 126 and 121.]

[143] R. B. Kearfott, *Rigorous Global Search: Continuous Problems*, Springer Science & Business Media, Mar. 2013. [Cited on pages 27 and 25.]

[144] J. Kennedy and R. C. Eberhardt, *Particle Swarm Optimization*, Proceedings of the 1995 IEEE International Conference on Neural Networks (Conference proceedings), 4 (1995), pp. 1942–1948. [Cited on page 46.]

[145] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, in International Conference on Learning Representations (ICLR), San Diego, May 2015, https://arxiv.org/abs/1412.6980. [Cited on pages 46, 74, 90, 156, 42, 43, 69, 86, and 148.]

[146] T. Klatzer, K. Hammernik, P. Knobelreiter, and T. Pock, *Learning joint demosaicing and denoising based on sequential energy minimization*, in 2016 IEEE International Conference on Computational Photography (ICCP), May 2016, pp. 1–11, https://doi.org/10.1109/ICCPHOT.2016.7492871. [Cited on pages 58, 76, 53, and 71.]

[147] P. Knobelreiter, C. Reinbacher, A. Shekhovtsov, and T. Pock, *End-to-End Training of Hybrid CNN-CRF Models for Stereo*, in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, July 2017, IEEE, pp. 1456–1465, https://doi.org/10.1109/CVPR.2017.159. [Cited on pages 58, 76, 53, and 71.]

[148] P. W. Koh and P. Liang, *Understanding Black-box Predictions via Influence Functions*, in International Conference on Machine Learning, July 2017, pp. 1885–1894. [Cited on pages 89, 113, 122, 85, 108, and 117.]

[149] P. W. Koh, J. Steinhardt, and P. Liang, *Stronger Data Poisoning Attacks Break Data Sanitization Defenses*, arXiv:1811.00741 [cs, stat], (2018), https://arxiv.org/abs/1811.00741. [Cited on pages 113 and 108.]

[150] A. Kolb, E. Barth, R. Koch, and R. Larsen, *Time-of-Flight Cameras in Computer Graphics*, in Computer Graphics Forum, vol. 29, Wiley Online Library, 2010, pp. 141–159. [Cited on pages 21 and 19.]

[151] K. Kolev and D. Cremers, *Integration of Multiview Stereo and Silhouettes Via Convex Functionals on Convex Domains*, in Computer Vision – ECCV 2008, Lecture Notes in Computer Science, Berlin, Heidelberg, 2008, Springer, pp. 752–765, https://doi.org/10.1007/978-3-540-88682-2_57. [Cited on pages 7 and 6.]

[152] V. Kolmogorov and R. Zabin, *What energy functions can be minimized via graph cuts?*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26 (2004), pp. 147–159, https://doi.org/10.1109/TPAMI.2004.1262177. [Cited on pages 12, 27, 41, 11, 25, and 39.]

[153] C. D. Kolstad and L. S. Lasdon, *Derivative evaluation and computational experience with large bilevel mathematical programs*, J Optim Theory Appl, 65 (1990), pp. 485–499, https://doi.org/10.1007/BF00939562. [Cited on pages 57, 75, 52, and 70.]

[154] J. Konečný, B. McMahan, and D. Ramage, *Federated Optimization:Distributed Optimization Beyond the Datacenter*, arXiv:1511.03575 [cs, math], (2015), https://arxiv.org/abs/1511.03575. [Cited on pages 86 and 82.]

[155] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, in Advances in Neural Information Processing Systems, 2012, pp. 1097–1105. [Cited on pages 56, 98, 123, 51, 94, and 119.]

[156] R. S. S. Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, *Adversarial Machine Learning-Industry Perspectives*, in 2020 IEEE Security and Privacy Workshops (SPW), May 2020, pp. 69–75, https://doi.org/10.1109/SPW50608.2020.00028. [Cited on pages 112 and 108.]

[157] K. Kunisch and T. Pock, *A Bilevel Optimization Approach for Parameter Learning in Variational Models*, SIAM Journal on Imaging Sciences, 6 (2013), pp. 938–983, https://doi.org/10.1137/120882706. [Cited on pages 57, 75, 80, 52, and 70.]

[158] G. Kuschk and D. Cremers, *Fast and Accurate Large-Scale Stereo Reconstruction Using Variational Methods*, in 2013 IEEE International Conference on Computer Vision Workshops, Dec. 2013, pp. 700–707, https://doi.org/10.1109/ICCVW.2013.96. [Cited on pages 44 and 41.]

[159] L. Landrieu and G. Obozinski, *Cut Pursuit: Fast algorithms to learn piecewise constant functions*, in Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, vol. 51 of Proceedings of Machine Learning Research, Cadiz, Spain, Apr. 2016, PMLR, pp. 1384–1393. [Cited on pages 7, 16, 17, 18, 6, and 15.]

[160] L. Landrieu and G. Obozinski, *Cut Pursuit: Fast Algorithms to Learn Piecewise Constant Functions on General Weighted Graphs*, SIAM J. Imaging Sci., 10 (2017), pp. 1724–1766, https://doi.org/10.1137/17M1113436. [Cited on pages 7 and 6.]

[161] L. Landrieu and M. Simonovsky, *Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs*, arXiv:1711.09869 [cs], (2017), https://arxiv.org/abs/1711.09869. [Cited on pages 8 and 7.]

[162] R. Lange, *3D Time-of-Flight Distance Measurement with Custom Solid-State Image Sensors in CMOS/CCD-Technology*, PhD thesis, University of Siegen, Siegen, June 2000. [Cited on pages 48, 49, 156, 44, 45, 46, and 148.]

[163] M. Larsson, A. Arnab, F. Kahl, S. Zheng, and P. Torr, *A Projected Gradient Descent Method for CRF Inference Allowing End-to-End Training of Arbitrary Pairwise Potentials*, in Energy Minimization Methods in Computer Vision and Pattern Recognition, Lecture Notes in Computer Science, Springer International Publishing, 2018, pp. 564–579. [Cited on pages 58, 76, 53, and 71.]

[164] M. Larsson, A. Arnab, S. Zheng, P. Torr, and F. Kahl, *Revisiting Deep Structured Models for Pixel-Level Labeling with Gradient-Based Inference*, SIAM J. Imaging Sci., (2018), pp. 2610–2628, https://doi.org/10.1137/18M1167267. [Cited on pages 58, 76, 53, and 71.]

[165] E. Laude, T. Möllenhoff, M. Moeller, J. Lellmann, and D. Cremers, *Sublabel-Accurate Convex Relaxation of Vectorial Multilabel Energies*, in Computer Vision – ECCV 2016, Lecture Notes in Computer Science, Springer, Cham, Oct. 2016, pp. 614–627, https://doi.org/10.1007/978-3-319-46448-0_37. [Cited on pages 7, 50, 6, and 46.]

[166] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436–444, https://doi.org/10.1038/nature14539. [Cited on pages 56, 112, 51, and 108.]

[167] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, *A tutorial on energy-based learning*, Predicting structured data, 1 (2006). [Cited on pages 59, 60, 77, 81, 54, 55, 72, and 76.]

[168] Y. LeCun and F. J. Huang, *Loss functions for discriminative training of energy-based models*, in AISTATS 2005 - Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics, 2005, pp. 206–213. [Cited on pages 2, 59, 60, 77, 81, 54, 55, 72, and 76.]

[169] J. Lellmann, J. Kappes, J. Yuan, F. Becker, and C. Schnörr, *Convex Multi-class Image Labeling by Simplex-Constrained Total Variation*, in Scale Space and Variational Methods in Computer Vision, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 150–162. [Cited on pages 7 and 6.]

[170] J. Lellmann, E. Strekalovskiy, S. Koetter, and D. Cremers, *Total Variation Regularization for Functions with Values in a Manifold*, in Proceedings of the 2013 IEEE International Conference on Computer Vision, ICCV '13, Washington, DC, USA, 2013, IEEE Computer Society, pp. 2944–2951, https://doi.org/10.1109/ICCV.2013.366. [Cited on pages 7 and 6.]

[171] A. S. Lewis and S. J. Wright, *A proximal method for composite minimization*, Math. Program., 158 (2016), pp. 501–546, https://doi.org/10.1007/s10107-015-0943-9. [Cited on pages 25, 77, 23, and 72.]

[172] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik, *Data Poisoning Attacks on Factorization-Based Collaborative Filtering*, in Advances in Neural Information Processing Systems 29, Curran Associates, Inc., 2016, pp. 1885–1893. [Cited on pages 112 and 108.]

[173] G. Lin, C. Shen, A. van den Hengel, and I. Reid, *Efficient Piecewise Training of Deep Structured Models for Semantic Segmentation*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3194–3203. [Cited on pages 58, 76, 53, and 71.]

[174] M. Lindner and A. Kolb, *Lateral and depth calibration of pmd-distance sensors*, in International Symposium on Visual Computing, Springer, 2006, pp. 524–533. [Cited on pages 48 and 44.]

[175] C. K. Liu, A. Hertzmann, and Z. Popovic, *Learning Physics-Based Motion Style with Nonlinear Inverse Optimization*, ACM Trans. Graph, 24 (2005), pp. 1071–1081. [Cited on pages 77 and 72.]

[176] D. C. Liu and J. Nocedal, *On the limited memory BFGS method for large scale optimization*, Mathematical Programming, 45 (1989), pp. 503–528, https://doi.org/10.1007/BF01589116. [Cited on pages 88 and 84.]

[177] Y. Liu, X. Chen, C. Liu, and D. Song, *Delving into Transferable Adversarial Examples and Black-box Attacks*, arXiv:1611.02770 [cs], (2017), https://arxiv.org/abs/1611.02770. [Cited on pages 116 and 112.]

[178] J. Long, E. Shelhamer, and T. Darrell, *Fully Convolutional Networks for Semantic Segmentation*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440. [Cited on pages 56 and 51.]

[179] G. Lovisotto, S. Eberz, and I. Martinovic, *Biometric Backdoors: A Poisoning Attack Against Unsupervised Template Updating*, arXiv:1905.09162 [cs], (2019), https://arxiv.org/abs/1905.09162. [Cited on pages 112 and 108.]

[180] H. Lu, R. M. Freund, and Y. Nesterov, *Relatively Smooth Convex Optimization by First-Order Methods, and Applications*, SIAM J. Optim., (2018), pp. 333–354, https://doi.org/10.1137/16M1099546. [Cited on pages 60 and 55.]

[181] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, Springer, New York, NY, 4th ed. 2016 edition ed., June 2015. [Cited on pages 44, 53, 41, and 49.]

[182] N. Lukas, Y. Zhang, and F. Kerschbaum, *Deep Neural Network Fingerprinting by Conferrable Adversarial Examples*, arXiv:1912.00888 [cs, stat], (2020), https://arxiv.org/abs/1912.00888. [Cited on pages 112 and 108.]

[183] Y. Ma, X. Zhu, and J. Hsu, *Data Poisoning against Differentially-Private Learners: Attacks and Defenses*, arXiv:1903.09860 [cs], (2019), https://arxiv.org/abs/1903.09860. [Cited on pages 120 and 115.]

[184] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, *Towards Deep Learning Models Resistant to Adversarial Attacks*, arXiv:1706.06083 [cs, stat], (2017), https://arxiv.org/abs/1706.06083. [Cited on pages 90, 113, 120, 86, 109, and 115.]

[185] A. Mahendran and A. Vedaldi, *Visualizing Deep Convolutional Neural Networks Using Natural Pre-images*, International Journal of Computer Vision, 120 (2016), pp. 233–255, https://doi.org/10.1007/s11263-016-0911-8. [Cited on pages 88, 89, 84, and 85.]

[186] J. Mairal, *Optimization with First-order Surrogate Functions*, in Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28, ICML'13, Atlanta, GA, USA, 2013, JMLR.org, pp. III–783–III–791. [Cited on pages 24, 64, 22, and 59.]

[187] J. Mairal, *Incremental Majorization-Minimization Optimization with Application to Large-Scale Machine Learning*, SIAM Journal on Optimization, 25 (2015), pp. 829–855, https://doi.org/10.1137/140957639. [Cited on pages 64 and 59.]

[188] D. Martin, C. Fowlkes, D. Tal, and J. Malik, *A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics*, in Proceedings of 8th International Conference on Computer Vision, vol. 2, July 2001, pp. 416–423. [Cited on pages 18, 68, 74, 78, 157, 17, 63, 69, 73, and 149.]

[189] A. F. T. Martins, N. A. Smith, and E. P. Xing, *Polyhedral outer approximations with application to natural language parsing*, in Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09, Montreal, Quebec, Canada, 2009, ACM Press, pp. 1–8, https://doi.org/10.1145/1553374.1553466. [Cited on pages 58, 76, 53, and 71.]

[190] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, *A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation*, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2016), pp. 4040–4048, https://doi.org/10.1109/CVPR.2016.438, https://arxiv.org/abs/1512.02134. [Cited on pages 56 and 51.]

[191] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, *Communication-Efficient Learning of Deep Networks from Decentralized Data*, arXiv:1602.05629 [cs], (2017), https://arxiv.org/abs/1602.05629. [Cited on pages 86, 93, 82, and 89.]

[192] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, *Learning Differentially Private Recurrent Language Models*, arXiv:1710.06963 [cs], (2018), https://arxiv.org/abs/1710.06963. [Cited on pages 93 and 89.]

[193] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, *Exploiting Unintended Feature Leakage in Collaborative Learning*, in 2019 IEEE Symposium on Security and Privacy (SP), May 2019, pp. 691–706, https://doi.org/10.1109/SP.2019.00029. [Cited on pages 88 and 84.]

[194] J. H. Metzen, M. C. Kumar, T. Brox, and V. Fischer, *Universal Adversarial Perturbations Against Semantic Image Segmentation*, in 2017 IEEE International Conference on Computer Vision (ICCV), Oct. 2017, pp. 2774–2783, https://doi.org/10.1109/ICCV.2017.300. [Cited on pages 56 and 51.]

[195] J. Mockus, *Bayesian Approach to Global Optimization: Theory and Applications*, Mathematics and Its Applications, Springer Netherlands, 1989. [Cited on pages 75 and 70.]

[196] T. Möllenhoff and D. Cremers, *Sublabel-Accurate Discretization of Nonconvex Free-Discontinuity Problems*, Proceedings of the IEEE International Conference on Computer Vision, (2017), pp. 1183–1191, https://doi.org/10.1109/ICCV.2017.134. [Cited on pages 7, 8, 9, 27, 6, and 25.]

[197] T. Möllenhoff, E. Laude, M. Moeller, J. Lellmann, and D. Cremers, *Sublabel-Accurate Relaxation of Nonconvex Energies*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3948–3956, https://doi.org/10.1109/CVPR.2016.428. [Cited on pages 7, 13, 14, 18, 27, 48, 155, 6, 12, 17, 25, 45, and 147.]

[198] M. Mosbach, M. Andriushchenko, T. Trost, M. Hein, and D. Klakow, *Logit Pairing Methods Can Fool Gradient-Based Attacks*, arXiv:1810.12042 [cs, stat], (2019), https://arxiv.org/abs/1810.12042. [Cited on pages 116 and 112.]

[199] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, *Towards Poisoning of Deep Learning Algorithms with Back-gradient Optimization*, in Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec '17, New York, NY, USA, 2017, ACM, pp. 27–38, https://doi.org/10.1145/3128572.3140451. [Cited on pages 113, 114, 108, and 110.]

[200] L. Muñoz-González, B. Pfitzner, M. Russo, J. Carnerero-Cano, and E. C. Lupu, *Poisoning Attacks with Generative Adversarial Nets*, arXiv:1906.07773 [cs, stat], (2019), https://arxiv.org/abs/1906.07773. [Cited on pages 122 and 118.]

[201] H. Mühlenbein, M. Schomisch, and J. Born, *The parallel genetic algorithm as function optimizer*, Parallel Computing, 17 (1991), pp. 619–632, https://doi.org/10.1016/S0167-8191(05)80052-3. [Cited on pages 43, 44, 156, 40, 41, and 148.]

[202] D. Mumford and J. Shah, *Optimal approximations by piecewise smooth functions and associated variational problems*, Communications on pure and applied mathematics, 42 (1989), pp. 577–685. [Cited on pages 6, 7, and 5.]

[203] R. Munroe, *A Bunch of Rocks*. https://xkcd.com/505/, Nov. 2008. [Cited on page viii.]

[204] Y. Nesterov, *Introductory Lectures on Convex Programming - Volume I: Basic Course*, July 1998. [Cited on pages 52 and 48.]

[205] Y. Nesterov, *Introductory Lectures on Convex Optimization*, vol. 87 of Applied Optimization, Springer US, Boston, MA, 2004. [Cited on pages 22 and 20.]

[206] Y. Nesterov, *Gradient methods for minimizing composite functions*, Math. Program., 140 (2013), pp. 125–161, https://doi.org/10.1007/s10107-012-0629-5. [Cited on pages 24 and 22.]

[207] C. Nieuwenhuis, E. Töppe, and D. Cremers, *A Survey and Comparison of Discrete and Continuous Multi-label Optimization Approaches for the Potts Model*, Int J Comput Vis, 104 (2013), pp. 223–240, https://doi.org/10.1007/s11263-013-0619-y. [Cited on pages 11, 65, 155, 10, 60, and 147.]

[208] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operations Research, Springer, New York, 2nd ed ed., 2006. [Cited on pages 116 and 112.]

[209] S. Nowozin and C. H. Lampert, *Structured Learning and Prediction in Computer Vision*, Found. Trends. Comput. Graph. Vis., 6 (2011), pp. 185–365, https://doi.org/10.1561/0600000033. [Cited on pages 76, 77, 80, 71, 72, and 75.]

[210] P. Ochs, *Unifying abstract inexact convergence theorems for descent methods and block coordinate variable metric iPiano*, arXiv:1602.07283 [math], (2016), https://arxiv.org/abs/1602.07283. [Cited on pages 36, 43, 34, and 40.]

[211] P. Ochs, Y. Chen, T. Brox, and T. Pock, *iPiano: Inertial Proximal Algorithm for Nonconvex Optimization*, SIAM J. Imaging Sci., 7 (2014), pp. 1388–1419, https://doi.org/10.1137/130942954. [Cited on pages 43 and 40.]

[212] P. Ochs, A. Dosovitskiy, T. Brox, and T. Pock, *An Iterated L1 Algorithm for Non-smooth Non-convex Optimization in Computer Vision*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, June 2013, pp. 1759–1766, https://doi.org/10.1109/CVPR.2013.230. [Cited on pages 40 and 38.]

[213] P. Ochs, A. Dosovitskiy, T. Brox, and T. Pock, *On Iteratively Reweighted Algorithms for Nonsmooth Nonconvex Optimization in Computer Vision*, SIAM J. Imaging Sci., 8 (2015), pp. 331–372, https://doi.org/10.1137/140971518. [Cited on pages 25, 40, 23, and 38.]

[214] P. Ochs, J. Fadili, and T. Brox, *Non-smooth Non-convex Bregman Minimization: Unification and new Algorithms*, arXiv:1707.02278 [cs, math], (2017), https://arxiv.org/abs/1707.02278. [Cited on pages 25, 30, 41, 22, 23, 27, and 38.]

[215] P. Ochs, R. Ranftl, T. Brox, and T. Pock, *Bilevel Optimization with Nonsmooth Lower Level Problems*, in Scale Space and Variational Methods in Computer Vision, Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 654–665. [Cited on pages 58, 76, 53, and 71.]

[216] P. Ochs, R. Ranftl, T. Brox, and T. Pock, *Techniques for Gradient-Based Bilevel Optimization with Non-smooth Lower Level Problems*, J Math Imaging Vis, 56 (2016), pp. 175–194, https://doi.org/10.1007/s10851-016-0663-7. [Cited on pages 74, 76, 84, 69, 71, and 79.]

[217] N. Papernot, *A Marauder's Map of Security and Privacy in Machine Learning*, arXiv:1811.01134 [cs], (2018), https://arxiv.org/abs/1811.01134. [Cited on pages 112 and 108.]

[218] N. Papernot and P. McDaniel, *Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning*, arXiv:1803.04765 [cs, stat], (2018), https://arxiv.org/abs/1803.04765. [Cited on pages 114 and 110.]

[219] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, *Automatic differentiation in PyTorch*, in NIPS 2017 Autodiff Workshop, Long Beach, CA, 2017. [Cited on pages 74 and 69.]

[220] A. Paudice, L. Muñoz-González, A. Gyorgy, and E. C. Lupu, *Detection of Adversarial Training Examples in Poisoning Attacks through Anomaly Detection*, arXiv:1802.03041 [cs, stat], (2018), https://arxiv.org/abs/1802.03041. [Cited on pages 119, 128, 115, and 122.]

[221] A. Paudice, L. Muñoz-González, and E. C. Lupu, *Label Sanitization Against Label Flipping Poisoning Attacks*, in ECML PKDD 2018 Workshops, Lecture Notes in Computer Science, Cham, 2019, Springer International Publishing, pp. 5–15, https://doi.org/10.1007/978-3-030-13453-2_1. [Cited on pages 114 and 109.]

[222] E. Pauwels, *The value function approach to convergence analysis in composite optimization*, Operations Research Letters, 44 (2016), pp. 790–795, https://doi.org/10.1016/j.orl.2016.10.003. [Cited on pages 25 and 23.]

[223] N. Peri, N. Gupta, W. R. Huang, L. Fowl, C. Zhu, S. Feizi, T. Goldstein, and J. P. Dickerson, *Deep k-NN Defense Against Clean-Label Data Poisoning Attacks*, in Computer Vision – ECCV 2020 Workshops, Lecture Notes in Computer Science, Cham, 2020, Springer International Publishing, pp. 55–70, https://doi.org/10.1007/978-3-030-66415-2_4. [Cited on pages 119, 128, 115, and 122.]

[224] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, *Privacy-Preserving Deep Learning: Revisited and Enhanced*, in Applications and Techniques in Information Security, Communications in Computer and Information Science, Singapore, 2017, Springer, pp. 100–110, https://doi.org/10.1007/978-981-10-5421-1_9. [Cited on pages 88, 89, 95, 84, 85, and 91.]

[225] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, *Privacy-Preserving Deep Learning via Additively Homomorphic Encryption*, Tech. Report 715, 2017. [Cited on pages 88, 95, 84, and 91.]

[226] T. Pock and A. Chambolle, *Diagonal preconditioning for first order primal-dual algorithms in convex optimization*, in 2011 International Conference on Computer Vision, Nov. 2011, pp. 1762–1769, https://doi.org/10.1109/ICCV.2011.6126441. [Cited on pages 18 and 17.]

[227] T. Pock, A. Chambolle, D. Cremers, and H. Bischof, *A convex relaxation approach for computing minimal partitions*, in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference On, IEEE, 2009, pp. 810–817. [Cited on pages 12 and 11.]

[228] T. Pock, D. Cremers, H. Bischof, and A. Chambolle, *Global Solutions of Variational Models with Convex Regularization*, SIAM J. Imaging Sci., 3 (2010), pp. 1122–1145, https://doi.org/10.1137/090757617. [Cited on pages 7, 13, 22, 27, 41, 42, 6, 12, 20, 25, and 39.]

[229] T. Pock, T. Schoenemann, G. Graber, H. Bischof, and D. Cremers, *A convex formulation of continuous multi-label problems*, in European Conference on Computer Vision, Springer, 2008, pp. 792–805. [Cited on pages 7 and 6.]

[230] R. B. Potts, *Some generalized order-disorder transformations*, in Mathematical Proceedings of the Cambridge Philosophical Society, vol. 48, Cambridge Univ Press, 1952, pp. 106–109. [Cited on pages 7 and 6.]

[231] R. Precup, *Methods in Nonlinear Integral Equations*, Springer Netherlands, 2002. [Cited on pages 39 and 36.]

[232] C. Qin, J. Martens, S. Gowal, D. Krishnan, K. Dvijotham, A. Fawzi, S. De, R. Stanforth, and P. Kohli, *Adversarial Robustness through Local Linearization*, arXiv:1907.02610 [cs, stat], (2019), https://arxiv.org/abs/1907.02610. [Cited on pages 116 and 112.]

[233] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof, *Pushing the limits of stereo using variational stereo estimation*, in Intelligent Vehicles Symposium (IV), 2012 IEEE, IEEE, 2012, pp. 401–407. [Cited on pages 6, 7, and 5.]

[234] R. Ranftl, T. Pock, and H. Bischof, *Minimizing TGV-Based Variational Models with Non-convex Data Terms*, in Scale Space and Variational Methods in Computer Vision, Lecture Notes in Computer Science, Springer Berlin Heidelberg, June 2013, pp. 282–293, https://doi.org/10.1007/978-3-642-38267-3_24. [Cited on pages 7 and 6.]

[235] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, *Maximum Margin Planning*, in Proceedings of the 23rd International Conference on Machine Learning, ICML '06, New York, NY, USA, 2006, ACM, pp. 729–736, https://doi.org/10.1145/1143844.1143936. [Cited on pages 76 and 71.]

[236] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, *Adaptive Federated Optimization*, arXiv:2003.00295 [cs, math, stat], (2020), https://arxiv.org/abs/2003.00295. [Cited on pages 93 and 89.]

[237] S. Reich and S. Sabach, *Existence and Approximation of Fixed Points of Bregman Firmly Nonexpansive Mappings in Reflexive Banach Spaces*, in Fixed-Point Algorithms for Inverse Problems in Science and Engineering, Springer Optimization and Its Applications, Springer, New York, NY, 2011, pp. 301–316, https://doi.org/10.1007/978-1-4419-9569-8_15. [Cited on pages 61, 69, 56, and 64.]

[238] G. Riegler, M. Rüther, and H. Bischof, *Atgv-net: Accurate depth super-resolution*, in European Conference on Computer Vision, Springer, 2016, pp. 268–284. [Cited on pages 58, 76, 53, and 71.]

[239] L. Ritschl, F. Bergner, C. Fleischmann, and M. K. s, *Improved total variation-based CT image reconstruction applied to clinical data*, Phys. Med. Biol., 56 (2011), pp. 1545–1561, https://doi.org/10.1088/0031-9155/56/6/003. [Cited on pages 56 and 51.]

[240] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, N.J, 1970. [Cited on pages 28, 66, 26, and 61.]

[241] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*, vol. 317 of Grundlehren Der Mathematischen Wissenschaften, Springer-Verlag Berlin Heidelberg, Berlin Heidelberg, 3rd ed., June 2009, https://doi.org/10.1007/978-3-642-02431-3. [Cited on pages 32, 34, 29, 30, and 31.]

[242] T. Rohlfing, C. R. Maurer, D. A. Bluemke, and M. A. Jacobs, *Volume-preserving nonrigid registration of MR breast images using free-form deformation with an incompressibility constraint*, IEEE Transactions on Medical Imaging, 22 (2003), pp. 730–741, https://doi.org/10.1109/TMI.2003.814791. [Cited on pages 56 and 51.]

[243] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional Networks for Biomedical Image Segmentation*, in Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 234–241. [Cited on pages 56 and 51.]

[244] F. Rosenblatt, *The perceptron: A probabilistic model for information storage and organization in the brain*, Psychological Review, 65 (1958), pp. 386–408, https://doi.org/10.1037/h0042519. [Cited on pages 60, 77, 55, and 72.]

[245] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Physica D: Nonlinear Phenomena, 60 (1992), pp. 259–268, https://doi.org/10.1016/0167-2789(92)90242-F. [Cited on pages 2, 7, 16, 41, 42, 56, 68, 89, 6, 15, 39, 51, 63, and 85.]

[246] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, *ImageNet Large Scale Visual Recognition Challenge*, Int J Comput Vis, 115 (2015), pp. 211–252, https://doi.org/10.1007/s11263-015-0816-y. [Cited on pages 112, 124, 160, 108, 119, and 152.]

[247] S. Sabach and S. Shtern, *A First Order Method for Solving Convex Bilevel Optimization Problems*, SIAM J. Optim., 27 (2017), pp. 640–660, https://doi.org/10.1137/16M105592X. [Cited on pages 80 and 75.]

[248] A. Saha, A. Subramanya, and H. Pirsiavash, *Hidden Trigger Backdoor Attacks*, AAAI, 34 (2020), pp. 11957–11965, https://doi.org/10.1609/aaai.v34i07.6871, https://arxiv.org/abs/1910.00033. [Not cited.]

[249] K. G. G. Samuel and M. F. Tappen, *Learning optimized MAP estimates in continuously-valued MRF models*, in IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, June 2009, IEEE, https://doi.org/10.1109/CVPRW.2009.5206774. [Cited on pages 57 and 52.]

[250] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, *MobileNetV2: Inverted Residuals and Linear Bottlenecks*, arXiv:1801.04381 [cs], (2018), https://arxiv.org/abs/1801.04381. [Cited on pages 120, 124, and 116.]

[251] G. Savard and J. Gauvin, *The steepest descent direction for the nonlinear bilevel programming problem*, Operations Research Letters, 15 (1994), pp. 265–272, https://doi.org/10.1016/0167-6377(94)90086-8. [Cited on pages 57, 75, 52, and 70.]

[252] A. M. Saxe, J. L. McClelland, and S. Ganguli, *Exact solutions to the nonlinear dynamics of learning in deep linear neural networks*, arXiv:1312.6120 [cond-mat, q-bio, stat], (2013), https://arxiv.org/abs/1312.6120. [Cited on pages 74 and 69.]

[253] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, *High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth*, in Pattern Recognition, Lecture Notes in Computer Science, Springer, Cham, Sept. 2014, pp. 31–42, https://doi.org/10.1007/978-3-319-11752-2_3. [Cited on pages 11, 18, 47, 49, 155, 156, 10, 17, 44, 45, 147, and 148.]

[254] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen, *Variational Methods in Imaging*, no. 167 in Applied Mathematical Sciences, Springer, New York, first ed., 2009. [Cited on pages 38 and 35.]

[255] A. Schwarzschild, M. Goldblum, A. Gupta, J. P. Dickerson, and T. Goldstein, *Just How Toxic is Data Poisoning? A Unified Benchmark for Backdoor and Data Poisoning Attacks*, arXiv:2006.12557 [cs, stat], (2020), https://arxiv.org/abs/2006.12557. [Cited on pages 112, 119, 160, 108, 114, 115, and 152.]

[256] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, *Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks*, arXiv:1804.00792 [cs, stat], (2018), https://arxiv.org/abs/1804.00792. [Cited on pages 112, 113, 114, 115, 119, 122, 126, 131, 159, 108, 109, 110, 111, 118, 121, 127, and 151.]

[257] A. Shafahi, W. R. Huang, C. Studer, S. Feizi, and T. Goldstein, *Are adversarial examples inevitable?*, in ICLR 2019, New Orleans, Sept. 2018. [Cited on pages 56 and 51.]

[258] S. Shan, E. Wenger, J. Zhang, H. Li, H. Zheng, and B. Y. Zhao, *Fawkes: Protecting Personal Privacy against Unauthorized Deep Learning Models*, arXiv:2002.08327 [cs, stat], (2020), https://arxiv.org/abs/2002.08327. [Cited on pages 112, 122, 108, and 118.]

[259] J. Shen, X. Zhu, and D. Ma, *TensorClog: An Imperceptible Poisoning Attack on Deep Neural Network Applications*, IEEE Access, 7 (2019), pp. 41498–41506, https://doi.org/10.1109/ACCESS.2019.2905915. [Cited on pages 112 and 108.]

[260] R. Shokri and V. Shmatikov, *Privacy-Preserving Deep Learning*, in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15, Denver, Colorado, USA, 2015, ACM Press, pp. 1310–1321, https://doi.org/10.1145/2810103.2813687. [Cited on pages 86 and 82.]

[261] K. Simonyan and A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv:1409.1556 [cs], (2014), https://arxiv.org/abs/1409.1556. [Cited on pages 120, 124, and 116.]

[262] A. Sinha, P. Malo, and K. Deb, *A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications*, IEEE Transactions on Evolutionary Computation, 22 (2018), pp. 276–295, https://doi.org/10.1109/TEVC.2017.2712906. [Cited on pages 75 and 70.]

[263] A. Smola and T. Hofmann, *Exponential Families for Estimation*, Technical Report, (2003). [Cited on pages 83 and 78.]

[264] J. Snoek, H. Larochelle, and R. P. Adams, *Practical Bayesian Optimization of Machine Learning Algorithms*, in Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2, NIPS'12, USA, 2012, Curran Associates Inc., pp. 2951–2959. [Cited on pages 75 and 70.]

[265] D. Solans, B. Biggio, and C. Castillo, *Poisoning Attacks on Algorithmic Fairness*, arXiv:2004.07401 [cs.LG], (2020), https://arxiv.org/abs/2004.07401. [Cited on pages 112 and 108.]

[266] M. Solodov, *An Explicit Descent Method for Bilevel Convex Optimization*, Conv Ana, 14 (2007), pp. 227–237. [Cited on pages 80 and 75.]

[267] M. Souiai, M. R. Oswald, Y. Kee, J. Kim, M. Pollefeys, and D. Cremers, *Entropy Minimization for Convex Relaxation Approaches*, in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1778–1786. [Cited on pages 7 and 6.]

[268] J. Steinhardt, P. W. W. Koh, and P. S. Liang, *Certified Defenses for Data Poisoning Attacks*, in Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 3517–3529. [Cited on pages 112, 119, 128, 108, 115, and 122.]

[269] E. Strekalovskiy, A. Chambolle, and D. Cremers, *Convex Relaxation of Vectorial Problems with Coupled Regularization*, SIAM Journal on Imaging Sciences, 7 (2014), pp. 294–336, https://doi.org/10.1137/130908348. [Cited on pages 7, 50, 6, and 46.]

[270] E. Strekalovskiy and D. Cremers, *Real-Time Minimization of the Piecewise Smooth Mumford-Shah Functional*, in Computer Vision – ECCV 2014, Lecture Notes in Computer Science, Springer, Cham, Sept. 2014, pp. 127–141, https://doi.org/10.1007/978-3-319-10605-2_9. [Cited on pages 17 and 16.]

[271] E. Strekalovskiy, B. Goldluecke, and D. Cremers, *Tight convex relaxations for vector-valued labeling problems*, in 2011 International Conference on Computer Vision, Nov. 2011, pp. 2328–2335, https://doi.org/10.1109/ICCV.2011.6126514. [Cited on pages 7 and 6.]

[272] O. Suciu, R. Marginean, Y. Kaya, H. D. Iii, and T. Dumitras, *When Does Machine Learning {FAIL}? Generalized Transferability for Evasion and Poisoning Attacks*, in 27th {USENIX} Security Symposium ({USENIX} Security 18), 2018, pp. 1299–1316. [Cited on pages 112, 114, 108, and 109.]

[273] Y. Sun, P. Babu, and D. P. Palomar, *Majorization-Minimization Algorithms in Signal Processing, Communications, and Machine Learning*, IEEE Transactions on Signal Processing, 65 (2017), pp. 794–816, https://doi.org/10.1109/TSP.2016.2601299. [Cited on pages 24, 64, 22, and 59.]

[274] C. Sutton and A. McCallum, *An Introduction to Conditional Random Fields*, Foundations and Trends® in Machine Learning, 4 (2012), pp. 267–373, https://doi.org/10.1561/2200000013. [Cited on pages 76 and 71.]

[275] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, *Intriguing properties of neural networks*, in arXiv:1312.6199 [Cs], Dec. 2013, https://arxiv.org/abs/1312.6199. [Cited on pages 56, 90, 113, 51, 86, and 109.]

[276] M. Szummer, P. Kohli, and D. Hoiem, *Learning CRFs Using Graph Cuts*, in Computer Vision – ECCV 2008, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 582–595. [Cited on pages 76 and 71.]

[277] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, *DeepFace: Closing the Gap to Human-Level Performance in Face Verification*, in 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, June 2014, IEEE, pp. 1701–1708, https://doi.org/10.1109/CVPR.2014.220. [Cited on pages 112 and 108.]

[278] P. D. Tao and L. T. H. An, *Convex analysis approach to dc programming: Theory, algorithms and applications*, Acta Mathematica Vietnamica, 22 (1997), pp. 289–355. [Cited on pages 40 and 38.]

[279] M. F. Tappen, C. Liu, E. H. Adelson, and W. T. Freeman, *Learning Gaussian Conditional Random Fields for Low-Level Vision*, in 2007 IEEE Conference on Computer Vision and Pattern Recognition, June 2007, pp. 1–8, https://doi.org/10.1109/CVPR.2007.382979. [Cited on pages 60 and 55.]

[280] B. Taskar, *Learning Structured Prediction Models - A Large Margin Approach*, doctoral Thesis, Stanford University, Stanford, USA, Dec. 2004. [Cited on pages 76, 81, 83, 71, and 78.]

[281] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin, *Learning Structured Prediction Models: A Large Margin Approach*, in Proceedings of the 22nd International Conference on Machine Learning, ICML '05, New York, NY, USA, 2005, ACM, pp. 896–903, https://doi.org/10.1145/1102351.1102464. [Cited on pages 60, 61, 76, 78, 55, 56, 71, and 73.]

[282] B. Taskar, C. Guestrin, and D. Koller, *Max-Margin Markov Networks*, in Advances in Neural Information Processing Systems 16, MIT Press, 2004, pp. 25–32. [Cited on pages 58, 76, 81, 53, and 71.]

[283] B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning, *Max-Margin Parsing*, in Proceedings of EMNLP 2004, Barcelona, Spain, July 2004, Association for Computational Linguistics, pp. 1–8. [Cited on pages 76 and 71.]

[284] B. Taskar, S. Lacoste-Julien, and M. I. Jordan, *Structured Prediction, Dual Extragradient and Bregman Projections*, Journal of Machine Learning Research, 7 (2006), pp. 1627–1653. [Cited on pages 60, 61, 55, and 56.]

[285] M. Teboulle, *A simplified view of first order methods for optimization*, Math. Program., (2018), pp. 1–30, https://doi.org/10.1007/s10107-018-1284-2. [Cited on pages 60, 63, 55, and 58.]

[286] D. Tenbrinck, F. Gaede, and M. Burger, *Variational Graph Methods for Efficient Point Cloud Sparsification*, arXiv:1903.02858 [cs, math], (2019), https://arxiv.org/abs/1903.02858. [Cited on pages 5, 11, 12, 16, 17, 18, 4, 10, and 15.]

[287] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, *Large Margin Methods for Structured and Interdependent Output Variables*, Journal of Machine Learning Research, 6 (2005), pp. 1453–1484. [Cited on pages 58, 76, 78, 79, 81, 83, 53, 71, 73, and 74.]

[288] A. Turner, D. Tsipras, and A. Madry, *Clean-Label Backdoor Attacks*, openreview, (2018). [Cited on pages 113, 119, 109, and 115.]

[289] Z. Ugray, L. Lasdon, J. Plummer, F. Glover, J. Kelly, and R. Martí, *Scatter Search and Local NLP Solvers: A Multistart Framework for Global Optimization*, INFORMS Journal on Computing, 19 (2007), pp. 328–340, https://doi.org/10.1287/ijoc.1060.0175. [Cited on pages 46 and 44.]

[290] R. Uziel, M. Ronen, and O. Freifeld, *Bayesian Adaptive Superpixel Segmentation*, in Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 8470–8479. [Cited on pages 7 and 6.]

[291] T. Valkonen, *A primal–dual hybrid gradient method for nonlinear operators with applications to MRI*, Inverse Problems, 30 (2014), p. 055012, https://doi.org/10.1088/0266-5611/30/5/055012. [Cited on pages 50 and 46.]

[292] V. Vapnik, *Statistical Learning Theory. 1998*, vol. 3, Wiley, New York, 1998. [Cited on pages 59, 60, 63, 76, 78, 54, 55, 58, 71, and 73.]

[293] M. Veale, R. Binns, and L. Edwards, *Algorithms that remember: Model inversion attacks and data protection law*, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 376 (2018), p. 20180083, https://doi.org/10.1098/rsta.2018.0083. [Cited on pages 96 and 92.]

[294] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, *Beyond Inferring Class Representatives: User-Level Privacy Leakage From Federated Learning*, arXiv:1812.00535 [cs], (2018), https://arxiv.org/abs/1812.00535. [Cited on pages 88, 89, 90, 91, 95, 97, 157, 84, 85, 86, 87, 93, and 149.]

[295] M. Werlberger, M. Unger, T. Pock, and H. Bischof, *Efficient Minimization of the Non-local Potts Model*, in Scale Space and Variational Methods in Computer Vision, Lecture Notes in Computer Science, Springer Berlin Heidelberg, May 2011, pp. 314–325, https://doi.org/10.1007/978-3-642-24785-9_27. [Cited on pages 7 and 6.]

[296] J. M. Winn, A. Criminisi, and T. P. Minka, *Object categorization by learned universal visual dictionary*, in Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, vol. 2, Oct. 2005, pp. 1800–1807 Vol. 2, https://doi.org/10.1109/ICCV.2005.171. [Cited on pages 18 and 17.]

[297] S. Wiseman and A. M. Rush, *Sequence-to-Sequence Learning as Beam-Search Optimization*, in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, Nov. 2016, Association for Computational Linguistics, pp. 1296–1306. [Cited on pages 58 and 53.]

[298] C. F. J. Wu, *On the Convergence Properties of the EM Algorithm*, The Annals of Statistics, 11 (1983), pp. 95–103, https://doi.org/10.2307/2240463. [Cited on pages 24 and 22.]

[299] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, *Is Feature Selection Secure against Training Data Poisoning?*, in International Conference on Machine Learning, June 2015, pp. 1689–1698. [Cited on pages 113 and 108.]

[300] L. Xiao, F. Heide, M. O'Toole, A. Kolb, M. B. Hullin, K. Kutulakos, and W. Heidrich, *Defocus deblurring and superresolution for time-of-flight depth cameras*, in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2015, pp. 2376–2384, https://doi.org/10.1109/CVPR.2015.7298851. [Cited on pages 48, 44, and 45.]

[301] Q. Yang, Y. Liu, T. Chen, and Y. Tong, *Federated Machine Learning: Concept and Applications*, arXiv:1902.04885 [cs], (2019), https://arxiv.org/abs/1902.04885. [Cited on pages 86 and 82.]

[302] C. Zach, D. Gallup, J.-M. Frahm, and M. Niethammer, *Fast Global Labeling for Real-Time Stereo Using Multiple Plane Sweeps*, in Vision, Modeling, and Visualization, Amsterdam, The Netherlands, Aug. 2008, IOS Press, pp. pp. 243–252. [Cited on pages 7 and 6.]

[303] C. Zach, C. Häne, and M. Pollefeys, *What is optimized in tight convex relaxations for multi-label problems?*, in 2012 IEEE Conference on Computer Vision and Pattern Recognition, June 2012, pp. 1664–1671, https://doi.org/10.1109/CVPR.2012.6247860. [Cited on pages 7 and 6.]

[304] C. Zach, T. Pock, and H. Bischof, *A Duality Based Approach for Realtime TV-L1 Optical Flow*, in Pattern Recognition, Springer, Berlin, Heidelberg, Sept. 2007, pp. 214–223, https://doi.org/10.1007/978-3-540-74936-3_22. [Cited on pages 7 and 6.]

[305] W. Zhan, J. Li, Y. Hu, and M. Tomizuka, *Safe and feasible motion generation for autonomous driving via constrained policy net*, in IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, Oct. 2017, pp. 4588–4593, https://doi.org/10.1109/IECON.2017.8216790. [Cited on pages 56 and 51.]

[306] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, *Understanding deep learning requires rethinking generalization*, arXiv:1611.03530 [cs], (2016), https://arxiv.org/abs/1611.03530. [Cited on pages 115 and 111.]

[307] H. Zhang, Y. Avrithis, T. Furon, and L. Amsaleg, *Smooth Adversarial Examples*, arXiv:1903.11862 [cs], (2019), https://arxiv.org/abs/1903.11862. [Cited on pages 126 and 121.]

[308] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, *Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising*, IEEE Transactions on Image Processing, 26 (2017), pp. 3142–3155, https://doi.org/10.1109/TIP.2017.2662206. [Cited on pages 56, 74, 51, and 69.]

[309] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song, *The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks*, arXiv:1911.07135 [cs, stat], (2019), https://arxiv.org/abs/1911.07135. [Cited on pages 88 and 84.]

[310] Y. Zhang, K. Sohn, R. Villegas, G. Pan, and H. Lee, *Improving object detection with deep convolutional networks via Bayesian optimization and structured prediction*, in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, June 2015, IEEE, pp. 249–258, https://doi.org/10.1109/CVPR.2015.7298621. [Cited on pages 76 and 71.]

[311] B. Zhao, K. R. Mopuri, and H. Bilen, *iDLG: Improved Deep Leakage from Gradients*, arXiv:2001.02610 [cs, stat], (2020), https://arxiv.org/abs/2001.02610. [Cited on pages 88, 89, 90, 95, 84, 85, 86, and 91.]

[312] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr, *Conditional Random Fields as Recurrent Neural Networks*, 2015 IEEE International Conference on Computer Vision (ICCV), (2015), pp. 1529–1537, https://doi.org/10.1109/ICCV.2015.179, https://arxiv.org/abs/1502.03240. [Cited on pages 58, 76, 53, and 71.]

[313] C. Zhu, W. R. Huang, A. Shafahi, H. Li, G. Taylor, C. Studer, and T. Goldstein, *Transferable Clean-Label Poisoning Attacks on Deep Neural Nets*, arXiv:1905.05897 [cs, stat], (2019), https://arxiv.org/abs/1905.05897. [Cited on pages 114, 116, 119, 122, 110, 112, 115, and 118.]

[314] L. Zhu, Z. Liu, and S. Han, *Deep Leakage from Gradients*, in Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 14774–14784. [Cited on pages 87, 88, 90, 91, 94, 95, 97, 99, 157, 83, 84, 86, 93, and 149.]

Index

List of Figures

List of Tables