Multi-Layer Conditional Random Fields

---

for Revealing Unobserved Entities

DISSERTATION
zur Erlangung des Grades eines Doktors
der Ingenieurwissenschaften

vorgelegt von
M.Sc. Sergey G. Kosov

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Siegen 2018

Gutachter der Dissertation

Prof. Dr. Marcin Grzegorzek
Siegen Universität Siegen

Prof. Dr. Klaus-Dieter Kuhnert
Siegen Universität Siegen

Tag der mündlichen Prüfung

19.07.2018

Acknowledgements

First of all, I would like to thank *Prof. Dr. Marcin Grzegorzek* for supervising my dissertation and giving me the opportunity to work in his group. The fruitful atmosphere there supported and motivated me during the preparation of this thesis. In particular, I would like to thank *Prof. Dr. Marcin Grzegorzek* for giving me a helping hand in critical moments and thus saving my soul from becoming a hardware engineer. Secondly, I want to thank *Prof. Dr. Roman Obermaisser* and *Prof. Dr. Markus Lohrey* who agreed to review this thesis.

Moreover, I want to thank all former and current members of our group: *Kimiaki Shirahama*, *Chen Li*, *Hassan Khan* and all others. They created such a friendly and nice atmosphere that working with them was really a pleasure. Each of them contributed in his way to the success of this thesis. In particular, I want to thank *Kimiaki Shirahama* and *Chen Li*, for our fruitful cooperation on conditional random fields. In this context, I also want to explicitly thank *Hassan Khan* for doing a work that is often underestimated. Also I would like to thank my colleagues of the firm Reality 7 *Max Konrad* and *Daniel Rembold* for proofreading parts of my thesis and supporting me with the system administration.

Furthermore, I want to thank my friends *Sergey Pushkarev*, *Eugenij Krjukov*, and *Juri Hanimann* as well as *Ulan Degenbaev* and *Pavel Emiljanenko* for their long time friendship. They accompanied me through most parts of my life and always believed in me.

Finally, I want to thank my mother *Irina* for giving me always the support I need and my girlfriend *Alexandra* for her love and for showing me that there is more in life than science.

Siegen, August 10, 2018 Sergey Kosov

Abstract

Understanding the role of each pixel in the image – the so-called semantic image segmentation – is one of the central problems in computer vision and pattern recognition. Allowing a mathematical sound integration of different image labeling concepts into a single framework, conditional random fields belong to the best performing and best understood techniques for solving this task. They belong to the class of undirected graphical models, where the scene is represented by a graph whose nodes are the random variables involved in the classification process and whose edges model dependencies between the random variables corresponding to the nodes. However, they are often considered as a statistical model of context, which has a smoothing effect on the classification results. In this thesis I show that the conditional random fields technique is a much more powerful tool for semantic image segmentation by making two important scientific contributions, described in Chapters 2 and 3.

The first part of this thesis is dedicated to construction of conditional random fields methods (Chapter 2). I first discuss some classical probabilistic models, used for initializing the graph nodes and edges, and then propose new more accurate and efficient models, which are based on classical ones. Thereby, I demonstrate that this toolkit allows for incredible flexibility in modeling the graph structure and thus binding various kinds of observations together. Here I also investigate the influence of different data-features, extracted from the observations on he entire labeling process. Finally, I construct a *local-global* classification engine – conditional random field, incorporating not only classical *local* nodes, but also additional *global* nodes, which correspond to the global features that describe the whole image *in toto*. Extensive qualitative and quantitative benchmarks for eight different node models and five edge models show the accuracy and the efficiency of the proposed implementations. At the current *status quo* this provides the most precise random fields approaches in the literature and allows me to make the second scientific contribution.

The second part of this thesis extends the previous scientific contributions to a novel *Multi-Layer-CRF* framework (Chapter 3) that allows for the integration of sophisticated occlusion potentials into the model and enables the automatic inference of the layer decomposition. I use a special message-passing algorithm to perform maximum a posterior inference on mixed graphs and demonstrate the ability to infer the correct labels of occluded regions in both the aerial near-vertical dataset and urban street-view dataset. A major innovation of the proposed framework is that the 3D structure of the scene is considered in the classification process. This is necessary to be able to deal with occlusions in a systematic way. In order to do so, multi-layer conditional random fields are built that use multiple nodes for the class labels at a certain position in object space, namely one corresponding to the base layer of the

scene (containing background objects that do not occlude other objects but may be occluded) and others, corresponding to the occlusion layers (containing objects that may occlude other objects). Quality and efficiency benchmarks show the success of this layered framework: the accuracy of classification on occluded areas becomes considerably higher in comparison to the classical random fields techniques.

Zusammenfassung

Die Rolle jedes einzelnen Pixels im Bild zu verstehen – die so genannte semantische Bildsegmentierung – ist eines der zentralen Probleme der Computer-Vision und Mustererkennung. Conditional Random Fields gehören zu den leistungsstärksten und am besten verstandenen Techniken zur Lösung dieser Aufgabe, da sie eine mathematische überzeugene Integration verschiedener Bildbeschriftungskonzepte in einem einzigen Framework ermöglichen. Sie gehören zur Klasse der ungerichteten Graphen, bei denen die Szene durch einen Graphen repräsentiert wird, dessen Knoten die Zufallsvariablen sind, die am Klassifizierungsprozess beteiligt sind, und deren Kanten die Abhängigkeiten zwischen den Zufallsvariablen, die den Knoten entsprechen, modellieren. Sie werden jedoch oft als statistisches Kontextmodell betrachtet, was auf die Klassifikationsergebnisse einen Glättungseffekt hat. In dieser Arbeit zeigen wir, dass die Technik der bedingten Zufallsfelder ein viel mächtigeres Werkzeug für die semantische Bildsegmentierung ist, indem wir zwei wichtige Beiträge leisten, die in den Kapiteln 2 und 3 beschrieben werden.

Der erste Teil dieser Arbeit widmet sich der Konstruktion von Methoden mit Conditional Random Fields (Kapitel 2). Wir besprechen zunächst einige klassische probabilistische Modelle, die für die Initialisierung der Graphenknoten und -kanten verwendet werden, und schlagen dann neue, genauere und effizientere Modelle vor, die auf klassischen Modellen basieren. Dabei zeigen wir, dass dieses Werkzeug eine unglaubliche Flexibilität bei der Modellierung der Graphenstruktur ermöglicht und somit verschiedene Arten von Beobachtungen miteinander verbindet. Hier untersuchen wir auch den Einfluss verschiedener Datenmerkmale, die aus den Beobachtungen extrahiert wurden, auf den gesamten Beschriftungsprozess. Schließlich konstruieren wir eine *lokal-globale* Klassifizierungsmethode, basieren auf Conditional Random Fields, das nicht nur klassische *lokale* Knoten enthält, sondern auch zusätzliche *globale* Knoten, die den globalen Merkmalen entsprechen, die das ganze Bild beschreiben *in toto*. Umfangreiche qualitative und quantitative Benchmarks für acht verschiedene Knotenmodelle und fünf Kantenmodelle zeigen die Genauigkeit und Effizienz der vorgeschlagenen Implementierungen. Beim aktuellen *Status quo* liefert uns dies die präzisesten Random Fields in der Literatur und erlaubt es uns, unseren zweiten Beitrag zu leisten.

Der zweite Teil dieser Arbeit erweitert unsere bisherigen Beiträge zu einem neuartigen *Multi-Layer-CRF* Framework (Kapitel 3), das die Integration anspruchsvoller Okklusionspotentiale in das Modell ermöglicht und die automatische Ableitung der Schichtzerlegung ermöglicht. Wir verwenden einen speziellen Message-Passing-Algorithmus, um eine Maximum a posteriori Inferenz auf gemischte Graphen durchzuführen und die Fähigkeit zu demonstrieren, die korrekten Bezeichnungen von verdeckten Regionen sowohl im Datensatz von Senkrechtauf-

nahmen als auch im urbanen Street-View-Datensatz herzuleiten. Eine wesentliche Neuerung des vorgeschlagenen Rahmens besteht darin, dass die 3D-Struktur der Szene bei der Klassifizierung berücksichtigt wird. Dies ist notwendig, um Okklusionen systematisch behandeln zu können. Dazu werden mehrschichtige bedingte Zufallsfelder aufgebaut, die mehrere Knoten für die Klassenbeschriftungen an einer bestimmten Position im Objektraum verwenden, nämlich einen, der der Basisebene der Szene entspricht (mit Hintergrundobjekten, die andere Objekte nicht verdecken, sondern verdecken können) und andere, die den Okklusionsebenen entsprechen (mit Objekten, die andere Objekte verdecken können). Qualitäts- und Effizienz-Benchmarks zeigen den Erfolg dieses mehrschichtigen Frameworks: Die Genauigkeit der Klassifikation auf den verdeckten Flächen wird im Vergleich zu den klassischen Random Fields deutlich erhöht.

To my second Grandmother

Contents

1 Introduction

"This work contains many things which are new and interesting. Unfortunately, everything that is new is not interesting, and everything which is interesting, is not new."

- Lev Davidovich Landau

1.1 Motivation

The two fundamental questions in Philosophy are *"what does reality consist of?"* and *"how does it originate?"*. Generally, all the philosophers in their attempts to answer these questions can be split in two primary categories, namely idealism, and materialism, which are both defined in contrast to each other. From the idealistic point of view the spirit, mind or the objects of mind (*ideas*) are primary, and matter is secondary. To materialists, it is the opposite so that matter is primary and the spirit, mind or ideas are secondary, which results in the product of matter acting upon matter. However, by itself materialism says nothing about how material substance should be characterized.

In this thesis I do not answer the fundamental questions in Philosophy, but address the problem of automatically revealing ideas, hidden behind the material substance. As we know, an idea may be reflected in a large variety of substances, which could be only perceived via definite *organa sensuum*. Thus, ideas appear to be hidden from us beyond the information

Fig. 1.1: Example of the classification problem. **Left:** A variety of non-material ideas; **Center:** Input data: the ideas in form of physical entities perceived via a digital camera sensor; **Right:** Wanted classification result: a mapping of the input image to categories. The color indicates the position of different categories in the image. How can we classify the input data in a fast and accurate manner?

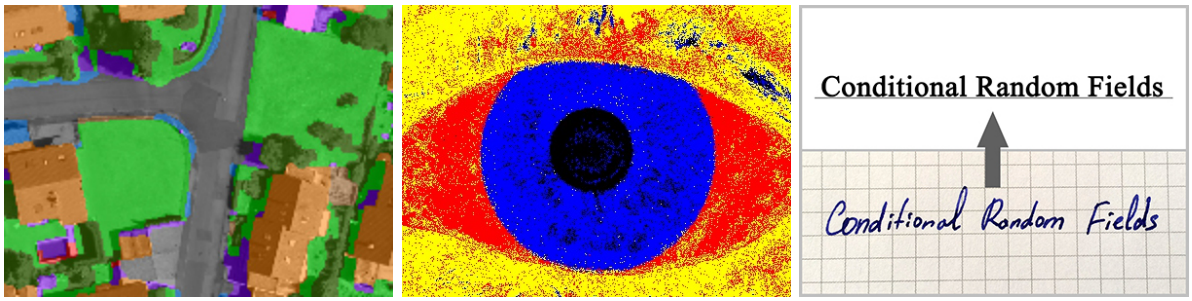


Fig. 1.2: Possible fields of application. **Left:** Classification of an optical satellite image; **Center:** Segmentation of a medical image of a human eye; **Right:** Handwritten text recognition. The color in first two images indicates different categories of interest.

provided by some "sensors". This information is limited, often incomplete and sometimes ambiguous. Only our experiences gained through growing up teach us to interpret, recognize and understand ideas.

In machine learning the problem of identifying to which category (idea) a new observation belongs is called *classification*. The main focus of this thesis is developing and studying a class of mathematical functions called *classifiers*, which are capable of mapping input data from *organa sensuum* to a category. Such a classifier can be taught with the help of vivid examples such as a training set of data containing observations (or instances) whose category membership is known. Figure 1.1 illustrates the described classification problem. In the visual data in a form of a digital image, we are interested in determining which objects were in front of the camera, where were they situated and how do they interact with each other?

Fields of Application. Although classification is only an ordered set of related categories used to group data according to its similarities, it proves to be useful for a variety of different tasks (see Figure 1.2). For example, in *remote sensing*, classification has been used for the detection of buildings in synthetic aperture radar (SAR) images, for the classification of optical satellite images, and for the reconstruction of a terrain surface from airborne laser scanner data [LOL09].

In pattern recognition, classification is used for *image labeling* [HZC04] and *object recognition* [QCD04]. Both are in great demand especially in medical image analysis for processing and classifying images, obtained by different types of image acquisition methods, like computer tomography (CT) and magnetic resonance imaging (MRI). In this example classification allows to distinguish diseases, detect and localize tumors [Bau+13]. In pharmacology and biology, classification is used for *drug discovery, development* [SHR10] and for *gene prediction* [DeC+07].

Due to industrialization we have a growing number of pollutants in recent decades, like

waste water entering the human environment. This increases the risks of serious diseases, such as cancer. Instead of using chemicals to eliminate such pollutants, a more harmless approach would be taking advantage of the natural consumption of *Environmental Microorganisms* (EM) [PGG15]. EMs are microscopic organisms living in natural and artificial environments (*e.g.*, forests and farmlands), which are useful for cleaning environments. For example, *Actinophrys* can digest the organic waste in sludge and increase the quality of fresh water and *Rotifera* can decompose rubbish in water and reduce the level of eutrophication. In order to achieve the environmental treatments, EM classification is necessary.

Additionally, classification has found applications in many other domains which deal with structured data. Especially in natural language processing, classification is currently a state of the art technique for many of its sub-tasks including basic *text segmentation* [TWH07], *part-of-speech tagging* [LMP01] and *shallow parsing* [SP03]. These techniques serve as an important step towards *human - machine interfaces*. This interaction with the environment goes far beyond a pure text and speech understanding. In combination with approaches from computer vision, motion patterns can be trained in such a way that the obtained algorithms even allow for the classification of human activities and gestures [KSG18a].

Furthermore, classification is also applied in telecommunications and networking for *intrusion detection* [GNK10] and *distributed sensor networks management* [ZAV07]. Classification allows not only for *recognition* of existing entities, but also the *prediction* of future events. A good example is the use in economics for *credit scoring* [BM12] and *stock–price prediction* [Che+09]. Classification may also be used for motion prediction, where it allows to track objects on their way through the scene, to keep them in focus and to follow them if desired. As a result computer vision allows us to detect and avoid obstacles. This makes it particularly useful for tasks where vehicles must be guided safely through an unknown environment. In this context one can either think of a robot navigation where one is interested in a fully autonomous behavior [DK02] or the design of drivers assistance systems where support is only required in certain situations [LMB02].

The direct relation to other machine learning problems and the previously discussed practical applications are only a few examples that demonstrate the usefulness of classification. This clearly shows why in the last few decades so much research has been carried out to develop accurate models and fast numerical schemes for its calculation.

1.2 Overview

The goal of the present work is to contribute to the field of classification. In order to do this I will start the overview with a formal definition of classification as the association of an

observation $y \in \mathbb{Y}$ with a *category* $l \in \mathbb{L}$, where $\mathbb{Y}$ is a set of all observations, which may be infinite and $\mathbb{L}$ is a pre-defined finite set of all the categories of interest. The categories are often called *ideas*, *classes* or *labels*, and thus classification is also known as *labeling* in the literature. The function $\mathcal{F} : \mathbb{Y} \to \mathbb{L}$, which maps a space of observations into a space of class labels is called *classifier*.

Figure 1.3 illustrates the underlying classification concept. An idea finds itself in a "one-to-many" relationship to physical objects, *i.e.* one idea may potentially reference several objects, whereas each object may be referenced by only one idea. In the real world, almost each physical object is unique and its observation also depends on the observer. Thus, the task of classification is to learn how ideas are related to observations and finding similarities in the observed data to answer "what is the underlying idea l behind this observation y?"

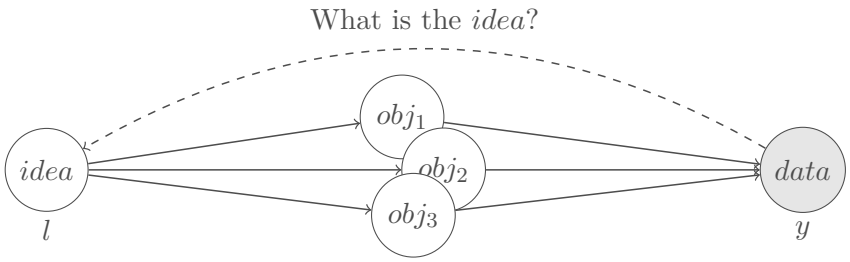


Fig. 1.3: A schematic representation of the classification concept: one idea may (category) cause numerous physical objects: obj_1, obj_2, obj_3, ... The task of classification is to reveal the idea, hidden under these objects, having only observed data about them: $l = \mathcal{F}(y)$.

As an example of classification let us consider a patient diagnostication: each inspected patient $y \in \mathbb{Y} = \{Ivanov, Petrov, \ldots\}$ falls under one of the categories $\mathbb{L} = \{healthy, infected, ill, convalescent\}$. In order to assign a patient to one of the categories, doctors should consider symptoms, take and inspect analysis. Mathematically saying, doctors extract those *features*, which they believe are most descriptive and useful for giving the right diagnosis. So a patient y is described by a set of features such as $f_{leucocytes}(y)$, which returns an integer number of white blood cells from a blood test, or a floating point $f_{temperature}(y)$ which returns the body temperature, or even a binary $f_{headache}(y)$, which is equal to one when the patient has a head ache or zero if not. To this end, any patient y could be substituted by a corresponding *feature vector* $\mathbf{f}(y)$ thus $y \equiv \mathbf{f}(y)$. Analyzing all features $\mathbf{f}(y)$, doctors, based on their knowledge and experience, diagnose the patient and assign a course of medical treatment if needed.

Usually, doctors are taught in universities and during practice. If patient diagnostication will be committed by a machine, it must be taught as well. This teaching process is a machine learning concept called *training*. We distinguish between *supervised* (learning with a teacher) and *unsupervised* training. When learning with a teacher, the fully labeled training dataset is provided, as if a teacher points out the correct category for a concrete data sample; whereas

the problem of unsupervised learning is that of trying to find a hidden structure in unlabeled data, so there is no error or reward signal to evaluate a potential solution. Henceforth this thesis will refer to the term "training" as supervised training, if not stated otherwise.

A special thing to note is that in the example of patient diagnostication, all the patients are considered separately, *i.e.* the diagnosis of one patient is completely independent from the diagnosis of another patient. Let us consider another example from natural language processing called *text segmentation*, which means the classification of words y of a textual sequence $\mathbb{Y}$. Here, words are not an arbitrary accumulation, but interdependent entities. Especially the order is important and grammatical constraints hold. The linguistically motivated categories here include part-of-speech tags and proper names: $\mathbb{L} = \{verb, noun, name, 0\}$, where 0 represents any other word in which we are not interested. Let us consider the text snippet:

```
... Mister Vladimir Vladimirovich Putin visited Hanover together with ...
```

Here, the task is to assign a fitting label sequence such as:

$$\ldots \; noun \; name \; name \; name \; verb \; name \; 0 \; 0 \; \ldots$$

where each label represents an entity class. In this example we deal with a linear sequence structure and multiple interdependent observations are considered. Now, classification can be approached with probability theory by specifying a probability distribution to select the most likely class l for a given observation y. One approach for modeling linear sequence structures, as can be found in natural language text, are *hidden Markov models* (HMM, [Rab89]). For the sake of simplicity, we assume strong independence between the observation, what impairs the accuracy of the model.

This thesis addresses the most difficult class of classification problems - *image* classification (visual data categorization). In this case, the data $\mathbb{Y}$ is given in a form of a digital image, and the complexity of the probability distribution becomes even higher. To this end, the current work is focused on the class of statistical classification approaches that currently gives the best results in the literature – so-called *conditional random fields* (CRF, [LMP01]). Conditional random fields are developed exactly to fill the gap, that HMM has: While CRFs make similar assumptions on the dependencies among the class variables, no assumptions on the dependencies among observations need to be made (Section 1.2.1).

Within this thesis, I am interested in improving the qualitative performance of current classification techniques. By discussing existing and novel concepts for designing such methods, a *systematic toolkit* for the construction of novel highly accurate CRF techniques is provided. Additionally, I am also interested in studying the possibility of the CRFs application to the problem of implicit data classification, where a solid probabilistic framework to handle such

kind of problems is presented. They arise when some part of data to be classified, due to some reasons, is not observed directly, but still might be "guessed" by observed neighboring data. This allows for making conditional random fields more robust for real-world problems where one faces incomplete, occluded or distorted input data. Furthermore, I am also interested in a fast computation of the results. Thus, a general numerical framework for CRF modeling that is based on efficient computational schemes is also presented. By addressing these aspects of the *modeling* and the *numerics* of conditional random fields, this thesis provides a common basis for the design of fast and accurate classification techniques.

To specify my scientific contributions in detail, let me give a short introduction to conditional random fields and to the use of multiple layers in the context of CRF modeling. In addition, the relevant work that is related to both fields of research will be discussed.

1.2.1 Classification with Conditional Random Fields

Since the prototypical approach of Lafferty *et al.* [LMP01] in 2001, conditional random fields are among the best performing and understood probabilistic techniques for classification. All the underlying probabilistic models in CRF technique could be solved purely by algebraic manipulation. However, it makes sense to use the advantage of augmenting the analysis using diagrammatic representations of probability distributions, called *probabilistic graphical models*. These give two major benefits:

1. *Representation.* Probabilistic graphical models provide a simple way to represent the structure of a probabilistic models and can be used to design and motivate new models.

2. *Simplification.* Complex computations, required to be performed in sophisticated models, can be expressed in terms of graphical manipulations, in which underlying mathematical expressions are carried along implicitly.

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ comprises *nodes* $\mathcal{V}$ (also called *vertexes*) connected by *links* $\mathcal{E}$ (also known as *edges*). In a probabilistic graphical model, each node v_i, $i \in \mathcal{V}$ represents a *random variable* (or group of random variables) and the links express probabilistic relationships between these variables. A *complete graph*, which has every pair of distinct nodes connected by a unique link, can represent any probability distribution. Such a construction reflects the fact, that all ideas find themselves in equilibrium and referenced physical objects are more or less related.

Despite universality of complete graphs, they, being computationally intractable, are hardly applicable to real-world problems. The major simplification to be made here is to neglect weak relations and corresponding graph edges. Thus, an absence of a link between

two nodes, will mean that the corresponding random variables, represented by these nodes, are *conditionally independent* (Appendix B). Conditional independence properties play an important role in using probabilistic models for pattern recognition by simplifying both the structure of a model and the computations needed to perform inference and learning under that model.

By introducing the conditional independence to the probabilistic model, the corresponding joint probability can be decomposed (factorized) into a number of less complex factors. The graph then captures the way in which the joint distribution over all of the random variables can be decomposed into a product of factors, each depending only on a subset of the variables. This leads to a graphical concept called *clique*, which is as a subset of the nodes in a graph, such that there exists a link between all pairs of nodes in this subset. In other words, the set of nodes in a clique is a complete sub-graph. Here, probabilistic graphical models give an insight into the properties of the model, including conditional independence properties, which can be obtained by inspection of the graph.

According to the CRF technique, the desired class map could be determined as the maximizer of a suitable probability distribution, where deviations from model assumptions are penalized. In general, this probability distribution consists of two terms: a *data term* that estimates probabilities of being distinct categories, for each observation, independently on all other observations; and a *smoothness term* that regularizes the often non-unique solution of the data term by an additional smoothness constraint by considering probabilistic relations between observations. Thus, the smoothness term stands for the assumption that neighboring observations probably belong to the same category and thus undergo same classes.

How Do Conditional Random Fields Work? Let me demonstrate this rather general strategy by a concrete example: the classical conditional random fields of Lafferty *et al.* [LMP01]. This method is based on the relaxation of two most frequently used assumptions in the classification literature:

- assumption of *conditional independence* of the observed data given the labels, commonly used in the *Markov random field* (MRF, [Li09]) framework and;
- *homogeneous regularization* which assumes that the resulting categories map is smooth everywhere, except the classifying objects' borders.

In order to formulate these two assumptions mathematically, let us consider a scalar-valued image $\mathbf{y} \in \mathbb{Y}$, that consists of n image sites (usually pixels or small segments). Each distinct image site $y_i \in \mathbf{y}$, $i \in \mathcal{V} = \{1, 2, \ldots, n\}$ will be treated as a single observation, where $\mathcal{V}$ corresponds to the array of all sites. With each site i one graph node v_i is associated; it represents

a random variable, which takes value l from a given set of classes $\mathbb{L} = \{l_1, l_2, \ldots, l_k\}$.

A probability distribution over variables v can be represented by an *undirected graphical model* using a product of non-negative functions of the maximal cliques of graph $\mathcal{G}$. The factorization is performed in a way that conditionally independent nodes do not appear within the same factor, that means that they belong to different cliques $\mathcal{C} \subset \mathcal{G}$. Collecting the random variables v_i in a vector $\mathbf{v} = (v_1, v_2, \ldots, v_n)^\top$, one can write the joint probability formula over these variables:

$$p(\mathbf{v}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \varphi_c(\mathbf{v}_c) \tag{1.1}$$

Here the factors $\varphi_c \geq 0$ are called *potential functions* of the random variables $\mathbf{v}_c$ within a clique $c \in \mathcal{C}$ and Z is a normalization constant.

The potential functions may be any arbitrary functions. Due to this generality the potential functions do not necessarily have to be probability functions. Thus, normalization of the product of potential functions is necessary to achieve a proper probability measure. This is yielded by a normalization constant Z, sometimes called the *partition function* and given by:

$$Z = \sum_{\mathbf{v}} \prod_{c \in \mathcal{C}} \varphi_c(\mathbf{v}_c) \tag{1.2}$$

which ensures that the distribution $p(\mathbf{v})$ given by Equation 1.1 is correctly normalized. Calculating Z is one of the main challenges during parameter learning as summing over all possible variables is necessary. By considering only potential functions which satisfy $\varphi_c(\mathbf{v}_c) \geq 0$ it is ensured that $p(\mathbf{v}) \geq 0$. Thus, CRF are discriminative models that directly model the probability $p(\mathbf{v})$.

To simplify the model in 1.1, we can restrict ourselves to *pairwise graphs* – graphs, where the maximal clique is consisted of maximal two nodes [KH06]:

$$p(\mathbf{v}) = \frac{1}{Z} \prod_{i \in \mathcal{V}} \varphi_i(v_i) \prod_{(i,j) \in \mathcal{E}} \psi_{ij}(v_i, v_j). \tag{1.3}$$

Here $\varphi_i(v_i)$ are the *unary potential functions*, which return a one-dimensional vector of potentials for the node v_i of being each of the classes $l \in \mathbb{L}$. Thus the length of the returning vector is equal to $|\mathbb{L}|$. Terms $\psi_{ij}(v_i, v_j)$ are the *pairwise potential functions*, returning a square matrix $|\mathbb{L}| \times |\mathbb{L}|$ of potentials for the nodes v_i and v_j to be at the same time classes l_1 and l_2. Thus, the pairwise potential functions model the dependencies between those graph nodes, which are connected with an edge. Thus, the product of unary potential functions substitutes the data term of the equation, and the product of the pairwise potentials – the smoothness term.

The definition of the unary and pairwise potential functions depends on the underlying probabilistic models, on the graph structure, and how their parameters could be estimated during the training phase. All these nuances are described more deeply in Sections 2.3 and 2.4 for unary and pairwise potential functions respectively.

> The ***first main goal*** in this thesis is to provide a general toolkit for the systematic construction of conditional random fields methods based on the general probability distribution in Equation 1.3. In order to achieve it, various established concepts for the design of both the data and the smoothness term will be discussed. Apart from investigating existing strategies, several novel ideas that address typical problems in classification such as noise, over-smoothing and outliers will be also presented.

Inference and Decoding. Finally, determination of the probability distribution $p(\mathbf{v})$ from Equation 1.3 with help of a set of training data is an *inference* problem. Its solution provides $|\mathbb{L}|$ probabilities of being each of the classes $l \in \mathbb{L}$ for each variable v_i, $i \in \mathcal{V}$. Thus, the joint distribution $p(\mathbf{v})$ gives the most complete probabilistic description of the model. Inference typically a very difficult problem, whose solution forms the subject of much of the Chapter 2 of this thesis.

Although $p(\mathbf{v})$ can be a very useful and informative quantity, in the end we are interested in a categories map $\tilde{\mathbf{v}}$ - a sequence of concrete class values for each node. This is the subject of the *decision* problem to answer the question how to make an optimal choice of a class l, for variable v_i, given the appropriate probability vector $p(v_i)$.

Both inference and decision problems form the *decoding* problem – finding the most probable configuration of labels $\tilde{\mathbf{v}}$ over the underlying graph. Thus, the decoding problem could be split into two separate stages: the *inference stage*, in which the training data is used to learn a model for $p(\mathbf{v})$, and the subsequent *decision* stage in which these posterior probabilities are used to make optimal class assignment. Once the inference problem is solved, the decision problem may be very simple, even, trivial if to choose $\tilde{\mathbf{v}}$ to be the classes with the highest probabilities, *i.e.*:

$$\tilde{\mathbf{v}} = \underset{\mathbf{v}}{\operatorname{argmax}}\, p(\mathbf{v}). \tag{1.4}$$

This an intuitively correct decision approach in case, when the aim is to minimize the chance of assigning v_i to a wrong class.

From the other hand one's objective may be more complex than simply minimizing the number of misclassifications. Mistakes of classifying different classes may cause different consequences, thus one might be interested of making fewer mistakes in the classification of one class in cost of more mistakes of other classes. This issue may be formalized through

the introduction of a *loss function* (also called a *cost function*), which is a single, overall measure of loss incurred in taking any of the available decisions or actions. The goal then is to minimize the total loss incurred. Suppose that, the true class for a variable v_i is l_t and that it is labeled as a class l_q (where q may or may not be equal to t). In so doing, some level of loss denoted by $L_{q,t}$ is incurred, which can be viewed as the q,t element of a *loss matrix*. At Figure 1.4 a loss matrix is depicted. It is a zero-diagonal quadratic $|\mathbb{L}| \times |\mathbb{L}|$ matrix, where each non-diagonal value $L_{q,t} > 0$.[1] Such matrix says that here is no loss incurred if the correct decision is made, and there is a loss of $L_{q,t}$ if a class l_t was classified as class l_q.

$$L = \begin{array}{c} \\ \text{assigned class} \end{array} \begin{array}{c} \text{true class} \\ \begin{pmatrix} 0 & L_{1,2} & \cdots & L_{1,k} \\ L_{2,1} & 0 & \cdots & L_{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ L_{k,1} & L_{k,2} & \cdots & 0 \end{pmatrix} \end{array}$$

Fig. 1.4: A loss matrix with elements $L_{q,t}$. The rows correspond to the true class, whereas the columns correspond to the assignment of class made by a decision criterion.

Thus, the optimal solution to the decision problem is the one which minimizes the loss function. However, the loss function depends on the true class, which is unknown. Nevertheless, the expectation of the loss (its average) can be still estimated. Thus the decision rule that minimizes the expected loss is the one that assigns class l to each new v_i for which the multiplication is a minimum:

$$\tilde{\mathbf{v}} = \arg\min_{\mathbf{v}} (L \times p(\mathbf{v})) \tag{1.5}$$

It is possible to show that for the case zero-one loss matrix, (*i.e.* when all the non-diagonal values $L_{q,t} = 1$, $\forall q \neq t$), the problem 1.5 is equivalent to the problem 1.4.

Why Conditional Random Fields? Although the basic idea of conditional random fields has been already explained, it was only briefly motivated so far why exactly this class of classification techniques has been chosen. This is in particular important with respect to the fact that there are also numerous other strategies in the literature to solve the underlying classification problem:

- such as *linear models* that seek the corresponding class based on the value of a linear combination of features, which represent an observation [MN89; MP00; CG01; LD05];

[1] Cases where $L_{q,t} = 0$ for $q \neq t$ can be dealt with, but lead to additional technical overhead, which is chosen to be avoided for the sake of clarity.

- also *non-probabilistic approaches*, like *decision trees* [FM99], *support vector machines* (SVM, [CS99]) or *neural networks* [Bis95], that try to assign an observation to a class without estimating the probabilities of being this class (further within this work it will be shown, how these models and approaches could be utilized in the CRF framework);
- and the most related techniques, which are also based on graphical models: *Bayesian networks* and *Markov random fields* that use very strong independence assumptions on conditioning the random variables upon observations [Jen96; Cow+07; Li09].

All of the mentioned methods, just like conditional random fields, may be employed for a raster-based classification in many labeling tasks in computer vision and pattern recognition. Let us now discuss the four main advantages that conditional random fields offer when compared to the aforementioned classes of classification techniques:

\+ *Transparent Modeling.* CRFs allow for a transparent modeling by construction: All assumptions on the observations and corresponding random variables are explicitly formulated in the underlying probability distribution. The use of a joint probabilistic framework allows thereby a mathematical sound integration of all desired assumptions. Moreover, modeling with help of a graph, gives insights into the fundamental properties of the model, including conditional independence properties.

\+ *Processing of Sequential Data.* CRFs allow to consider and model dependencies between different observations and corresponding random variables within a solid probabilistic model. Graphs, in its turn, provide a simple way to visualize the structure of these inter-dependencies and can be used to design and motivate new models. This ability makes CRF the most preferable tool for classifying sequential and structured sets of inter-dependent observations, like digital images.

\+ *Discriminative Classifier.* In contrast to MRF, in CRF each random variable v_i is conditioned not only upon a single corresponding observation y_i, but may also be conditioned upon a set of observations $\mathbf{y}$, *i.e.* each clique potential function $\varphi_c(v_c)$ is a mapping from all assignments to both the clique c and the observations $\mathbf{y}$ to the non-negative real numbers. Thus CRF belongs to the class of discriminative classifier, but capable of integrating strong points of both generative and discriminative classification approaches.

\+ *Better Qualitative Performance.* As shown in different performance evaluations [RFF06; Sch12] and in the recent literature on CRFs [LMP01; KH03; SP03; HZC04], CRFs are those techniques that currently offer the highest precision in terms of error measures [Faw06].

These advantages make the superiority of conditional random fields explicit. However, all known classification techniques by construction are extremely dependent on the observation data. In those cases when a reliable observation is not available or some ambiguities in its interpretation exist, most classification algorithms fail to deliver the proper label for this observation. Such a situation is common *e.g.* when the classifying object is partially occluded by another object. Model-based techniques have tried to overcome this problem by treating such objects as context objects in an ad-hoc manner [HB03], but a systematic statistical model for dealing with occlusions is still missing.

The same problem arises if the object of interest (*e.g.* EM) is semi-transparent. Nevertheless a good feature makes it easier to assign the appropriate label to the pixel. With respect to this, features extracted with a *Deep Convolutional Neural Network* (DCNN) were chosen to be used in this work because of DCNN's impressive performance in many computer vision problems, including classification, segmentation and captioning of images/videos, object detection and action recognition [Gu+17].

Another problem which is common to probabilistic models of context, used to model dependencies between random variables at neighboring image sites, is the *over-smoothing* in the objects' boundaries [Sch12]. Although smooth boundaries improve results in most cases they also introduce an undesirable side effect – smoothed label images compared to local classifiers. This creates a problem of extracting the fine contours of certain object classes such as trees and bushes. However CRFs have been shown to produce reliable results, they often tend to the over-smoothing in areas where the image content changes abruptly and quite often do not match the actual object contour.

> The ***second main goal*** in this thesis is to develop a probabilistic framework for the efficient classification of the occluded image regions with objects' boundaries preservation. In order to achieve it my framework will be based on the conditional random fields technique, since it provides the flexibility of modeling the graph in such way, that the information from neighboring non-occluded objects will be shared in order to support the decision on the occluded objects. In particular, the novel concepts for CRF-based classification as *global nodes*, *multiple layers* and *mixed graphs* will be presented. This development will be supported with an investigation on better models for unary and pairwise potentials and features.

Small Dataset Problem. In respect to the task of EM classification, environmental investigations are always operated in outdoor environments, where conditions like temperature and salinity are changing continuously. Because EMs are very sensitive to these conditions, their

quantity is easily influenced. So, one faces the *small training dataset* problem [SGU14], where it is difficult to collect sufficient EM images for training a DCNN with numerous parameters to be optimized.

In this thesis the proposed solution for this is inspired by the "pre-training and fine-tuning" approach that pre-trains a DCNN on a large auxiliary dataset, followed by domain-specific fine-tuning on a small dataset [Gir+14; Cha+14]. However, DCNNs usually used in this approach target at extracting image-level or region-level features, and cannot be used for the pixel-level feature extraction. Hence, a DCNN pre-trained on a large image dataset is re-purposed by replacing fully connected layers with convolutional layers with upsampled (dilated) filters [Che+16]. Then, EM images are used to fine-tune this DCNN to produce dense pixel-level feature maps for an image. Here, each pixel is represented as a feature vector consisting of values in these maps. It should be noted that the feature implicitly includes spatial relations between the pixel and surrounding ones, because the field of view of units in the DCNN are gradually enlarged by passing layers. In order to explicitly handle spatial characteristics of EMs, on top of such pixel-level features, the two extensions described below are implemented [Kos+18].

In addition, global features are useful in applications where a rough segmentation of the object of interest is available. Whereas pixel-level features are extracted in a 'bottom-up' fashion that only exploits physical pixel values in an image, the global features are utilized as a 'top-down' prior knowledge about contours, shapes and textures of EMs. Global features provide such different kinds of information and it is expected that classifiers that use both of them will outperform classifiers based only on pixel-level features. CRFs offer great flexibility in modeling dependencies between random variables, providing a principled way to bind random variables not only for handling spatial relations among pixel-level features, but also for integrating them with global features [Kos+18].

1.2.2 Multi-Layered Conditional Random Fields

After the main goals of this thesis were clarified, let me indicate how they will be reached. First, let us concentrate on two main problems of the image classification with conditional random fields: occlusions and over-smoothing.

Occlusion Problem. Usually classification techniques determine a class label for each pixel of an image. The problem arises when an object to be classified is partially occluded, *i.e.* the observation, corresponding to the occluded object describes an occluding one. For instance, the appearance of streets, sidewalks and buildings may not be clear for a computer if they are largely occluded by objects such as cars or trees. In remote sensing images, characterized by

near-vertical views, this has been known to be a problem for a long time, in particular in the context of automated road extraction [May+06]. I propose to associate with each observation several random variables, corresponding to the occluded and occluding objects. Since there is no prior knowledge weather occlusion took place, the interaction between these variables will be driven by the ambiguity, arising when interpreting the observation.

In order to integrate this idea into a solid probabilistic framework I take the classical CRF framework as a base and develop upon it a generalization, which is called *multi-layered conditional random fields* (ML-CRF). The CRF technique is chosen to be the base technique because it allows for a flexible modeling of various inter-dependencies between random variables, and thus lets information to propagate from non-occluded regions to occluded. Classical CRFs are based on undirected graphical models and are useful for expressing soft constraints between random variables, whereas directed graphs are better suited for expressing casual relationships between random variables. When dealing with occlusions, it is important to model the graph in such a way, that the information propagates from the 'non-occluded' and 'occluding' nodes in the direction of the 'occluded' nodes and no ambiguity leaves them. Concerning inference, since at the end there remains a fixed joint probability distribution, the differences between directed and undirected graphical models are less important.

> With respect to the design of conditional random fields both *undirected* and *directed* graph links will be discussed. In the latter case a suitable probabilistic framework, based on mixed graphs will be also presented. This framework will allow to increase the classification ratio for images with vast occlusions.

A prior knowledge about the three-dimensional structure of the classifying scene may greatly simplify the application of the multi-layered CRFs. In the case, when multiple images of the scene are available, it is possible to apply an *optical flow technique* to reconstruct a *depth map* of the scene [Kos08]. In aerial imagery, in much the same way, multiple overlap images could be used to derive a *digital surface model* (DSM). Variational methods are among the best performing techniques for the dense optical flow estimation, and allow for a real-time performance [KTS09]. Together with an input image, a corresponding depth map (or a DSM) might be used for forming the feature vector that, in its turn, is used in the classification.

Over-Smoothing Problem. In order to handle this problem, a number of different strategies were proposed by me. First of all, to support the classes, which suffer from over-smoothing the most, additional *confidence features* were proposed by me [Kos+13b]. These features are supposed to have a high response on image segments, covered by target objects, and low response on the rest of image area. Such a response could be generated with help of

an object detector and a confidence feature may be reconstructed directly from its output. Within current work, for generating a confidence features, object detectors, which are based on rotation invariant features and support vector machines were used.

Another two strategies aim the improvement of the unary and pairwise potentials. For unary potentials, generative *Gaussian mixture model* (GMM, [Rey09]) and discriminative *K-nearest neighbors* (KNN, [Alt92]) are among the most accurate approaches [MB88; MP00]. However, the application of such approaches to the real-world tasks is problematic, due to high memory and processor time demands of training algorithms. In order to reduce the memory consumption and to speed up training, a new sequential learning scheme for the GMM is proposed by me [KRH13]. This scheme is considerably faster than widely used *expectation maximization* [DLR77; MK08]. For the KNN a new high-efficient neighbors search algorithm is proposed. Moreover, I propose to define pairwise potentials on image segments, generated using unsupervised segmentation algorithms. These potentials enforce label consistency in image regions and can be seen as a strict generalization of the commonly used for objects' border preservation pairwise contrast sensitive smoothness potentials.

The last strategy for increasing the classification rate for classes, which suffer from over-smoothing deals with the loss matrix. Till now many researchers have used only simplistic loss functions such as the Hamming loss, to enable efficient inference. The *higher order loss functions*, [PK12] can be incorporated in the learning process. Such loss functions ensure that the resulting label map does not deviate much from the ground truth in terms of certain higher order statistics.

1.3 Related Work

After the main goals of the thesis were clarified and the paths for reaching them were indicated, let us now discuss some relevant work that is related to the most important scientific contributions: the modeling of the local-global and multi-layer conditional random fields; the usage of improved techniques, as global, confidence and DCNN features, sequential GMM learning, efficient KNN classification and concatenated pairwise potentials for the ML-CRF efficient application.

1.3.1 Modeling of Conditional Random Fields

The modeling of conditional random fields for classification goes back to the prototypical approach of Lafferty *et al.* in 2001 [LMP01]. Since then, a lot of different CRF structures have been proposed by [SM04; FGM05; SGU15; SM07] to improve the performance of such techniques. Due to the vast amount of literature, concepts for the design of features, unary

and pairwise potentials are discussed separately. Moreover, only to the most relevant work is considered. More detailed references are given throughout the Chapter 2, where the design of conditional random fields is studied.

Features. According to the main scientific contributions to the feature extraction techniques in this thesis, the following fields of interest in the literature are considered: *confidence features* that use an additional object detector in order to improve the classification accuracy for particular classes [QCD04; Sch+09; Yan+10] and *DCNN features*, where deep learning builds a DCNN representing feature hierarchies with higher-level features formed by the composition of lower-level ones [NSJ15]. DCNN features are extremely helpful when the objects of interest are hard to describe with features, commonly used in classification problems: for example microscopic organisms, which are semi-transparent and are hardly distinguished even by humans.

In the task of EM detection and segmentation there are two basic categories of feature extraction techniques: *hand-crafting* and *feature learning* [BCV13]. Hand-crafted features are manually designed based on prior knowledge and investigation, including shape [NHB03; Li+13; Ver+15], texture [TIS06; KS09; PWS14], color [Kru+16; KS08; KS10], *etc.* However, hand-crafted features are insufficient for representing diverse appearances of EMs, because all those appearances cannot be assumed in advance. Compared to this, feature learning aims to extract useful features from a large amount of images. For example, Bag of Visual Words (BoVW) performs clustering of numerous local features (*e.g.*, SIFT) to find statistically characteristic ones called visual words [LSG15a]. A deep learning approach is adopted because of its superior expressive power over BoVW [BCV13; SG16].

- *Confidence Features.* The idea of utilizing an object detector to support multi-class classification is not new in the computer vision world: for example, in order to increase the classification accuracy of specific classes additional motorbike or pedestrian detectors were used [Sch+09; Yan+10]. However, these models approached object detection and classification as separate problems. Extracted features were not affected by the object detectors.

  In contrast to these works, in [Kos+13b] the output of a vehicle detector was used directly to generate the *car confidence feature*. In its turn, the car detection may also be integrated in to the classification framework as an intermediate stage, which is also based on further specific features, like rotational invariant *histogram of oriented gradients* (HOG), *local binary pattern* and Haar-like features [Gra+08]. Another interesting approach was shown in [KHD11], where new types of features for vehicle detection were

introduced: *color probability maps* and *pairs of pixel*. The latter were used to extract symmetric properties of image objects. An extensive overview of related work on vehicle detection can be found in [HBS06].

One original approach for object detection was described in [QCD04], where the objects were modeled as flexible constellations of parts, conditioned on local observations, found by an interest operator. Thus, the input image was split into smaller patches and for each class the probability of a given assignment of patches to local features was modeled. However, this approach implies that there is always one object of interest in the scene.

- *DCNN Features.* DCNN is a model that mimics the process of the visual cortex in a human brain. The effectiveness of features extracted from such a biologically inspired model has been validated in many works [ST10]. In [NSJ15], deep learning has been used for EM classification and segmentation tasks. First, a convolutional deep belief network and an SVM classifier is used to segment possible object regions, then a DCNN consisting of six layers is applied to predict the class of each possible region. In contrast, deep learning is adopted to jointly address EM classification and segmentation problems in a CRF framework. Because of this single integrated framework, my method is more copositive and effective. Furthermore, in [NSJ15] the authors train the DCNN for the EM classification task from scratch, so a data augmentation approach is applied to solve the small dataset problem. In contrast, an existing pre-trained DCNN is transferred to manage the small dataset problem.

  Recently researchers have proposed several DCNN-based image segmentation approaches that produce dense (high resolution) feature maps and can be extended to the pixel-level feature extraction [Che+16; BKC15; LSD15]. One approach for obtaining dense feature maps is to up sample feature maps using a deconvolution layer [LSD15] or using a decoder based on the down sampling record that represents value locations selected by a max-pooling layer [BKC15]. However, this requires to learn filter weights used in the deconvolution or decoder, and causes a significant increase of parameters. Hence, up-sampling feature maps is not suitable for EM classification involving the small dataset problem. Thus, another approach that upsamples 'filters' by inserting zeros (holes) between filter weights is adopted [Che+16]. By utilizing these upsampled filters with the stride of size 1, the resolution of feature maps can be efficiently maintained by suppressing the increase of parameters.

Unary Potentials. With respect to the design of unary potentials, there are three main fields of interest in the literature, namely: *Features* that were described above, *generative approaches*

that allow for an accurate mapping of observations to categories by reconstructing the unknown underlying probability distributions from training data [Fea04; LD05; MK08] and *discriminative approaches* that model posterior probabilities directly [FM99; Pla99; LLW07].

- *Generative Approaches.* Talking about the generative approaches in classification, it is impossible to avoid the naïve Bayes classifier [LD05]. Bayes model uses very strong independence assumptions and, therefore, it is among the most simple and understood methods, but from another hand it is very far from accurate approximation of probability distributions met in praxis. Nevertheless, it is surprisingly well applicable to a broad bunch of machine learning problems. It was shown that naïve Bayes inference is orders of magnitude faster than Bayesian network inference, when using belief propagation. This makes naïve Bayes models a very attractive approach for general probability estimation, particularly in large or real-time domains.

  In the case when a problem requires higher accuracy, one may consider the Gaussian mixture models – one of the most accurate methods among generative linear approaches. Training of GMM may be performed by maximizing the likelihood function with the help of some iterative numerical optimization techniques [Fle87; NW06] or by expectation maximization, which is also due to its iterative nature, relatively slow and requires all the training data to be held in the memory [MK08]. Another alternative method for estimation GMM parameters could be the sequential Monte Carlo method, also known as *particle filters* (PF, [Fea04]), which are usually used to estimate Bayesian models. Even though the PFs have sequential nature, they are still based on the simulation model and, therefore, are very memory demanding.

- *Discriminative Approaches.* Discriminative algorithms are compared in Table 1.1. Currently in the literature the most popular classifiers are: *Support Vector Machines* (SVM), *K-Nearest Neighbors* (KNN), *Artificial Neural Network* (ANN), *Random Forest* (RF) and *Convolutional Neural Network* (CNN). Besides these general classification methods, many algorithms are specially designed to solve some highly specialized classification problem. However, it is difficult to represent the spatial relations among local image regions (pixels) using the classifiers described above. In contrast, CRF is used to explicitly model such spatial relations as well as the relations between pixel-level and global features.

  One of the most popular among the discriminative approaches is the support vector machines. Standard SVM, being a non-probabilistic approach, does not provide calibrated posterior probabilities, needed to be utilized within the CRF framework. One method to achieve probabilities from SVM is to directly train a kernel classifier with a logic link

function and a regularized maximum likelihood score. However, training with a maximum likelihood score will produce non-sparse kernel machines. Instead, Platt proposed first to train an SVM, then train the parameters of an additional sigmoid function to map the SVM outputs into probabilities [Pla99], what gave a comparable quality to the regularized maximum likelihood kernel method, while retaining the sparseness of the SVM. More recently Lin *et al.* proposed an improved algorithm to the Platt's method, that converges in theory and avoids numerical difficulties [LLW07].

Another discriminative approach, that needs to be mentioned is the random forest classifier, which is based on boosting procedures to decision tree algorithms. It became extremely popular with a generalization of Freund and Llew [FM99], which describes new rules yielding a natural measure of classification confidence, that can be used to improve the accuracy at the cost of abstaining from predicting examples that are hard to classify. Being very accurate, this approach also belongs to the class of non-probabilistic classifiers and thus, cannot be used within a CRF framework directly. It is described in Section 2.3.2.3 how to overcome this problem.

Classifier	Related work	Classifier	Related work
KNN	[TIS06; Yu+11]	RF	[Kru+16]
ANN	[KS10; AE14]	CNN	[NSJ15]
SVM	[Fil+15; Ver+15]	Other methods	[Zou+16; Gut+14; PWS14]

Tab. 1.1: Overview of microorganism classification methods grouped by utilized classifiers. Nearest Neighbor (NN), K-nearest Neighbor (KNN), Artificial Neural Network (ANN), Support Vector Machine (SVM), Random Forest (RF), Convolutional Neural Network (CNN).

Pairwise Potentials. In the case of the pairwise potentials the literature mainly focuses on three aspects: The *data independent* potentials, that require no training procedure and observed data [Bla+04; Wer+11; CSK13], the *contrast sensitive* potentials, that still require no training, but takes in account the similarity between corresponding observations [SM00; BJ01; FH04; Pra+06; KLT08] and the *data dependent* potentials, that should be trained in a similar manner to the unary potentials [BSC08; GZT11; KRH12].

- *Data Independent.* The most naïve approaches to model pairwise potentials are the Ising model [Bla+04], suited for binary classification problems and its generalization for multi-class classification – Potts model [Wer+11]. Both approaches guarantee smooth label maps, require no training and due to these reasons are very popular in unsupervised classification, but also find application within CRFs [CSK13].

- *Contrast Sensitive.* Contrast sensitive approaches are usually based on Ising or Potts models, but take the similarity of adjacent observations in account [BJ01]. This allows to preserve objects' borders in a label map. As the similarity metric, one can use *e.g.* the Euclidean distance between the colors of adjacent pixels [FH04], or between multidimensional features in a feature space [SM00].

  Above mentioned approaches are very sensitive to edges, treating them equally and ignoring the fact that classifying objects may also contain edges, which do not indicate the objects' borders. Prasad *et al.* made use of small regions around edges in order to distinguish edges, belonging to the objects' borders and to the object itself for pruning the latter [Pra+06]. Such an approach allows to preserve detail in low-variability image regions while ignoring detail in high-variability regions.

  More recently, Kohli *et al.* proposed a method, which is based on higher order CRFs and uses potentials defined on image segments, generated by an unsupervised segmentation algorithm [KLT08]. These potentials enforce label consistency in image regions and can be seen as a generalization of the commonly used contrast sensitive pairwise potentials.

- *Data Dependent.* Data dependent pairwise potentials incorporate a trained function, whose parameters are estimated from the training data. Usually, the first thing to be estimated is the prior probability of labels co-occurrence at neighboring nodes. One method for this is to generate a 2D histogram of such co-occurrences [KRH12]. In order to make the potential also contrast sensitive, the diagonal elements of this histogram are multiplied by a weight, depending on a similarity metric, which is calculated from corresponding test observations in a way similar to [SM00].

  Alternatively, as an extension of [Pra+06], an algorithm to learn *class-specific* contrast sensitive pairwise potentials was presented in [BSC08]. These pairwise potentials are learned during the learning stage as a function of two feature vectors and the classes to which the corresponding pair of nodes may belong. This allows to learn specific sensitivity to contrast, for each object class, to increase the accuracy of its border preservation. Authors of [GZT11] suggested a generalization of the pairwise potentials design, which concatenates a pair of classifiers for two nodes and thus, depends on both label differences *and* their observed data.

Local-Global CRF. The flexibility of modeling dependencies between random variables in CRFs gave birth to many classification models, based on CRFs. Particularly, the dense CRF model in [KK11] can handle a variety of dependencies for all possible pairs of random variables (pixels). Although the exact probabilistic inference of such a model is infeasible,

the researchers use a mean field approximation and high-dimensional filtering to make the inference sublinear in the number of pairwise dependencies. Also, instead of popular l_2 norm, the CRF model in [Son+15] leverages the l_1 norm to regularize model parameters, to enhance the robustness to outliers and the effectiveness for high-dimensional features. Among existing CRF models, there is one that combines local and global features [Lis+05]. In the local-global CRF, the former characterizes objects in a bottom-up fashion, while the latter reflects top-down prior knowledge about their overall characteristics. The usefulness of CRFs for incorporating local and global random variables within a solid graphical model can scarcely be overestimated.

Model Control Parameters and Loss Functions. Despite all the advantages of conditional random fields, they have limited expressive power and often cannot represent the posterior distribution correctly. While learning such models, which have insufficient expressivity, one may use loss functions to penalize certain misrepresentations of the solution space. In previous work on this topic were used simple choices of the loss function, such as the Hamming loss or squared loss, which lead to tractable learning algorithms [DP97]. However, many real world applications require more general loss functions, training of which, until recently, was an intractable problem.

In order to solve this problem, Joachims used an SVM for the loss function optimization [Joa05], while Szummer *et al.* utilized fast inference algorithms to learn *control parameters* of random fields with high efficiency [SKH08]. This learning was formulated as minimizing a loss function on the training set. More recently Pletscher and Kohli incorporated higher-order loss functions in the learning process and proposed an efficient learning algorithm for them [PK12]. These loss functions ensure that the *maximum posterior*(MAP) solution does not deviate much from the ground truth in terms of certain higher-order statistics. The above mentioned methods were developed for the binary classification problems only.

> The detailed discussion of the preceding seven concepts – three for the unary-, three for the pairwise potentials and one for the loss functions – forms the basis of the systematic toolkit for the design of accurate CRF methods in this thesis. Their combination with novel own ideas will allow the construction of the currently most precise classification techniques in the literature.

1.3.2 Modeling of Multi-Layer Conditional Random Fields

In contrast to the modeling of conditional random fields approaches that is a fruitful field of research since decades, only a few works exist in the literature that deal with the occlusions

in classification and construction of multi-layered approaches for conditional random fields techniques.

Additional Layers. In the literature exist a few attempts to include multiple layers of class labels in CRFs. In [KH05] and [Sch+09], multiple layers represent a hierarchical object structure, *i.e.* each object on higher level interacts with its smaller parts on lower level. In [WS06], the part-based model is motivated by the method's potential to incorporate information about the relative alignment of object parts and to model long-range interactions. Such a part-based approach has no universality: it is not applicable *e.g.* to a class of objects such as roads in near-vertical views. Roads do not consist of parts having a specific appearance and appearing in a fixed spatial structure. Besides, the spatial structure of such part-based models is not rotation-invariant and, thus, requires the availability of a reference direction (the vertical in images with a horizontal viewing direction), which is not available in remote sensing imagery. As a consequence, methods relying on such a reference direction are not applicable to this class of images.

In [LOL09] authors apply CRF to a given DSM data in order to estimate the *digital terrain model* (DTM), which, in contrast to a DSM, represents the bare ground surface without any objects like plants and buildings. For this purpose, authors associate each DSM point with two random variables: one discreet and one continues. These variables form two layers, representing a binary map, to distinguish ground from non-ground points and continues elevation of the bare ground surface in the DTM. In [YC07], MRF is also expanded by additional layers in the temporal domain, related to the previous and subsequent frames in a video sequence. The interactions between these temporal layers are designed for the detection of moving objects.

In [HZC04] the additional two layers are modeled implicitly, such that resulting three layers correspond to individual classification models, providing information from different aspects of the image: one for standard local image classification, one for local label patterns and one for large, coarse label patterns. Since one additional layer focuses on fine resolution patterns while the second on global structures, the spatial relationships between objects are taken in account. This model represents the large-scale interactions directly and devotes resources for modeling the label space, but not the image space. However, in all these papers the additional layers do not explicitly refer to occlusions, but encode another label structure. Occlusions are not dealt within these publication.

On the other hand, some layered models were developed for revealing occluded areas [Yan+12; CLY15; TNL14]: in [Yan+12] the layers of a generative model are associated with the different outcomes from an object detector; in [CLY15; TNL14] the task of object segmen-

tation with occlusion handling is solved with help of a compound pipeline of object detectors and segmentation algorithms.

Occlusions. Previous work on the recognition of partially occluded objects may be separated into two domains by the applied methods. To the first domain one ascribes depth-driven methods, and to the second – part-based methods.

To the domain of the depth-driven methods belongs [HG10; KKS13; Nur+16] and [LKF16]. In [HG10] a method to separate semi-transparent objects in X-Ray images is proposed. This method mostly relies on the disparity maps, estimated from multi-view images. In [KKS13] and [LKF16] authors present Voxel-CRF and Occlusion-CRF models, respectively and in [Nur+16] classification is based on SVMs. The data terms of these models are based on the depth information, encoded in RGB-D images. The depth-driven methods domain is very categorical and requires special hardware as stereo cameras or laser depth measurement instruments. The evaluation of the methods, described in the above-mentioned works is done only for the indoor scenes. For dynamic outdoor scenes the depth estimation is often inaccurate or even impossible. Additional depth information may still be useful for the proposed approach, but it will be shown that it is not a necessity.

Domain of the part-based methods includes [LLS08], where the objects in the scene are represented as an assembly of parts. The method is robust to the cases where some parts are occluded and, thus, can predict labels for occluded parts from neighboring unoccluded sites. However, it can only handle small occlusions, and it does not consider the relations between the occluded and the occlusion objects. In [GH12], occluded areas are recovered by fitting geometrical primitives to large background objects using their visible parts, so the whole classification process is supported with additional frameworks, namely contextual prediction [TB10], and non-parametric label transfer [LYT11]. It is worth noting that none of these publications use depth information as an additional cue to deal with occlusions.

Mixed Graphs. In the literature, there are almost no publications incorporating mixed graphs in the CRF framework. However, the graphical model, used by CRFs can be extended in a consistent way to graphs that include both directed and undirected links. These are called *chain graphs* [LW89; Fry90] or *ancestral graphs* [RS02], and contain the directed and undirected edges. Such graphs can represent a broader class of distributions that are either directed or undirected alone.

> In this thesis I will develop suitable multi-layer graphical model for the conditional random fields methods that will be capable of modeling occlusions explicitly via layers, corresponding to objects in the scene, depending on their position to the

observer. In the case when the depth map of the scene is not available, certain ambiguities arising when relating scene objects to the layers, are resolved via utilizing the mixed graphs. This representation forms the basis of my general framework for the construction of efficient multi-layer methods for conditional random fields techniques.

1.4 Organization and Contributions

This thesis makes a significant scientific contribution to the theory of conditional random fields and illustrates how these may motivate new probabilistic models. Different parts of the work presented in this thesis have been published at conferences [Kos08; Kos+09; KTS09; KTS10; KTS11; Kos+12; KRH12; Kos+13a; KRH13; Kos+13b] or journals [Kos+18; KSG18b; Bra+14]. The presented algorithms have been implemented in the *direct graphical models* (DGM) C++ library [Kos15] and some of them were also contributed to the OpenCV library [BK08]. With respect to a better readability the main scientific contributions are split into two chapters – one chapter on standard conditional random fields and one on the multi-layer model for its application to occlusion problems.

Conditional Random Fields

In Chapter 2, a brief overview of pairwise graphical models is given. The building blocks of such models are unary and pairwise potential functions. These functions are trained and operate, using real-world data, represented via extracted features. The problem of over-smoothing the classes, especially those which represent a relative small amount of pixels in an image are considered by studying a special *DCNN* and *confidence features*. The DCNN features are achieved using deep learning techniques and the confidence feature is reconstructed directly from the output of an object detector.

Further, the classical and well-established probabilistic models like Bayes [LD05], Gaussian Mixture [Rey09], Random Forest [Bre01], K-Nearest Neighbors [Alt92], *etc.* are considered for unary potentials. In order to reduce the memory consumption and to speed up training, a new sequential learning scheme of the parameters of Gaussian mixture models was proposed by me [KRH13]. This sequential scheme is proved to be considerably faster than expectation maximization. In addition a new efficient implementation of the K-Nearest Neighbors algorithm, which significantly speeds up the classification was also proposed. For the pairwise potentials, based on contrast-sensitive Potts model and histogram matrices [KRH12], a new data-dependent approach was developed. This approach is constructed by concatenation of two classifiers similar to those, which are used for unary potentials.

The methods for inference (Section 2.5) and random model training (Section 2.6) lie also in focus of this chapter. In scope of these methods the possibility of loss functions training is investigated. Finally, the Chapter is concluded with the experiments section, where extensive experimental evaluation with various synthetic and real-world scenarios was performed in order to investigate the advantages and shortcomings of the proposed prototypes. These experiments will accompany the reader through the detailed discussion and visualize the impact of all concepts on the overall result in both a quantitative and a qualitative ways. Thereby in particular the novel approaches yield excellent results. In particular the challenging task of EM classification is considered, where a full-automatic EM classification system is proposed. This system incorporates pixel-level features and auxiliary global features into a CRF framework.

In order to handle the small dataset problem one of the most successful DCNNs, called VGG-16 [SZ14] was re-purposed by training on 1.3×10^6 images in ImageNet dataset [Den+09], and fine-tuning using EM images (see Figure 1.5) [Kos+18]. For an image, feature maps output by the second last layer of the re-purposed VGG-16 are used to generate pixel-level features. Then extracted features together with the ground truth data are used to train unary classifiers by analyzing pixel-level and global features in training images. The trained unary classifiers are used as the unary potentials by the CRF model. Finally, together with the pairwise potentials, the CRF model is applied to localize and label into the classes the objects of interest in the test EM images. The proposed framework significantly improves the classification rate by combining global and pixel-level features.

The following list sums up all scientific contributions from Chapter 2:

- The study of application new types of features to the classification with CRFs (Section 2.2): **1.** *Global* features; **2.** *DCNN* features; **3.** *confidence* features.

- The development of new highly-efficient and accurate algorithms for unary and pairwise CRF potentials: **1.** *sequential GMM* algorithm (Section 2.3.1.3); **2.** *efficient KNN* algorithm (Section 2.3.2.1); **3.** *data-dependent* interaction models for pairwise potentials (Section 2.4.2).

- The introduction of the *local-global CRF* framework and demonstrating it on the challenging problem of EM classification. (Section 2.1.4).

Multi-Layer Conditional Random Fields

In Chapter 3 in order to increase the classification performance in the occluded areas, my framework of Chapter 2 is exploited and a multi-layer approach of constructing conditional

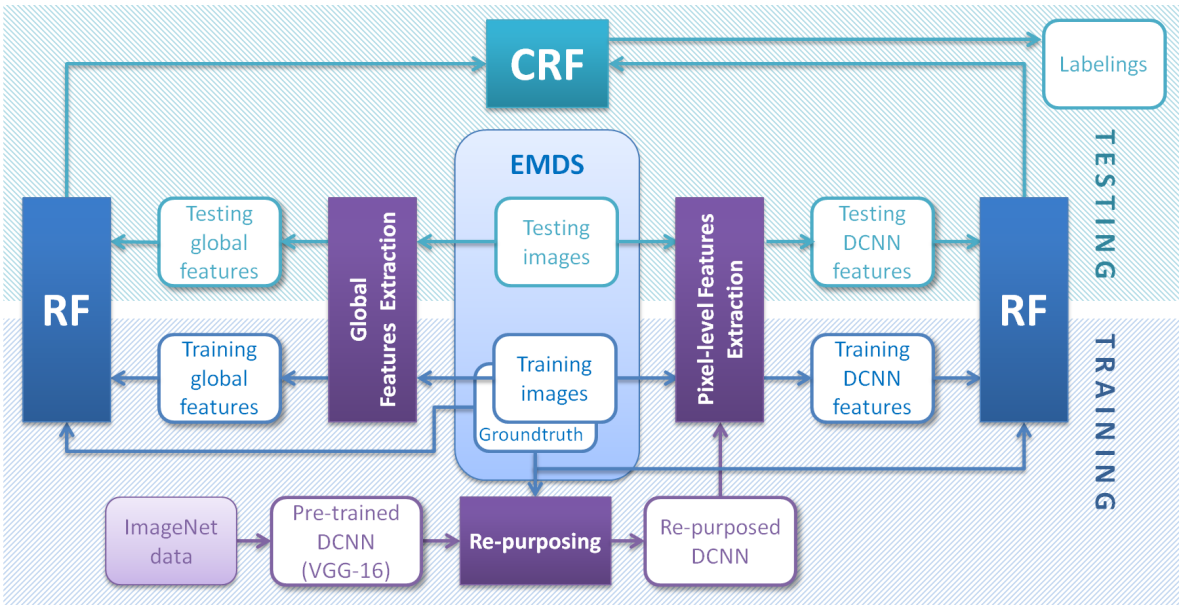


Fig. 1.5: An overview of the proposed CRF-based EM classification and segmentation framework. As an unary classifier the Random Forest (RF) is shown.

random fields is presented. The chapter starts with a description of my occlusion model. After that, in Section 3.5, as a transitional step, the *two-layer conditional random field* is introduced. Two-layer CRF explicitly models *two* class labels for each image site: one for the occluded object and one for the occluding one. The relations between the two class labels per site and the mutual dependencies between class labels at neighboring sites in each of the two layers are explicitly modeled. Thus, the information from neighboring unoccluded objects as well as information from the occluding layer will contribute to an improved labeling of occluded objects.

Two layers are sufficient for many applications, nevertheless, this approach is generalized to an n-layer model in Section 3.3. In order to make my multi-layer model self-consistent, I resort to the help of mixed graphs, so that the interactions between layers are modeled with the help of directed links. My model does not need additional foreground object detectors to separate the foreground from the background level. The information from neighboring unoccluded objects as well as from the occluding layer will contribute to an improved labeling of occluded objects, t least under the assumption that occluded objects show some spatial continuity. Classification might also be supported by depth information obtained from image matching.

The proposed methods are demonstrated on the task of labeling urban scenes (street-view as well as vertical-view images). Not surprisingly, this theoretically sound modeling leads to excellent qualitative results. Various quantitative and qualitative experiments show that

the proposed multi-layer approach does not only outperform classical one-layer conditional random fields from Chapter 2 in most cases, it currently even produces the most accurate label maps in the literature, particularly for the occluded areas.

The two main scientific contributions of Chapter 3 are:

- The introduction of *multi-layer graphical models*, based on mixed graphs and capable to label unobserved data (Section 3.3).
- The introduction of the *two-layer CRF* framework as a common case of the multi-layer CRF, but based on the classical undirected graphical models (Section 3.5).

2 Conditional Random Fields

"The theory of probability as a mathematical discipline can and should be developed from axioms in exactly the same way as geometry and algebra."
- Andrey Nikolaevich Kolmogorov

This chapter discusses the modeling of conditional random fields, using graphical models. As indicated in the introduction, such methods are very useful for estimating posterior distribution over the labellings because of its flexibility in modeling dependencies between the labels and the image data. These dependencies are commonly described by potential functions, which form a joint probability function. We compute the desired label map as a maximizer of this probability function.

The potential functions, in general, may consider any number of nodes included in a clique, but in this thesis for the sake of simplicity only the cliques with maximal two nodes are considered. This results in only two types of potentials: unary potentials, which describe dependency of a label on image features, without considering other labels; and pairwise potentials, which describe the relationship between labels with considering corresponding to these labels image features. Thus, unary potentials substitute the data-term of the joint probability function and the pairwise potentials - the smoothness term.

In the first part of this chapter the design of data and smoothness terms for this particular family of random fields are investigated. Starting from the general formulation of the probabilistic graphical models in Section 2.1 I describe and analyze in detail data features in Section 2.2, unary potentials in Section 2.3 and pairwise potentials in Section 2.4. I believe that the features, describing the observed data, play a crucial role for a successful application of the unary potentials. Thus criteria for feature importance are investigated. By discussing the unary potentials, I address in particular the meaning of graph nodes and show how these potential functions can be modified in such a way that they become faster in training and more accurate. In this context, different concepts such as generative and discriminative classification are discussed. Application of pairwise potentials is the key idea of conditional random fields, so I start from naïve Potts model for pairwise potentials, which requires no training procedure and is independent from observations. Then I exploit pairwise potentials to be first contrast sensitive and finally also trained on the train data. Thereby in particular the meaning of graph edges is addressed.

The scientific contributions to the field of conditional random fields all belong to the first part of this chapter and they are:

- *Global features* which support state-of-the-art *local* features when a rough segmentation of the object of interest is available;
- *DCNN features* extracted using deep learning techniques and extremely helpful when the objects of interest is hard to describe with state-of-the-art features;
- *Confidence features* generated with the help of an object detector and useful for preserving small-classes from over-smoothing effect of CRFs;
- *Sequential algorithm for Gaussian mixture learning.* Being a powerful tool, Gaussian mixture model is very inconvenient for a real-world application because of its training complexity, which takes a lot of computational efforts. Here, a novel sequential algorithm for training Gaussian mixture model, which allows fast and accurate estimation of the its parameters is proposed;
- *Efficient algorithm for K-Nearest Neighbors implementation.* Being extremely simple and accurate, the KNN model is also very inconvenient for a real-world application because the time required for the nearest neighbor search increases exponentially with the number of dimensions in the feature space. In this thesis a more efficient search algorithm based on binary trees is proposed;
- *Concatenated pairwise potentials* are general data-dependent interaction models that can use unary potential functions as underlying models;
- *Local-global CRFs* which can combine state-of-the-art local features with global features in a solid classification framework which is proposed in this thesis. Thus local-global CRFs may benefit from additional top-down prior knowledge about objects overall characteristics.

In the second part of this chapter inference (decoding) and parameter estimation for the built probabilistic model are described. Inference, *i.e.* computing exact posterior distribution of the labels is computationally intractable, so approximate methods ought to be applied. In Section 2.5 approximate inference methods, which are capable of computing both marginal and posterior probabilities are described. All the considered methods belong to the family of the message passing algorithms, which are possible to apply to graphs with cycles. Furthermore it is described how to assign proper class labels from the estimated probabilities of all observations being particular classes. This leads to the consideration of the decision theory, which describes how to minimize the number of wrong labels, considering the posterior

distribution of the labels. The use of a loss function may increase the classification accuracy for particular classes at the price of decreasing the classification accuracy for the rest of the classes.

The influence of data and smoothness terms on the energy functional is regularized by model parameters. Learning the parameters is usually done by maximum likelihood learning. It is possible to show that under certain circumstances this optimization is convex [SM12]. This in turn means, that only one (unique) solution exists which can be found by any globally convergent algorithm. Moreover, well-posed results can be established that show the continuous dependency of the solution on both the input data and the model parameters. To this end, in Section 2.6, the model control parameter estimation techniques, which are an important part of the whole conditional random fields approach were used. It is shown in detail how the different parameter estimation techniques can be derived and applied. Moreover, it is discussed how the weighted information of unary and pairwise potentials can be used to improve the quality of labeling with the conditional random fields.

The chapter is concluded with the evaluation part in Section 2.7, where the performance of the described state-of-the-art and novel approaches are compared. Recommendations, which are the most suited cases for different approaches to be applied are also made.

2.1 Graphical Models

In this section, we shall focus on the key aspects of graphical models as needed for applications in pattern recognition and machine learning. More general treatment of graphical models can be found in the books of [Lau96; Jen96; CGH97; Jor99; Cow+07; Whi09].

The underlying probability distributions of probabilistic models for sake of simplicity and flexibility of modeling are usually represented in a graphical form, this is why they are often called *probabilistic graphical models*. A probabilistic graphical model is a diagrammatic representation of a probability distribution. In such a graph there is a node for each random variable and relations between these variables are represented via graph links.

Representation of probability distributions as a *complete graph*, *i.e.* when each node is linked with all other nodes, is applicable to any choice of distribution. However, it is the *absence* of links in the graph that conveys interesting information about the properties of the class of distributions that the graph represents. The absence of an edge between two nodes represents *conditional independence* between corresponding variables (refer to Appendix B.2). From such graphs, one can read the conditional independence properties of the underlying distribution. Thus a fully connected graph does not contain any information about the probability distribution, only the absence of edges is informative.

Conditional independence is an important concept as it can be used to decompose complex probability distributions into a product of factors, each consisting of the subset of corresponding random variables. This concept makes complex computations (which are for example necessary for learning or inference) much more efficient.

In Section 1.2.1 we have defined a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as the structure, build upon nodes $\mathcal{V}$, which represent random variables and edges $\mathcal{E}$, that defines relations between them. In the sequel, we will distinguish two types of nodes: first, corresponding to the *observed variables* $y \in \mathcal{Y} \subset \mathcal{V}$, that stay for input observations, and second – to the *latent variables* $x \in \mathcal{X} \subset \mathcal{V}$, that stay for output class variables; arrays $\mathcal{X}$ and $\mathcal{Y}$ are non-intersecting. Hence observed and latent variables are also often called *data* and *class* random variables, respectively. Also a graph can have directed or undirected edges, depending on the kind of graphical model it represents. At graphical representations (*e.g.* the one shown in Figure 2.1), we will sketch observed and latent variables via blue squares and red circles correspondingly, while directed edges will be sketched via arrows, pointing to the independent variables. When the difference between x and y is not important, we will reference both types of variables through v.

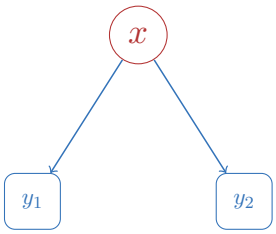


Fig. 2.1: Example directed graphical model, comprising three random variables.

In order to illustrate how the graphical models are bind to the probability distributions, let us consider a model, which is common in data fusion domain and which represents the distribution $p(x, y_1, y_2)$. Figure 2.1 shows one possible factorization of this distribution as $p(x, y_1, y_2) = p(x \,|\, y_1, y_2) \cdot p(y_1) \cdot p(y_2)$. Here, y_1 and y_2 are conditionally independent given x[1], so we can write $p(x, y_1, y_2) = p(y_1 \,|\, x) \cdot p(y_2 \,|\, x) \cdot p(x)$. Thus, the distribution has the following factors: $p(y_1 \,|\, x) \propto \varphi(y_1) \cdot \psi(y_1, x)$, $p(y_2 \,|\, x) \propto \varphi(y_2) \cdot \psi(y_2, x)$, and $p(x) \propto \varphi(x)$. Please note, that this factorization is identical to the factorization of the same undirected graph structure (Appendix B.5), the difference play role only for inference.

Let us now briefly discuss some nuances of utilizing two major classes of graphical models, namely directed and undirected graphical models. The application of mixed graphs will be discussed later in Section 3.2.

[1] Because of the absence of an edge between variables y_1 and y_2.

2.1.1 Directed Graphical Models | Bayesian Networks

A joint distribution $p(\mathbf{v})$ can be factorized into the product of conditional distributions for each node v_i, so that each such conditional distribution is conditioned on its set of parent nodes $\mathbf{v}_i^p$:

$$p(\mathbf{v}) = \prod_{i \in \mathcal{V}} p(v_i \,|\, \mathbf{v}_i^p) \tag{2.1}$$

This is the same kind of factorization as shown in Figure 2.1 for the example distribution $p(x, y_1, y_2)$, where node x acts as the parent node for y_1, y_2.

This key equation expresses the factorization properties of the joint distribution for a directed graphical model. It is easy to show that the representation on the right-hand side of Equation 2.1 is always correctly normalized provided the individual conditional distributions are normalized. However this family of graphical models has an important restriction – it should not contain any directed cycles, *i.e.* there is no such way that we can move from node to node along links following the direction of the arrows and end up back at the starting node. Such graphs are also called *directed acyclic graphs* (DAGs).

2.1.2 Undirected Graphical Models | Markov Random Fields

In directed graphical models a factorization of the joint distribution over a set of variables into a product of local conditional distributions was specified and a set of conditional independence properties was defined. In contrast to them, in undirected graphical models, factorization of a probability distribution is defined as the product of its factors φ_c, with $\mathbf{v}_c$ – the subset of the respective random variables constituting such a factor (Equation 1.1):

$$p(\mathbf{v}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \varphi_c(\mathbf{v}_c)$$

where Z is a normalization constant, which id defined by Equation 1.2.

In contrast to directed graphs where the joint distribution is factorized into a product of conditional distributions, in undirected graphs factors φ_c do not necessarily have to be probability functions. This generalization causes one important consequence – the explicit introduction of the partition function Z, which presence is the major limitation of undirected graphs. If we have model with n nodes and $|\mathbb{L}|$ classes, then the evaluation of the partition function involves summing over $n^{|\mathbb{L}|}$ states so is exponential in the size of the model. The partition function is needed for parameter learning, however we can work with the unnormalized joint distribution and then normalize the marginals explicitly at the end of decision process.

It is possible to generalize this construction, so that we can convert any distribution specified by a factorization over a directed graph into one specified by the factorization over an undirected graph. This can be achieved if the clique potentials of the undirected graph given by the conditional distributions of the directed graph [Bis06].

2.1.3 Conditional Random Fields

We have already discussed some well-known probabilistic models from a mathematical perspective. Moreover, we have shown the graphical representation, which characterizes the underlying probability distribution of these models. In the following, the idea and theoretical foundation of conditional random fields is illustrated. First, a general formulation of random fields is given followed by an in-depth discussion of the conditional random fields. A main focus is aspects of modeling.

We address the general problem of learning a mapping from input observations $y \in \mathbb{Y}$ to discrete response variables $x \in \mathbb{X}$, based on a training sample of input-output pairs (x_1, y_1), $\ldots$, $(x_n, y_n) \in \mathbb{X} \times \mathbb{Y}$ drawn from some fixed but unknown probability distribution. As we address image classification, let us assume an image $\mathbf{y}$ to consist of n image sites (pixels or segments) $i \in \mathcal{V} = \{1, 2, \ldots, n\}$ with observed data y_i, *i.e.*, $\mathbf{y} = (y_1, y_2, \ldots, y_n)^\top$, where $\mathcal{V}$ is the array of all sites, corresponding to the nodes of an associated graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, whose edges $\mathcal{E}$ model interactions between adjacent sites. With each site i we associate also a discrete class variable $x_i \in \mathbb{X}$ which takes values from a given set of classes $\mathbb{L} = \{l_1, l_2, \ldots, l_k\}$. [2]

According to [LMP01] *conditional random fields* are probabilistic models for computing the posterior probability $p(\mathbf{x}|\mathbf{y})$ of a possible output $\mathbf{x} = (x_1, \ldots, x_n)^\top \in \mathbb{X}^n$ given the input $\mathbf{y} = (y_1, \ldots, y_n)^\top \in \mathbb{Y}^n$. A CRF in general can be derived from Bayes Law and Equation 1.1, so the conditional probability $p(\mathbf{x}|\mathbf{y})$ can be written as

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}) &= \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{y})} \\ &= \frac{p(\mathbf{x},\mathbf{y})}{\sum_{\mathbf{x}'} p(\mathbf{x}',\mathbf{y})} \\ &= \frac{\frac{1}{Z'}\prod_{c\in\mathcal{C}} \varphi_c(\mathbf{x}_c;\mathbf{y})}{\frac{1}{Z'}\sum_{\mathbf{x}'}\prod_{c\in\mathcal{C}} \varphi_c(\mathbf{x'}_c;\mathbf{y})} \end{aligned} \tag{2.2}$$

Again φ_c are the different potential functions, which correspond to maximal cliques in the

[2] Please note that we use the same symbols for designating both graph nodes $y_i \in \mathcal{Y} \subset \mathcal{V}$ and random variables $y_i \in \mathbb{Y}$; $i \in \mathcal{V}$. We do this intentionally in order to accentuate the interconnection between these two concepts. However, one should be aware that a graph node y_i is usually associated with a site i, what does not lead to $y_i \equiv i$.

graph (see [KFL06]). From this, the general model formulation of CRFs is derived:

$$p(\mathbf{x}\,|\,\mathbf{y}) = \frac{1}{Z}\prod_{c\in\mathcal{C}}\varphi_c(\mathbf{x}_c;\mathbf{y}) \tag{2.3}$$

The normalization follows from the denominator of equation 2.2

$$Z = \sum_{\mathbf{x}}\prod_{c\in\mathcal{C}}\varphi_c(\mathbf{x}_c;\mathbf{y}) \tag{2.4}$$

Restricting ourselves to pairwise interactions, we can rewrite the Equation 2.3 in the following form:

$$p(\mathbf{x}\,|\,\mathbf{y}) = \frac{1}{Z}\prod_{i\in\mathcal{Y}}\varphi_i(x_i;\mathbf{y})\prod_{(i,j)\in\mathcal{E}}\psi_{ij}(x_i,x_j;\mathbf{y}). \tag{2.5}$$

In Equation 2.5 unary potentials $\varphi_i(x_i;\mathbf{y})$ associate the observations with the label variables at site i, and thus are often called *association potentials*; pairwise potentials $\psi_{ij}(x_i,x_j;\mathbf{y})$ model the interaction of the label variables at two adjacent sites i and j and the data $\mathbf{y}$, and thus are also called *interaction potentials*; and Z is the *partition function*, given by Equation 2.4.

Finally, we can formulate the problem of image classification as finding the label configuration $\tilde{\mathbf{x}}$ that maximizes the posterior probability of the labels given the observations $p(\mathbf{x}\,|\,\mathbf{y})$:

$$\tilde{\mathbf{x}} = \operatorname*{argmax}_{\mathbf{x}} p(\mathbf{x}\,|\,\mathbf{y}). \tag{2.6}$$

Applications of the CRF model differ in the way they define the graph structure and in the models used for the potential functions. The main aim of such models is to emulate the probabilistic dependencies between observations and classes. Usually, each of the models has a certain set of parameters, estimated from fully labeled training images during the training phase. As soon as the parameters of a model are estimated, we call the potential function to be *trained*. We discuss various approaches for modeling and training unary and pairwise potentials in Sections 2.3 and 2.4 correspondingly.

In many practical applications, CRFs have lack of expressivity and a groundtruth label map may not be the solution of the Equation 2.6. In order to handle this problem and get more control over the classification process with CRFs, we may introduce additional *control parameters* θ to the model 2.5:

$$p(\mathbf{x}\,|\,\mathbf{y},\boldsymbol{\theta}) = \frac{1}{Z}\prod_{i\in\mathcal{Y}}\langle\varphi_i(x_i;\mathbf{y}),\boldsymbol{\theta}_\varphi\rangle\prod_{(i,j)\in\mathcal{E}}\langle\psi_{ij}(x_i,x_j;\mathbf{y}),\boldsymbol{\theta}_\psi\rangle\,. \tag{2.7}$$

Here, we gathered parameters, related to the data and smoothness terms separately in vectors

$\boldsymbol{\theta}_\varphi$ and $\boldsymbol{\theta}_\psi$, correspondingly.

In contrast to the internal parameters of the potential functions, parameters θ are estimated separately, after the potentials are trained. This results in an additional second training phase. We discuss parameter θ estimation in detail in Section 2.6.

2.1.4 Local-Global CRFs

In order to incorporate the global features to our model, we add one more extra graph node, initialized with the potentials, trained on the global features [Kos+18]. This 'global' node is connected with all other 'local' nodes via graph edges, initialized with the interaction potentials, represented by a data-independent model (see Section 2.4.1). We describe the global features in more detail in Section 2.2.1. We can rewrite Equation 2.5 in form:

$$p(\mathbf{x}\,|\,\mathbf{y}) = \frac{1}{Z}\,\varphi_g(x_g;\mathbf{y}) \prod_{i\in\mathcal{Y}} \varphi_i(x_i;\mathbf{y})\psi(x_g,x_i) \prod_{(i,j)\in\mathcal{E}} \psi_{ij}(x_i,x_j;\mathbf{y}). \tag{2.8}$$

The structure of our graphical model is depicted in Figure 2.2. The first unary potential $\varphi_g(x_g;\mathbf{y})$ in Equation 2.8 corresponds to the global node x_g (shown with magenta color in Figure 2.2) and provides the CRF model with a single prediction about the object depicted at the image $\mathbf{y}$. The second term $\varphi_i(x_i;\mathbf{y})$ corresponds to the 'local' nodes x_i (red nodes in Figure 2.2), providing per-pixel predictions for every image site $i \in \mathcal{Y}$. This term corresponds to the unary potentials in Equation 2.5. The pairwise potentials $\psi(x_g,x_i)$ correspond to connections between the global node x_g and all local nodes x_i, thus every two local nodes x_i and x_j are also bound through the global node x_g: $x_i \leftrightarrow x_g \leftrightarrow x_j,\ \forall i,j \in \mathcal{Y}$. Finally, the last term $\psi_{ij}(x_i,x_j;\mathbf{y})$ corresponds to the pairwise potentials in Equation 2.5. The partition function Z in Equation 2.8 is represented in the similar way to Equation 2.4.

2.2 Features

For the practical application, the original input data is *preprocessed* to transform it into some new space of descriptors (features) where, it is hoped, the classification problem will be easier to solve. The main idea of this preprocessing is to reduce the variability of input data for each class, and thus to make it much easier for a subsequent classification algorithm to distinguish between the different classes. This preprocessing stage is also called *feature extraction*. Note that new test data must be preprocessed using the same steps as the training data.

The second goal of performing feature extraction is to speed up computation. For example, if the goal is a real-time face detection in a high-resolution video stream, the computer must handle huge numbers of pixels per second, and presenting these directly to a complex

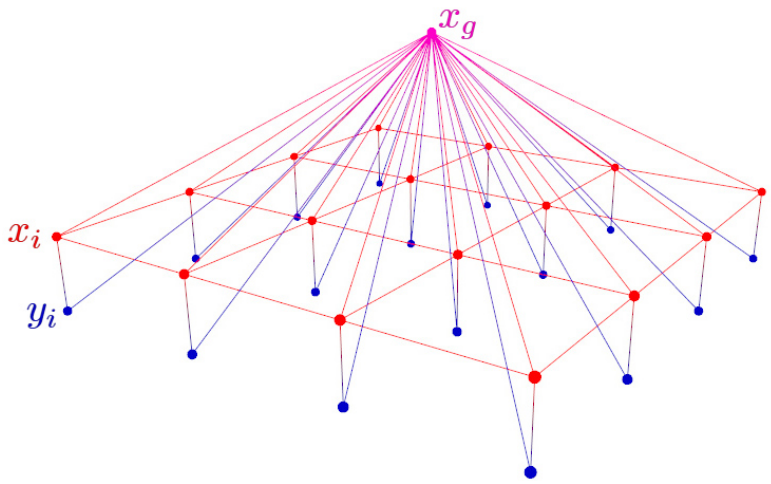


Fig. 2.2: The structure of our graphical model. Blue, red and the magenta nodes correspond to the observation $y_i \in \mathbf{y}$, the labels $x_i \in \mathbf{x}$ and the global node x_g, respectively. Each label x_i is connected with the corresponding observation y_i (unary potentials); and also with four nearest neighbors and with the global node x_g (pairwise potentials). Please note that the global node x_g is also connected with every observation, building the whole image $\mathbf{y}$.

classification algorithm may be computationally infeasible. Instead, the aim is to find useful features that are fast to compute, and yet that also preserve useful discriminatory information enabling faces to be distinguished from non-faces [Kos+09]. These features are then used as the input to the classification algorithm. For instance, the average value of the image intensity over a rectangular sub-region can be evaluated extremely efficiently [VJ04], and a set of such features can rove very effective in fast face detection. Care must be taken during feature extraction because often information is discarded, and if this information is important to the solution of the problem then the overall accuracy of the system can suffer.

Most algorithms describe an individual observation, which category is to be predicted with a *feature vector* of distinct, measurable properties of the observation. Each property is termed a feature, also known in statistics as an *explanatory variable* (or *independent variable*, although in general different features may or may not be *statistically independent*). Features may variously be *binary* (*e.g.* weather a patient has headache or not); *categorical* (*e.g.*"A", "B", "AB" or "O", for blood type); *integer* (*e.g.* the number of leucocytes in a blood test); or *floating point* (*e.g.* a measurement of body temperature).

In our case, when observations are the image sites, the feature values might correspond to the channels of a site, variance of pixel values, *etc.* We derive a feature vector $\mathbf{f}_i(\mathbf{y}) = (f_{i1}, f_{i,2}, \ldots, f_{im})^\top$ for each image site i that consists of m features, derived from the original data $\mathbf{y}$, where $f_{ij} \equiv f_{ij}(\mathbf{y})$. In order to express the feature dependency on the whole data, but not only on a corresponding data site, we make use of *multi-scale* features, thus the derivation

of each feature implies contributes from more data sites (see Figure 2.3). Within this thesis, we consider two types of observations: *local* (y) - a single pixel and *global* ($\mathbf{y}$)- the whole image *in toto*. The excessive description of the local features derived from the datasets, used in this thesis, is given in Appendix D.3 and the description of the global features – in Section 2.2.1.

The *vector space* associated with these vectors is often called the *feature space*. In order to reduce the dimensionality of the feature space, a number of *dimensionality reduction* techniques can be employed.

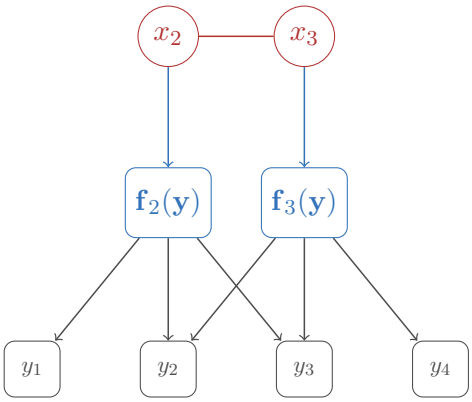


Fig. 2.3: Usage of the multi-scale features for two latent variables, where each corresponding feature vector depends from three neighboring image sites. The second dimension and additional links between data and labels are omitted for simplicity.

2.2.1 Global Features

Many object recognition systems make use of the *global features* that represent an entire image with a single point in a high-dimensional feature space. Such a representation is sensitive to clutter and occlusion, but for the tasks as EM classification where the most of scenes contains a single microorganism, and only in a few scenes several microorganism or particles are present.

We use 7 global features: three shape-driven features *perimeter*, *area* and *compactness* (perimeter squared over area); two Hough-transform-driven features *number of lines* and *number of circles*; and also we make use of *variance* and *opacity* [Kos+18]. In order to extract the shape-driven features we first separate an object of interest from the background by applying a simple global bimodal segmentation. We use expectation maximization to fit a mixture of two Gaussian functions to the histogram of gray values for a given image [DLR77]. The Bayesian decision boundary defines the cut point between the foreground and background. After that, morphological hole filling [Soi03] is used to capture the stray bright pixels inside the object. Next, we apply a Hough transform [DH72] to detect lines and circles in the original images and use the number of corresponding detections as two more features. Finally, we

calculate the variance of image gray-values, and object opacity that is evaluated as

$$\frac{1}{m}\sum_{i=1}^{m}(1-\mathbb{E}_i)\cdot|\mu-y_i|, \tag{2.9}$$

where $\mathbb{E}_i$ is the normalized Euclidean distance between a spacial position of pixel y_i and the image center and μ is the image mean value.

Finally we describe the global observation with one feature vector $\mathbf{f}_g(\mathbf{y})$ consisting of seven global features described above. Thus we can rewrite the unary potential corresponding to the global node x_g in Equation 2.8 as:

$$\varphi_g(x_g;\mathbf{y}) \equiv \varphi_g(x_g;\mathbf{f}_g(\mathbf{y})). \tag{2.10}$$

2.2.2 DCNN Features

Figure 2.4 illustrates an overview of our *DCNN features* extractor [Kos+18] that is an extension of the DCNN-based image segmentation approach, called *DeepLab*, introduced in [Che+16]. First, a pre-trained DCNN *VGG-16* for image classification is adapted to a segmentation model *DeepLab-VGG-16* by re-using/modifying the bottom and middle layers, as depicted by the dashed arrows in Figure 2.4. In addition, three fully connected layers at the top of VGG-16 are replaced with one average-pooling, three convolution and one (bilinear) interpolation layers. The right side of Figure 2.4 provides a more detailed view of these three convolution layers. While the bottom one computes region-wise convolution in 512 feature maps obtained at the average-pooling layer, the other two layers perform 1×1 pixel-wise convolution to enhance the non-linearity of pixel classification. Finally, letting $k\equiv|\mathbb{L}|$ be the number of classes assigned to pixels, k feature maps at the top convolution layer are resized to the original image size using the interpolation layer. These resized feature maps represent a dense segmentation result where every pixel is associated with k scores, expressing how likely the pixel belongs to each class. We consider that, rather than pixel-wise convolution at the top layer, a better segmentation could be accomplished if 1024 feature maps at the penultimate layer would be resized to the original image size and used as pixel-level features for a more sophisticated classifier (*e.g.*, CRF), as shown in the rightmost part of Figure 2.4. This feature extraction is detailed below.

VGG-16 is a DCNN consisting of 16 weight layers (*i.e.*, convolution and fully connected layers) [SZ14]. While a depth is one very important factor for accurate recognition, a deep architecture involves a huge number of parameters, and its appropriate optimization is difficult even using large-scale training data. Compared to this, VGG-16 adopts a very small field of view (3×3) for each convolution filter (see Figure 2.4), so that its deep architecture contains

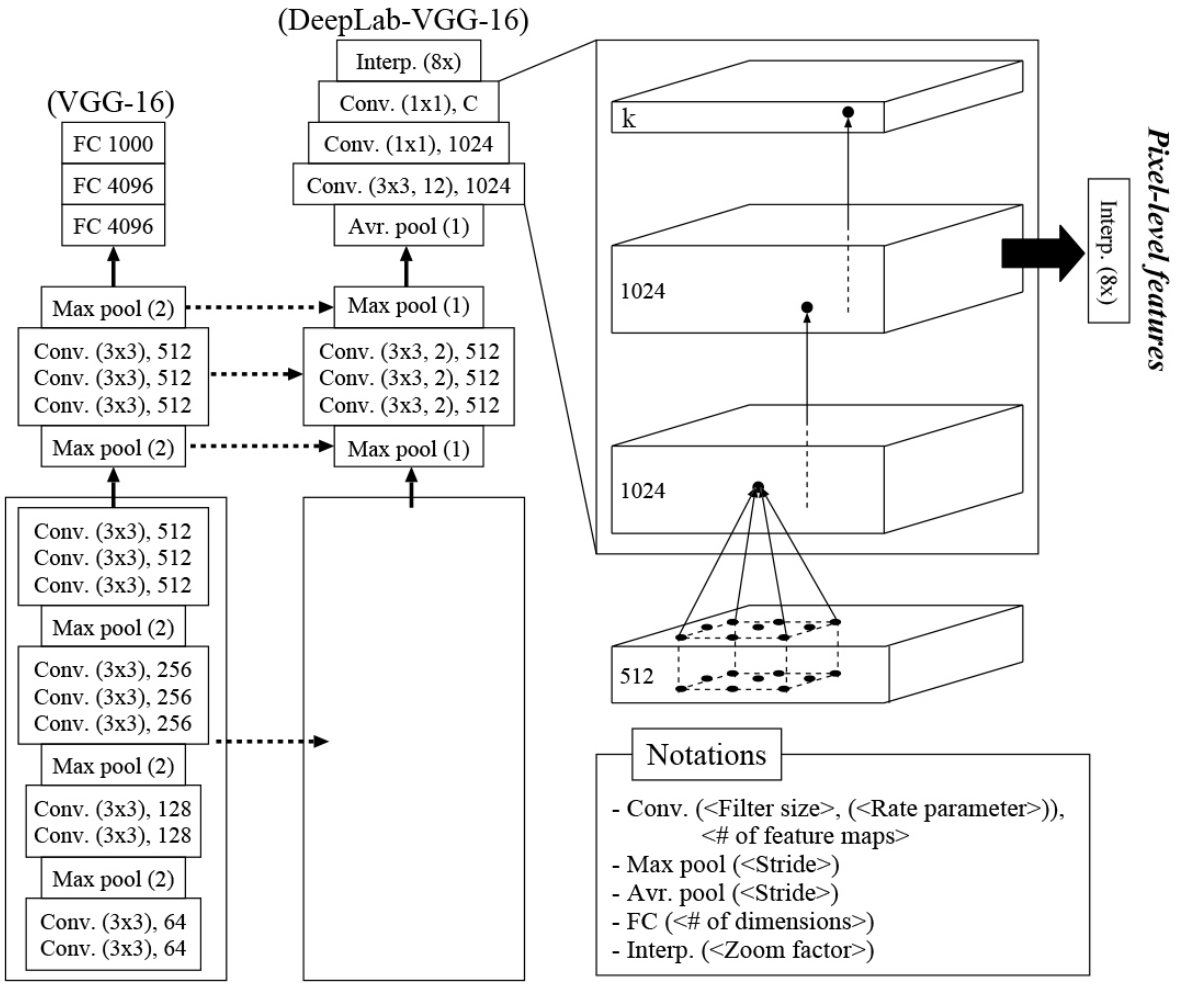


Fig. 2.4: An overview of our pixel-level feature extraction where the pre-trained VGG-16 is re-purposed to DeepLab-VGG-16, and feature maps at the penultimate convolution layer are drawn out as pixel-level features. 'Conv.' indicates a convolution layer where the number in brackets represents the size of the filter. If one more number is included, it represents the rate parameter for atrous convolution. The number in brackets of 'Max pool' or 'Avr. pool' presents the stride size, and the number in brackets of of 'Interp' denotes a zoom factor for bilinear interpolation.

a much smaller number of parameters. This property of VGG-16 is suitable for the small training dataset problem in EM classification. Actually VGG-16 trained on 1.3 million images in ImageNet dataset [Den+09] demonstrated excellent performances in many tasks [SZ14; Che+16; LSD15; Mat+16].

VGG-16 is re-purposed to DeepLab-VGG-16 that aims to effectively maintain the spatial resolution of feature maps. In VGG-16, five max-pooling layers with stride 2 reduce the resolution of feature maps by a factor of 32 compared to the original image (see Figure 2.4), so a lot of detailed information is lost. To take a good trade-off between the accuracy and efficiency, in DeepLab-VGG-16, the stride of the top two max-pooling layers is set to 1,

and feature maps with one-eighth of the original resolution are processed. In addition, the following *atrous convolution* is utilized to efficiently widen the field of view of a convolution filter:

$$f_l(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f_{l-1}(x + r \cdot i,\ y + r \cdot j)\ w(i,j). \tag{2.11}$$

For simplicity, we assume that the l^{th} and $(l-1)^{\text{th}}$ layers have single feature maps f_l and f_{l-1}, respectively (it is straightforward to extend this to multiple feature maps). In Equation 2.11, a convolution filter of size $(2k+1)^2$ is represented by $w(i,j)$. But, based on the 'rate' parameter r, the convolution is done for every r values in f_{l-1} as depicted by the set of small dots in the right side of Figure 2.4. In other words, the filter is dilated by introducing zeros (*i.e.*, holes) for values in f_{l-1} that are excluded from the convolution. This way, the field of view of the convolution filter is enlarged without requiring any extra parameters.

We train DeepLab-VGG-16 using 200 training EM images containing $k = 21$ classes (refer to Appendix D). Then, each EM image is fed into the trained DeepLab-VGG-16, and 1024 feature maps at the penultimate convolution layer are extracted and bilinearly interpolated to the original image size. As a result, each pixel is now represented by a 1024-dimensional feature vector. Figure 2.5 visualizes pixel-level features extracted for three example images in a very simple way, where each pixel is characterized by the index of the dimension having the highest value among 1024 dimensions. Such indexes are then scaled and visualized as an image. As can be seen from Figure 2.5, even with this simple visualization, the region of each EM is outlined, which implies the effectiveness of extracted pixel-level features. It is worth noting that we tested to train DeepLab-VGG-16 using natural images in PASCAL VOC 2012 dataset [Eve+15], but pixel-level features extracted from it were not so useful. It is considered that a general-purpose feature extractor trained on natural images is not suitable for a special type of EM images. Finally, it could be possible to extract further useful features by re-purposing a more advanced DCNN than VGG-16 like ResNet [He+16]. We leave this as a future work because its re-purposing needs much RAM and cannot be performed on our current GPUs.

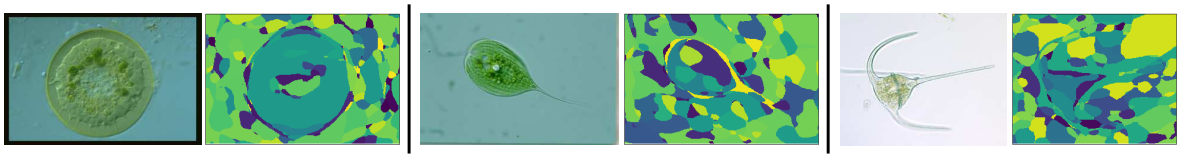


Fig. 2.5: Simple visualization of pixel-level features for three example images.

2.2.3 Confidence Features

As we stated before, the problem of over-smoothing, which is common to the CRF techniques, may considerably affect the resulting label map. In some cases in may even lead to complete vanishing of classes, represented by small objects, like cars in aerial imagery. As a countermeasure we introduce *confidence features* to support the data term of our CRF model [Kos+13b]. These features are supposed to have a high response on image segments, covered by target objects, and low response on the rest of the image area. We generate such response with help of an object detector, what leads to incorporating a part-based models.

Within this thesis we demonstrate the advantage of incorporating the confidence features on example of the class *car* at airborne images. Our car detector makes use of the *histogram of oriented gradients* features, which can be efficiently calculated with help of integral histograms [Por05]. Training and classification is performed using nonlinear SVMs with radial basis functions as kernel. The kernel parameter and error weight of slack variables are estimated on the training data. The membership of each pixel i to target object class given its feature vector $\mathbf{f}_i(\mathbf{y})$, is calculated by:

$$\delta(\mathbf{f}_i(\mathbf{y})) = sign\left(\mathbf{w}^\top \eta(\mathbf{f}_i(\mathbf{y})) + d\right) \tag{2.12}$$

where $\mathbf{w}$ is the normal vector and d the vertical distance to feature space origin of the separating hyperplane in the transformed feature space. Transformation of feature vectors is given by the transform function $\eta(\mathbf{f}_i(\mathbf{y}))$. As we cab see from Equation 2.12 the function $\delta(\mathbf{f}_i(\mathbf{y}))$ gives only a binary decision and, in order to transform it into a probability distribution over classes, we apply *Platt scaling* (refer to Section 2.3.2.2 for more details).

2.3 Association Potentials

The association potential functions $\varphi_i(x_i; \mathbf{y})$ from Equation 2.5 are related to the probability of a random variables x_i taking a label $l \in \mathbb{L}$ given the data $\mathbf{y}$ by [KH06][3]:

$$\varphi_i(x_i; \mathbf{y}) \propto p(x_i = l \,|\, \mathbf{f}_i(\mathbf{y})), \tag{2.13}$$

where the image data are represented by site-wise feature vectors $\mathbf{f}_i(\mathbf{y})$ that may depend on all the observed data $\mathbf{y}$. Both the definition of the features and the dimension of the feature vectors $\mathbf{f}_i(\mathbf{y})$ may vary with the dataset.

[3] $A \propto B$ indicates that A is proportional to B, *i.e.* equal till some constant factor.

2.3.1 Generative Approaches

In practical applications, the most complex approaches to solving decision problems are generative approaches. The core idea, underlying these approaches is to estimate (or generate) *i.e.* using the training data, the class-conditional densities $p(\mathbf{f}(\mathbf{y}) \,|\, x = l)$ for each class $l \in \mathbb{L}$ as well as the prior class probability $p(x = l)$ and then use Bayes theorem:

$$p(x = l \,|\, \mathbf{f}(\mathbf{y})) = \frac{p(\mathbf{f}(\mathbf{y}) \,|\, x = l) \cdot p(x = l)}{p(\mathbf{f}(\mathbf{y}))} \tag{2.14}$$

to find the posterior class probabilities $p(x = l \,|\, \mathbf{f}(\mathbf{y}))$. The denominator in Bayes' theorem can be expressed as

$$p(\mathbf{f}(\mathbf{y})) = \sum_{l \in \mathbb{L}} p(\mathbf{f}(\mathbf{y}) \,|\, x = l) \cdot p(x = l) \tag{2.15}$$

Having found the posterior probabilities, we use decision theory to determine class membership for each new input y. Approaches that explicitly or implicitly model the distribution of inputs as well as outputs are known as *generative models*, because by sampling from them it is possible to generate synthetic data points in the input space.

Estimating the probability density functions $p(\mathbf{f}(\mathbf{y}) \,|\, x = l)$ from a set of training data is typically a very difficult problem whose solution forms the subject of much of this section. Further in this section, some well-known generative probabilistic models are discussed. Conditional random fields are founded on the underlying ideas and concepts of these approaches.

2.3.1.1 Naïve Bayes Model

Modeling all dependencies in a probability distribution is typically very complex due to interdependencies between features. The naïve Bayes assumption of all features $f \in \mathbf{f}(\mathbf{y})$ being *conditionally independent* is an approach to address this problem [DP97]. In order to make necessary computations manageable nearly all probabilistic models have similar independence assumptions for some variables.

Let us consider a probability distribution $p(x \,|\, \mathbf{f}(\mathbf{y}))$ with an input feature vector $\mathbf{f}(\mathbf{y}) = (f_1, \ldots, f_m)^\top$, where f_j, $j \in [1; m]$ are particular features and x is the latent random variable to be classified (predicted). This probability distribution can be reformulated with Bayes law from Equation 2.14. The denominator $p(\mathbf{f}(\mathbf{y}))$, given by Equation 2.15 is not important for classification as it can be understood as a normalization constant which can be computed by considering all possible values for x. The numerator can also be written as a joint probability

$$p(\mathbf{f}(\mathbf{y}) \,|\, x) \cdot p(x) = p(x, \mathbf{f}(\mathbf{y})) \tag{2.16}$$

which can be too complex to compute directly (especially when the number of features m is high). A general decomposition of the joint probability 2.16 can be formulated applying the *chain rule* (refer to Appendix B.3)[4]:

$$p(x, \mathbf{f}(\mathbf{y})) \overset{B.3.2}{=} p(x) \prod_{j=1}^{m} p(f_j \,|\, f_{j-1}, \ldots, f_1, x) \tag{2.17}$$

In order to simplify Equation 2.17, we make use of the naïve Bayes assumption, that all features f_j are conditionally independent of each other, given x. In other words, we assume that $p(f_j \,|\, f_k, x) = p(f_j \,|\, x)$ holds for all $j \neq k$ (refer to Appendix B.2). Based on this simplification, a model known as the *naïve Bayes model* is formulated as

$$p(x \,|\, \mathbf{f}(\mathbf{y})) \propto p(x, \mathbf{f}(\mathbf{y})) = p(x) \prod_{j=1}^{m} p(f_j \,|\, x). \tag{2.18}$$

This probability distribution is less complex than the one, formulated in Equation 2.17. Dependencies between the input features f_j are not modeled, and that may lead to rather poor representations of the class-conditional densities. Nevertheless, its advantage is that the naïve Bayes assumption is helpful when the dimensionality m of the feature space is high, making density estimation in the full m-dimensional space more challenging. It is also very fast to determine in training and the naïve Bayes model also performs surprisingly well in many real world applications, *e.g.* email classification [KM01]. Figure 2.6 graphically represents the naïve Bayes model for three input features. The corresponding probability distribution factorizes as $p(x, \mathbf{f}(\mathbf{y})) = p(x) \cdot p(f_1 \,|\, x) \cdot p(f_2 \,|\, x) \cdot p(f_3 \,|\, x)$.

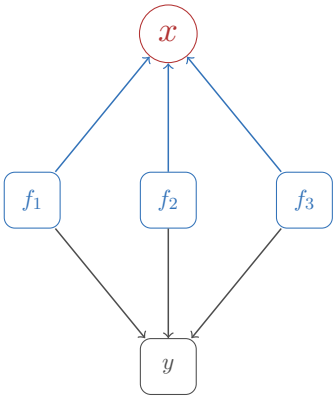


Fig. 2.6: Example graphical representation of the naïve Bayes model for one observation y, which is represented via three features f_1, f_2 and f_3, connected to the class variable x.

Now let us consider the association potential $\varphi_i(x_i; \mathbf{y})$ from Equation 2.5. Having in mind

[4]The initial probability distribution is assumed to be included as $p(f_1 \,|\, f_0, f_1, x) = p(f_1 \,|\, x)$.

the Equations 2.13 and 2.18, we can model it as the naïve Bayes model:

$$\varphi_i(x_i; \mathbf{y}) = p(x_i) \prod_{j=1}^{m} p(f_{ij} \,|\, x_i), \tag{2.19}$$

where f_{ij} is the j^{th} component of $\mathbf{f}_i(\mathbf{y})$.

During the training phase, using feature cooccurrences for different classes, we can estimate one-dimensional probability density functions (histograms) $H_{jl}(f)$ for all features $j \in [1;m]$ and for all classes $l \in \mathbb{L}$. After normalization and smoothing, which results in a vector $h_{jl}(f)$, we can use it directly to model the likelyhood: $p(f_{ij} \,|\, x_i = l) \equiv h_{jl}(f_{ij})$.

Assuming the uniform prior on the class labels and neglecting terms that are constant over the classes we achieve:

$$\varphi(x_i = l, \mathbf{y}) = \prod_{j=1}^{m} h_{jl}(f_{ij}). \tag{2.20}$$

2.3.1.2 Gaussian Models and Gaussian Mixture Models

Gaussian mixture model is one of the most well performing and studied generative model. It is widely used in data mining, pattern recognition, machine learning, and statistical analysis. In many applications, their parameters are determined from training data by maximum likelihood, typically using the expectation maximization algorithm. However, as we will see shortly there are some significant limitations to the maximum likelihood approach, and in Section 2.3.1.3 we will show that a sequential estimation of the GMM's parameters can be given. This requires less computation and memory consumption compared with expectation maximization [KRH13].

Gaussian Distribution. The Gaussian, also known as the *normal distribution*, is a widely used model for the distribution of continuous variables. In a case of a single variable y, the Gaussian distribution can be written in form:

$$\mathcal{N}(y; \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{1}{2\sigma^2}(y-\mu)^2\right) \tag{2.21}$$

where μ is the mean and σ^2 is the variance. For a m-dimensional vector $\mathbf{y}$, the multivariate Gaussian distribution takes the following form:

$$\mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{m/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{y}-\boldsymbol{\mu})\right) \tag{2.22}$$

where $\boldsymbol{\mu}$ is an m-dimensional mean vector, $\boldsymbol{\Sigma}$ is an $m \times m$ covariance matrix, and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$.

Let us also consider the geometrical form of the Gaussian distribution. The functional dependence of the Gaussian on $\mathbf{y}$ is through the quadratic form

$$\Delta^2(\mathbf{y}) = (\mathbf{y} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \boldsymbol{\mu}) \tag{2.23}$$

which appears in the exponent in Equation 2.22. The quantity Δ is called the *Mahalanobis distance* from $\boldsymbol{\mu}$ to $\mathbf{y}$ and reduces to the Euclidean distance when $\boldsymbol{\Sigma}$ is the identity matrix. The Gaussian distribution will be constant on surfaces in $\mathbf{y}$-space for which this quadratic form is constant.

If the Mahalanobis distance serves as a distance metric between a Gaussian and a point, the *Kullback-Leibler divergence* (or relative entropy) could serve as a similarity measure between two Gaussians. The Kullback-Leibler divergence between two probabilita distributions $\mathcal{N}_1(\mathbf{y}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\mathcal{N}_2(\mathbf{y}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ is calculated by the formula:

$$\mathbb{D}_{KL}(\mathcal{N}_1 \,||\, \mathcal{N}_2) = \frac{1}{2} \left(\mathrm{tr}(\boldsymbol{\Sigma}_2^{-1} \boldsymbol{\Sigma}_1) + \Delta_2^2(\boldsymbol{\mu}_1) - m - \ln \left(\frac{|\boldsymbol{\Sigma}_1|}{|\boldsymbol{\Sigma}_2|} \right) \right) \tag{2.24}$$

and expressed in *nats*[5]. Here $\Delta_2^2(\boldsymbol{\mu}_1)$ is the Mahalanobis distance between the centers of multivariate normal distributions $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ with respect to $\mathcal{N}_2$. Please note, that it is not a symmetrical quantity, that is to say $\mathbb{D}_{KL}(\mathcal{N}_1 \,||\, \mathcal{N}_2) \neq \mathbb{D}5_{KL}(\mathcal{N}_2 \,||\, \mathcal{N}_1)$ and so could be hardly used as a "distance".

The Gaussian distribution arises in many different contexts and can be motivated from a variety of different perspectives. For example, according to the *central limit theorem*, the sum of the a set of random variables, which is of course itself a random variable, has a distribution that becomes increasingly Gaussian as the number of terms in the sum increases [Wal69]. We can illustrate this by considering k variables $y_1, y_2, \ldots, y_k$ each of which has a uniform distribution over the interval [0, 1] and then considering the distribution of the mean $(y_1 + y_2 + \cdots + y_k)/k$. For large k, this distribution tends to Gaussian. In practical applications, as it was shown in [YS08] it is already sufficient $k = 12$ variables for approximating a reliable for our tasks Gaussian.

Now suppose that we have n observations, drawn *independently* from an unknown distribution. We will assume that this distribution is a Gaussian, with unknown mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. In order to estimate these parameters we first write the likelihood function for

[5]The natural unit of information is a unit of information or entropy, based on natural logarithms and powers of e, rather than the powers of 2 and base 2 logarithms which define the bit.

the Gaussian in form of the dataset probability:

$$p(\mathbf{f}(\mathbf{y}) \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} \mathcal{N}(\mathbf{f}_i(\mathbf{y}); \boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{2.25}$$

One common criterion for determining the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ in a probability distribution using training dataset is to find the parameter values that maximize the likelihood function from Equation 2.25. In practice it is more convenient to maximize the log of the likelihood function. Because the logarithm is a monotonically increasing function of its argument, maximization of the log of a function is equivalent to maximization of the function itself. Taking the log not only simplifies the subsequent mathematical analysis, but it also helps numerically because the product of a large number of small probabilities can easily underflow the numerical precision of the computer, and this is resolved by computing instead the sum of the log probabilities. From Equations 2.22 and 2.25, the log likelihood function can be written in form

$$\log p(\mathbf{f}(\mathbf{y}) \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{nm}{2}\log(2\pi) - \frac{n}{2}\log|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{i=1}^{n}(\mathbf{f}_i(\mathbf{y}) - \boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\mathbf{f}_i(\mathbf{y}) - \boldsymbol{\mu}) \tag{2.26}$$

Maximizing 2.26 with respect to $\boldsymbol{\mu}$, we obtain the maximum likelihood solution given by

$$\boldsymbol{\mu}_{ML} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_i(\mathbf{y}) \tag{2.27}$$

which is the mean of the feature vectors from training data. Similarly, maximizing 2.26 with respect to $\boldsymbol{\Sigma}$, we obtain the maximum likelihood solution for the covariance in the form [MN99]:

$$\boldsymbol{\Sigma}_{ML} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{f}_i(\mathbf{y}) - \boldsymbol{\mu}_{ML})(\mathbf{f}_i(\mathbf{y}) - \boldsymbol{\mu}_{ML})^{\top} \tag{2.28}$$

which involves $\boldsymbol{\mu}_{ML}$ because this is the result of a joint maximization with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Note, that the solution 2.27 for $\boldsymbol{\mu}_{ML}$ does not depend on $\boldsymbol{\Sigma}_{ML}$, ans so it is possible to evaluate $\boldsymbol{\mu}_{ML}$ first and use it to evaluate $\boldsymbol{\Sigma}_{ML}$.

If we evaluate the the expectations of the maximum likelihood solutions under the true distribution, we obtain the following results:

$$\mathbb{E}[\boldsymbol{\mu}_{ML}] = \boldsymbol{\mu}, \tag{2.29}$$

$$\mathbb{E}[\boldsymbol{\Sigma}_{ML}] = \frac{n-1}{n}\boldsymbol{\Sigma}. \tag{2.30}$$

We see that the expectation of the maximum likelihood estimate for the mean is equal

to the true mean. However, the maximum likelihood estimate for the covariance has an expectation that is less than the true value, and hence it is biased. Note, that the bias of the maximum likelihood solution becomes less significant as the number n of training sample points increases, and in the limit $n \to \infty$ the maximum likelihood solution for the covariance equals the true variance of the distribution that generated the data. In practice, for anything other than small n, this bias will not prove to be a serious problem. We will use the maximum likelihood estimations for the expectation and covariance in Section 2.3.1.3 for deriving the sequential estimation algorithm for the Gaussian mixture model.

Mixture of Gaussians Nevertheless the Gaussian distribution is the most commonly found in nature, it suffers from the lack of generality; *i.e.* it has obvious limitations when it comes to approximating complex distributions. Whereas a linear superposition of number of Gaussians is free from these limitations and in most cases can give us a better characterization of a data set.

Such superpositions, formed by taking linear combinations of more basic distributions such as Gaussians, can be formulated as probabilistic models known as *mixture distributions* [MB88; MP00]. By using a sufficient number of Gaussians, and by adjusting their means and covariances as well as the coefficients in the linear combination, almost any continuous density can be approximated to arbitrary accuracy.

We therefore consider a superposition of G Gaussian densities of the form:

$$p(\mathbf{y}) = \sum_{k=1}^{G} \omega_k \, \mathcal{N}_k(\mathbf{y}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.31}$$

wich is called a *mixture of Gaussians*. Each Gaussian density $\mathcal{N}_k(\mathbf{y}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is called a *component* of the mixture and has its own mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$. The weights ω_k are called *mixture coefficients*. If we integrate both sides of Equation 2.31 with respect to $\mathbf{y}$, and note that both $p(\mathbf{y})$ and the individual Gaussian components are normalized, we obtain

$$\sum_{k=1}^{G} \omega_k = 1. \tag{2.32}$$

Also, given that $\mathcal{N}_k(\mathbf{y}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \geq 0$, a sufficient condition for the requirement $p(\mathbf{y}) \geq 0$ is that $\omega_k \geq 0$, $\forall k \in [1; G]$. Combining this with the condition 2.32 we obtain

$$0 \leq \omega_k \leq 1. \tag{2.33}$$

We therefore see that the mixture coefficients satisfy the requirements to be probabilities,

and so, could be considered as as the prior probabilities of picking the k^{th} component. The densities $\mathcal{N}_k(\mathbf{y}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, in its turn, could be be considered as the probabilities of $\mathbf{y}$ conditioned on k: $p(\mathbf{y} \,|\, k)$. From Bayes' theorem we can write

$$p(k \,|\, \mathbf{y}) \propto p(\mathbf{y}) \cdot p(\mathbf{y} \,|\, k) = \omega_k \cdot \mathcal{N}_k(\mathbf{y}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{2.34}$$

Estimation of the parameters of the mixture components with the mixture weights is not a trivial problem. In some extent, this problem could be simplified to a problem of finding clusters in a set of training data points, which could be solved, for example, by using a non-probabilistic technique called the K-means algorithm [Llo82]. The goal of this algorithm is to partition the training data set into some number K of clusters. Intuitively, we might think of a cluster as a comprising a group of data points whose inter-point distances are small compared with the distances to points outside of the cluster.

But a more general probabilistic technique for finding maximum likelihood estimators is the expectation maximization algorithm.

2.3.1.3 Sequential Gaussian Mixture Model

As we discussed in Section 2.3.1.2, a generative approach, called Gaussian mixture model allows us to approximate almost any continuous probability density function to arbitrary accuracy. In order to utilize it for multi-class classification problems within the CRF framework, we start with the rewriting Equation 2.31 in the form [Rey09]:

$$p(x = l \,|\, \mathbf{f}(\mathbf{y})) = \sum_{k=1}^{G_l} \omega_{lk} \; \mathcal{N}_{lk}(\mathbf{f}(\mathbf{y}); \boldsymbol{\mu}_{lk}, \boldsymbol{\Sigma}_{lk}). \tag{2.35}$$

In Equation 2.35, $\mathcal{N}_{lk}(\mathbf{f}(\mathbf{y}); \boldsymbol{\mu}_{lk}, \boldsymbol{\Sigma}_{lk})$ are the mixture components, *i.e.* Gaussian probability density functions with expectations $\boldsymbol{\mu}_{lk}$ and covariance matrices $\boldsymbol{\Sigma}_{lk}$, and ω_{lk} are the mixture coefficients measuring the contribution of component $\mathcal{N}_{lk}$ to the joint probability density of class l. For each class $l \in \mathbb{L}$ there are G_l sets of parameters ω_{lk}, $\boldsymbol{\mu}_{lk}$, $\boldsymbol{\Sigma}_{lk}$, $k \in [1; G_l]$ which define the form of the Gaussian mixture distribution.

One way to estimate the values of these parameters from the training data is to use expectation maximization algorithm [DLR77; MK08]. It alternates between performing an *expectation* (E) step, which, based on the current estimate for the parameters, determines an estimate for the posterior probability that a specific cluster was responsible for creating each sample point, and a *maximization* (M) step based on a maximum likelihood estimation of the parameters, using the results of the E step as weights for the contribution of each observation to the determination of the parameters of a specific cluster. Expectation maximization algo-

rithm takes a number of iterations in order to reach (approximate) convergence and each cycle requires heavy computation. Moreover, expectation maximization requires the simultaneous storage and processing of all the training samples and the prior definition of the number G of Gaussians in the mixture model [MK08].

In order to overcome these limitations of the expectation maximization algorithm we propose a sequential training method for estimating the GMM parameters [KRH13] (*cf.* Algorithm 1). It is based on two assumptions about incoming sample points[6]: first – that the maximum possible distance between them in feature space is known and second – that the sequence of input sample points has a random order, since our algorithm is sensible to the ordering of incoming samples. The second assumption guaranties us a uniform distribution of the mixture components' centers ($\boldsymbol{\mu}_{lk}$) in feature space.

Our discussion of the maximum likelihood solutions for the parameters of the Gaussian distribution in Section 2.3.1.2 provides a convenient opportunity to give a more general discussion of the topic of sequential estimation for maximum likelihood. Sequential methods allow data points to be processed one at a time and then discarded and are important for online applications, and also where large data sets are involved so that batch processing of all data points an once is infeasible.

Consider the Equation 2.27 for the maximum likelihood estimator of the mean $\boldsymbol{\mu}_{ML}$, which we will denote by $\boldsymbol{\mu}_{ML}^{(n)}$ when it is based on n observations. If we dissect out the contribution from the final sample point $\mathbf{f}_n(\mathbf{y})$, we obtain

$$\begin{aligned}
\boldsymbol{\mu}_{ML}^{(n+1)} &= \frac{1}{n+1}\sum_{i=1}^{n+1}\mathbf{f}_i(\mathbf{y}) \\
&= \frac{1}{n+1}\mathbf{f}_{n+1}(\mathbf{y}) + \frac{1}{n+1}\sum_{i=1}^{n}\mathbf{f}_i(\mathbf{y}) \\
&= \frac{1}{n+1}\mathbf{f}_{n+1}(\mathbf{y}) + \frac{n}{n+1}\boldsymbol{\mu}_{ML}^{(n)} \\
&= \boldsymbol{\mu}_{ML}^{(n)} + \frac{1}{n+1}(\mathbf{f}_{n+1}(\mathbf{y}) - \boldsymbol{\mu}_{ML}^{(n)}). \qquad (2.36)
\end{aligned}$$

This result has a nice interpretation, as follows. After observing n data points we have estimated $\boldsymbol{\mu}$ by $\boldsymbol{\mu}_{ML}^{(n)}$. We now observe data point $\mathbf{f}_{n+1}(\mathbf{y})$, and we obtain our revised estimate $\boldsymbol{\mu}_{ML}^{(n+1)}$ by moving the old estimate a small amount, proportional to $\frac{1}{n+1}$, in the direction of the 'error signal' $(\mathbf{f}_{n+1}(\mathbf{y}) - \boldsymbol{\mu}_{ML}^{(n)})$. Note that, as n increases, so the contribution from successive data points gets smaller.

In the same manner, using the Equation 2.28, we can derive the sequential updating rule

[6] In our case, sample points are given by feature vectors $\mathbf{f}(\mathbf{y})$.

for the covariance:

$$\mathbf{\Sigma}_{ML}^{(n+1)} = \mathbf{\Sigma}_{ML}^{(n)} + \frac{1}{n+1}\left((\mathbf{f}_{n+1}(\mathbf{y}) - \boldsymbol{\mu}_{ML}^{(n+1)})(\mathbf{f}_{n+1}(\mathbf{y}) - \boldsymbol{\mu}_{ML}^{(n+1)})^{\top} - \mathbf{\Sigma}_{ML}^{(n)}\right). \tag{2.37}$$

The function *AddPoint*, described in the Algorithm 5, Appendix C is based on these two Equations 2.36 and 2.37. For the very first data point we use $\boldsymbol{\mu}_{ML}^{(1)} = \mathbf{f}_1(\mathbf{y})$ for initializing the expectation and zero matrix for initializing the covariance: $\mathbf{\Sigma}_{ML}^{(1)} = \mathbf{0}$.

Our algorithm requires a number of parameters, which we would like to describe. The first of them is the dst_θ, defining the threshold for distance between a sample point and a Gaussian. This parameter helps to define if a new sample point should support an existing Gaussian or give birth to a new one. The algorithm can use either Euclidean or Mahalanobis distance for this purpose and a user may decide which of them to use. The second parameter div_θ defines the threshold for Kullback-Leibler divergence between two Gaussians. This parameter is used for defining if any two components in the mixture during updating its coefficients became too similar to be merged together into one. Unlike expectation maximization or K-Mean algorithms, our algorithm does not need the predefined number of mixture components to be given (or predefined number of segments), but we make use of the parameter G_{max}, limiting the maximum number of Gaussians in the mixture model.

The last two parameters are numbers $\hat{N}_{min}$, $\check{N}_{min}$, such that $\hat{N}_{min} \geq \check{N}_{min}$, which describe the minimal number of sample points (updates) needed for sequential estimation of a reliable Gaussian function. Though the parameter $\check{N}_{min}$ describes the lower boundary of samples, below which a Gaussian is considered to be a "noise"; Gaussians which were estimated with help of number of sample points lying between $\hat{N}_{min}$ and $\check{N}_{min}$ are to be merged with the most similar Gaussian, estimated on the large enough number of sample points.

Let us now consider Algorithm 1 where for each class $l \in \mathbb{L}$ one Gaussian mixture is trained more closely. We consider each training sample point as an evidence for parameters μ_{lk} and $\mathbf{\Sigma}_{lk}$ of one of the Gaussians in Equation 2.35. The sample points are processed sequentially in the order which they are collected. For each new sample point we check whether it belongs to an existing mixture component by evaluating the distances dst_k between the new sample point $\mathbf{f}$ and existing components $\mathcal{N}_k$, $k \in [1;G]$. If the smallest distance dst_{min} is shorter than dst_θ, the sample point is assigned to the component $\mathcal{N}_{k_{min}}$ corresponding to dst_{min}, and the parameters $\boldsymbol{\mu}_{k_{min}}$ and $\mathbf{\Sigma}_{k_{min}}$ of that component are updated. If the training sample point does not fit to any existing component (which is, of course, the case for the first sample point to be processed), we generate a new Gaussian and initialize its center $\boldsymbol{\mu}$ by that sample point (refer to Algorithm 5). However, this is only done if the number of components in a Gaussian mixture is lower than the limit G_{max}, otherwise we update the nearest component of that

Algorithm 1: Sequential GMM training | Part I

Data: distance threshold $dst_\theta = \{Euclidian\_dst_\theta$ OR $Mahalanobis\_dst_\theta\}$; divergence threshold div_θ; maximal number of Gaussians G_{max}; minimal number of sample points $\hat{N}_{min}$; train sample points $\mathbf{f}$ with given groundtruth label l

Result: $GaussianMixture$

```
while sample points do
    (f, l) ← GetNextPoint();
    if GaussianMixture_l.numGaussians = 0 then
        N ← new Gaussian();
        N.AddPoint(f);
        GaussMixture_l.Append(N);
    else
        forall N_k ∈ GaussianMixture_l do
            dst_k = distance(N_k, f);
        (dst_min, k_min) ← MIN_k(dst);
        if (dst_min > dst_θ) AND (GaussianMixture_l.numGaussians < G_max) then
            N ← new Gaussian();
            N.AddPoint(f);
            GaussMixture_l.Append(N);
        else
            N_{k_min}.AddPoint(f);
            if (div_θ) AND (N_{k_min}.numPoints ≥ N̂_min) then
                forall N_m ∈ GaussianMixture \ N_{k_min} do
                    div_m = divergence(N_m, N_{k_min});
                (div_min, m_min) ← MIN_m(div);
                if div_min < div_θ then
                    N_{m_min}.MergeWith(N_{k_min});
                    GaussianMixture_l.Erase(N_{k_min});
```

mixture.

For evaluating distances dst_k the algorithm uses the Euclidean distance $\mathbb{E}(\mathbf{f}, \boldsymbol{\mu})$ by default. But it also allows to specify weather the Mahalanobis distance $\Delta(\mathbf{f}, \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}))$ should be used instead. This is done by defining the parameter $Mahalanobis\_dst_\theta$ to be non-zero (*cf.* Algorithm 6). The Euclidean distance is defined between two points, whereas the Mahalanobis distance – between a point and a Gaussian function (*cf.* Equation 2.23). In our case, we calculate the Euclidean distance between a sample point $\mathbf{f}$ and the center of a mixture component $\boldsymbol{\mu}$ and for calculating the Mahalanobis distance, additionally the covariance $\boldsymbol{\Sigma}$ is needed.

In case when the Mahalanobis distance is in use, we take care for a mixture component,

to which we want to estimate the distance, to be reliable, *i.e.* weather it was estimated on a set of sample points large enough for estimating reliable covariance $\mathbf{\Sigma}$. If this component was estimated with help of less than $\hat{N}_{min}$ sample points, the Euclidean distance takes over. This is done, because parameter $\mathbf{\Sigma}$ needs much more sample points to be estimated than parameter $\boldsymbol{\mu}$. In order to make the Euclidean distance comparable with the Mahalanobis distance, we perform scaling of the Euclidean distance by the factor $Mahalanobis\_dst_\theta / Euclidean\_dst_\theta$.

Each update of a Gaussian affects its position and form trough parameters $\boldsymbol{\mu}_{lk}$ and $\mathbf{\Sigma}_{lk}$; so consequently we may check if it is possible to merge the updated Gaussian with any of the others, this time by comparing the Kulback-Leibler divergence $\mathbb{D}_{KL}(\mathcal{N}||\mathcal{N})$ from Equation 2.24 of the mixture components to the threshold div_θ in Algorithm 7. This is done to avoid having too many components in a mixture and correct possible mistakes of the very first stage of training. This step is optional and performed in case when parameter div_θ is set.

The Kullback-Leibler divergence is defined for two Gaussians. So, we check weather they were estimated on large enough set of sample points, in the same way as described above.

Algorithm 2: Sequential GMM training | Part II

Data: $GaussianMixture$; minimal number of sample points $\{\hat{N}_{min}, \check{N}_{min} : \hat{N}_{min} > \check{N}_{min}\}$
Result: $GaussianMixture$; minimal mixture coefficient ω_{min}

```
1  ω_min ← ∞;
2  forall l ∈ 𝕃 do
3      forall 𝒩_k ∈ GaussianMixture_l do
4          if 𝒩_k.numPoints < N̂_min then
5              if 𝒩_k.numPoints ≥ Ň_min then
6                  forall 𝒩_m ∈ GaussianMixture_l \ 𝒩_k do
7                      div_m = divergence(𝒩_m, 𝒩_k);
8                  m_min ← MIN_m(div);
9                  𝒩_{m_min}.MergeWith(𝒩_k);
10             GaussianMixture_l.Erase(𝒩_k);
11     forall 𝒩_k ∈ GaussianMixture_l do
12         if 𝒩_k.ω → ∞ then
13             GaussianMixture_l.Erase(𝒩_k);
14         ω_min ← MIN(ω_min, 𝒩_k.ω);
```

After all the training sample points have been process, we perform the post-training procedure (Algorithm 2), which aims to reduce the number of outliers, *i.e.* those mixture components, which were estimated on too few training sample points, or became unbounded.

The second aim of the post-training procedure is to find the smallest mixture coefficient ω_{min} among all Gaussians and all mixtures. This coefficient will be needed for the classification in order to normalize resulting potential values.

In order to reduce number of outliers, we compare the number of sample points, used for estimating each Gaussian in a mixture with thresholds $\hat{N}_{min}$ and $\check{N}_{min}$. If the mixture component was estimated with too small number of sample points, *i.e.* which is less than threshold $\check{N}_{min}$, then this component will be discarded; otherwise, if the number of used sample points greater or equal to $\check{N}_{min}$, but still less than $\hat{N}_{min}$, such mixture component will be merged with another, the most similar in terms of Kullback-Leibler divergence component. All components in a Gaussian mixture, having the unbounded mixture coefficient ω are also to be discarded.

Algorithm 3: Sequential GMM prediction

Data: $GaussianMixture$; test sample point $\mathbf{f}$
Result: Potential vector $\varphi(x = l \,|\, \mathbf{f}(\mathbf{y})) \equiv \varphi_l,\ \forall l \in \mathbb{L}$

```
forall l ∈ 𝕃 do
    numAllPoints ← 0;
    forall 𝒩_k ∈ GaussianMixture_l do
        numAllPoints ← numAllPoints + 𝒩_k.numPoints;
    if numAllPoints ≠ 0 then
        forall 𝒩_k ∈ GaussianMixture_l do
            ω ← 𝒩_k.numPoints / numAllPoints;
            φ_l ← φ_l + ω · 𝒩_k(x)/ω_min;
    else
        φ_l ← 0;
```

In the classification phase, when our Gaussian mixture model is trained, our task is to estimate potential of a sample point to belong to one of predefined classes $l \in \mathbb{L}$. Algorithm 3 performs this by utilizing the base Equation 2.35. Please note, that we calculate mixture coefficients ω_{lk} based on the number of sample points, used for estimating each component: $\omega_{lk} = \frac{numPoints_{lk}}{\sum_{j=1}^{G} numPoints_{lj}}$; thus a mixture coefficient ω_{lk} for the k^{th} component of the mixture is given by the average responsibility (also knows as posterior probability) which that component takes for explaining the data points. We also normalize all the potentials by factor $1/\omega_{min}$, calculated in Algorithm 2. In practice, mixture coefficients may be very small, and the normalization is performed in order to keep the potentials $\varphi(x = l \,|\, \mathbf{f}(\mathbf{y}) >> 0$.

Our method is fast because no iterations are required, and it does not require much memory due to its sequential nature. Moreover, we do not need to define the strict number of Gaussians in the GMM, but this number is adjusted to the training data. Our algorithm

does not guarantee the unique maximum likelihood solution, neither does expectation maximization algorithm. The solution might also differ under different order of training sample points.

The proposed algorithm is evaluated in Section 2.7.2.1, where we compare it with the expectation maximization training and report the timings, memory consumption as well as the achieved classification accuracy.

2.3.2 Discriminative Approaches

Another approaches to solve the decision problem are discriminative approaches. The core idea, underlying these approaches is to solve first the inference problem of determining the posterior class probabilities $p(x = l \,|\, \mathbf{f}(\mathbf{y}))$, and then subsequently use decision theory to assign each new $\mathbf{f}(\mathbf{y})$ to one of the classes. Approaches that model the posterior probabilities directly are called *discriminative models*.

The most of the discriminative models determine the posterior class probabilities and use decision theory implicitly, producing as an output the class label l. This prevents us from smooth incorporation of the discriminative models into the Conditional Random Fields framework in the same way as we incorporated generative approaches. In order to achieve the potentials $p(x = l \,|\, \mathbf{f}(\mathbf{y}))$ using a discriminative model we have to develop individual strategies, which will be based on the specific characteristics of that model. In this section we will describe some of these strategies.

2.3.2.1 K-Nearest Neighbors

The *K-nearest neighbors* classifier (KNN, [Alt92]) is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. Thus, the KNN approach is among the simplest of all discriminative approaches, but this classifier is still especially effective for low-dimensional feature spaces.

The input for the KNN algorithm consists of the K closest training samples in the feature space and the output is a class label l. An observation (or testing sample) $\mathbf{f}(\mathbf{y})$ is classified by a majority vote of its neighbors, with the observation being labeled by the class most common among its K nearest neighbors. In case of $K = 1$ the class of that single nearest neighbor is simply assigned to the observation $\mathbf{f}(\mathbf{y})$.

In order to estimate the potentials we consider the class of every neighbor as a vote for the most likely class of the observation. If the number of neighbors, having class l is K_l we can define the probability of the association potentials as: (see Figure 2.7)

$$p(x = l \,|\, \mathbf{f}(\mathbf{y})) = \frac{K_l}{K}. \tag{2.38}$$

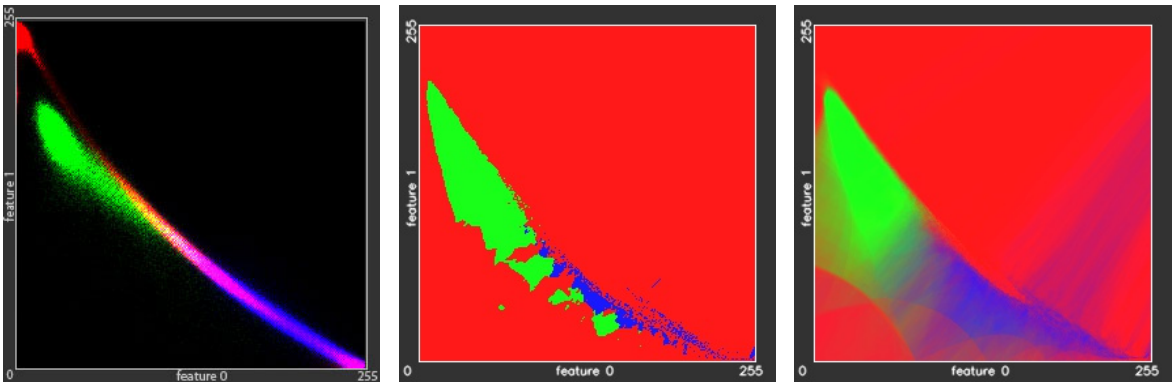


Fig. 2.7: Conversion of the KNN decision output to the potentials. **Left:** The original distributions of 160'000 samples from the *Green Fields* dataset; **Center:** Resulting KNN decision map; **Right:** Achieved potential map.

It can be useful to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $1/r$, where r is the distance to the neighbor. For our weighting scheme we modify this idea as follows: let r will be the Euclidean distance from the test sample to the nearest training sample in feature space and r_i – Euclidean distance to every found neighbor. Then we can rewrite Equation 2.38 with weighting coefficient:

$$p(x = l \,|\, \mathbf{f}(\mathbf{y})) = \frac{1}{K} \sum_i \frac{1_l}{(1 + r_i - r)^2}, \tag{2.39}$$

where 1_l means 1 if the class of the training sample is l and 0 otherwise.

The search algorithm aims usually to find exactly K nearest neighbors. However it may happen, that distant neighbors do not affect probability from Equation 2.39 much. For example, the nearest neighbor with $r_i = r$ contributes value of $1/K$ to the probability. And a neighbor, twice as distant from the testing sample ($r_i = 2r$) will contribute only $1/K(1+r)^2$. For the optimization purpose we stop the search once the distance from the test sample to the next nearest neighbor exceeds $2r$. Thus, only $K' \leq K$ neighbors in area enclosed between two spheroids of radii r and $2r$ are considered (see Figure 2.8) and weighted according to the Equation 2.38: $p(x = l \,|\, \mathbf{f}(\mathbf{y})) = K_l/K'$.

The neighbors are taken from a set of objects for which the class is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. A peculiarity of the KNN algorithm is that it is sensitive to the local structure of the data.

2.3.2.2 Support Vector Machine

Support vector machines (SVMs, [CS99]) belong to the class of binary classifiers that represent the training samples $\mathbf{f}(\mathbf{y})$ as multi-dimensional points in feature space, mapped so that the

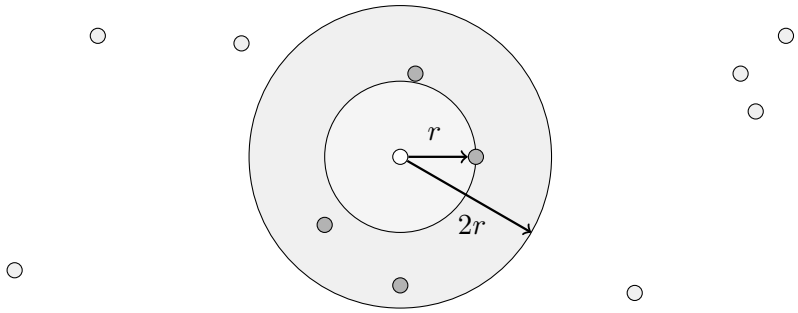


Fig. 2.8: Illustration of the nearest neighbors screening: if the distance to the nearest neighbor is r, we take into consideration only those neighbors that lie closer then $2r$ distance.

samples of two different classes are separated by a clear gap that is as wide as possible. The testing samples are then mapped into that same feature space and classified based on which side of the gap they fall. SVMs can efficiently perform a non-linear classification by implicitly mapping their inputs into high-dimensional feature spaces.

In order to use SVMs for the multi-class classification we use the *one-against-the-rest* technique [HL02], where we unite $|\mathbb{L}|$ SVN classifiers $\eta_l(\mathbf{f}(\mathbf{y})), l \in \mathbb{L}$, where η_l are real-valued score functions. The resulting class is then chosen by the highest score. Hence, the SVMs do not provide posterior probabilities $p(x\,|\,\mathbf{f}(\mathbf{y}))$ directly. In order to represent SVN classifier decision in form of the posterior probability, a sigmoid function, proposed by John Platt [Pla99], is used:

$$p(x = l\,|\,\mathbf{f}(\mathbf{y})) = \frac{1}{1+\exp(A_l\eta_l(\mathbf{f}(\mathbf{y})) + B_l)}, \tag{2.40}$$

where A_l and B_l are scalar parameters that are learned by the algorithm given in [LLW07]. This approach has got the the name *Platt Scaling* (see Figure 2.9).

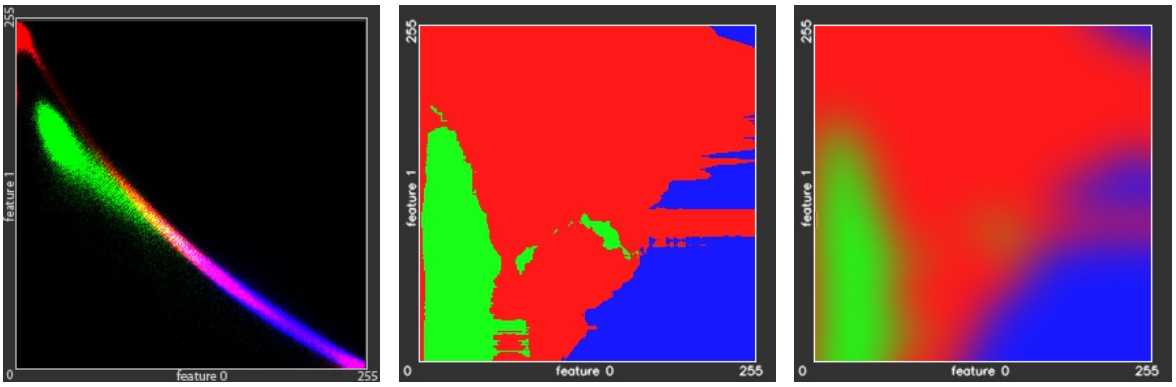


Fig. 2.9: Conversion of the SVM decision output to the potentials. **Left:** The original distributions of 160'000 samples from the *Green Fields* dataset; **Center:** Resulting SVM decision map; **Right:** Achieved with the Platts scaling potential map.

2.3.2.3 Random Forests

Random forest classifier (RF, [Bre01]) is an ensemble learning method, that operate by constructing a multitude of *decision trees* [Qui86] at training time and outputting the class l that is the mode of the classes of the individual trees. Random forests correct for decision trees' habit of overfitting to their training set.

For the RF discriminative approach we achieve potentials by accounting the votes for the most likely class cast by every decision tree (see Figure 2.10). If the RF consists of N_T decision trees and the number of votes cast for a class l is N_l, the probability underlying our definition of the association potentials is

$$p(x = l \,|\, \mathbf{f}(\mathbf{y})) = N_l / N_T. \tag{2.41}$$

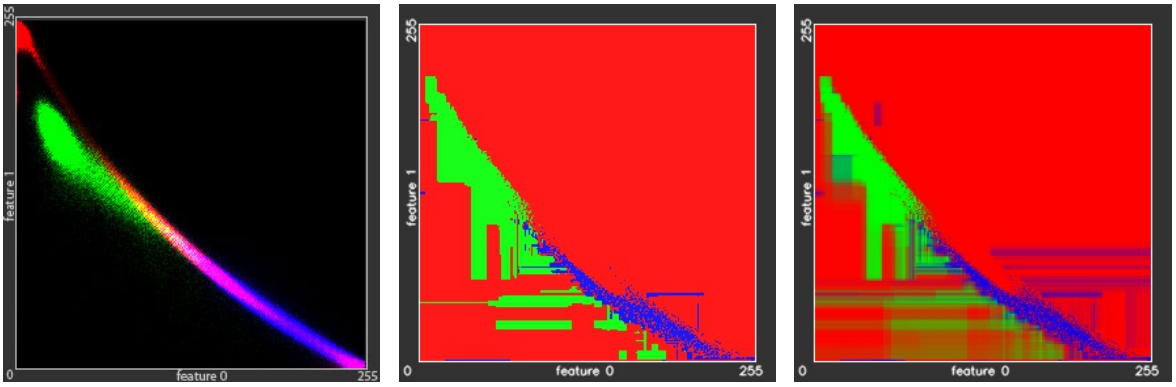


Fig. 2.10: Conversion of the RF decision output to the potentials. **Left:** The original distributions of 160'000 samples from the *Green Fields* dataset; **Center:** Resulting RF decision map; **Right:** Achieved potential map.

Feature Importance Criteria. The random forest approach allows us to estimate the impact of each feature $f_j \in \mathbf{f}(\mathbf{y})$ on classification and consequently provides us with a feature importance criteria. This is done by estimating how often the decisions in the nodes of random trees are based on certain features. These numbers are accumulated for each feature, then normalized and united in a single *feature importance vector*.

2.3.2.4 Artificial Neural Network

Artificial neural networks classifier (ANNs, [RB93]) is inspired by the biological neural networks that constitute mammal brains. The original goal of such approaches was to solve problems in the same way that a human brain would. Hence, ANN classifiers learn (or

progressively improve performance) by considering training samples, generally without class-specific features. In image recognition, for example, they might learn to identify image patches containing cars by analyzing sample image patches manually labeled as true positive or true negative and using the analytic results to identify cars in other images.

An ANN is based on a collection of connected units (neurons). Each connection (synapse) between neurons can transmit a signal to another neuron. The receiving neuron can process the signals and then signal downstream neurons connected to it. Neurons and synapses have a weight represented by a real number between 0 and 1 that varies as learning proceeds, which can increase or decrease the strength of the signal that it sends downstream. Further, they have a threshold such that only if the aggregate signal is above that level is the downstream signal sent. Typically, neurons are organized in layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first (input), to the last (output) layer, possibly after traversing the layers multiple times.

We initialize the number of neurons in the input layer to be equal to the number of features $|\mathbf{f}(\mathbf{y})|$ and the number of neurons in the output layer to be equal to the number of labels $|\mathbb{L}| \equiv k$. In order to represent the ANN decision on the correct label into the form of the posterior probability, we take the k scaled signals leaving the neurons from the output layer and use these signals' strength directly for filling the needed potentials $p(x_i = l \,|\, \mathbf{f}_i(\mathbf{y}))$ (see Figure 2.11).

Fig. 2.11: Conversion of the ANN decision output to the potentials. **Left:** The original distributions of 160'000 samples from the *Green Fields* dataset; **Center:** Resulting ANN decision map; **Right:** Achieved potential map.

2.4 Interaction Potentials

The interaction potential functions $\psi_{ij}(x_i, x_j; \mathbf{y})$ in Equation 2.5 are related to the probability of the random variables x_i, x_j taking a pair of labels l, l' given the data $\mathbf{y}$ [KH06]. Since

many models for interaction potential functions include control parameters, we will now refer to Equation 2.7 and model $\langle \psi_{ij}(x_i, x_j; \mathbf{y}), \boldsymbol{\theta}_\psi \rangle$ in general case directly by:

$$\langle \psi_{ij}(x_i, x_j; \mathbf{y}), \boldsymbol{\theta}_\psi \rangle \propto p(x_i = l, x_j = l' \,|\, \mathbf{f}_i(\mathbf{y}), \mathbf{f}_j(\mathbf{y}), \boldsymbol{\theta}) \tag{2.42}$$

where the image data are represented by site-wise feature vectors $\mathbf{f}_i(\mathbf{y})$, $\mathbf{f}_j(\mathbf{y})$ that may depend on all the observed data $\mathbf{y}$ and parameters $\boldsymbol{\theta} \in \boldsymbol{\theta}_\psi$.

There are exist many strategies of constructing the interaction potential functions, and for convenience we can split them into three groups of approaches, namely data independent, contrast sensitive and data dependent approaches (see Table 2.1). We start this section with providing brief introduction to the data independent and contrast sensitive approaches, and continue with our data dependent models.

Approach	potential function	train data	test data
data independent	$\psi_{ij}(x_i, x_j)$	–	–
contrast sensitive	$\psi_{ij}(x_i, x_j; d_{ij})$	–	✓
data dependent	$\psi_{ij}(x_i, x_j; \mathbf{y})$	✓	✓

Tab. 2.1: The test- / train-data dependencies of the interaction potential function modeling approaches. The term d_{ij} in the contrast sensitve potential functions corresponds to the difference metric between observations y_i and y_j and may be given *e.g.* by Equation 2.44.

2.4.1 Data Independent and Contrast Sensitive Approaches

Data Independent. The most naïve approach to model pairwise potentials in multi-class CRF classifiers is the Potts model [Wer+11]. It guarantees smooth label maps and ignores both training and test data, *i.e.* $\langle \psi_{ij}(x_i, x_j; \mathbf{y}), \boldsymbol{\theta}_\psi \rangle \propto p(x_i, x_j \,|\, \boldsymbol{\theta})$.

$$p(x_i = l, x_j = l' \,|\, \boldsymbol{\theta}) = \begin{cases} \theta_l & \text{if } l = l' \\ 1 & \text{otherwise} \end{cases} \tag{2.43}$$

In Equation 2.43, the parameters $\theta_l \in \boldsymbol{\theta}$, $l \in \mathbb{L}$ modulate the degree to which the interaction potential favors identical classes at neighboring sites. They could be considered as smoothness weight coefficients for all classes and usually are larger then 1.

Contrast Sensitive. The contrast sensitive Potts model [BJ01] could be considered as an "upgrade" version of the naïve Potts model 2.43. It is still independent from the training data, but takes into account the difference (contrast) between observations on adjacent sites. This difference may be expressed in terms of Euclidean distance between corresponding feature

vectors [SM00]:

$$d_{ij} = \mathbb{E}(\mathbf{f}_i(\mathbf{y}), \mathbf{f}_j(\mathbf{y})) \tag{2.44}$$

thus Equation 2.42 could be simplified to: $\langle \psi_{ij}(x_i, x_j, \mathbf{y}), \boldsymbol{\theta}_\psi \rangle \propto p(x_i, x_j \,|\, d_{ij}, \boldsymbol{\theta})$.

Having defined the difference measure d_{ij}, that is supposed to be high on segments' borders and low at homogenous regions of classifying images, one chooses a regularization function $\mathcal{P}(d_{ij})$, which ought to penalize smoothness term on regions, corresponding to abrupt changes of observed data:

$$p(x_i = l, x_j = l' \,|\, d_{ij}, \boldsymbol{\theta}) = \begin{cases} \theta_l \cdot \mathcal{P}(d_{ij}) & \text{if } l = l' \\ 1 & \text{otherwise} \end{cases} \tag{2.45}$$

The function $\mathcal{P}(d_{ij}) : \mathbb{R} \to \mathbb{R}$ may be any arbitrary penalization function, *e.g.* well-known Charbonnier [Cha+94] or Perrona-Malik [PM90] regularizers and also may depend on additional control parameters. In this thesis we use an exponential regularizrer in the form:

$$\mathcal{P}(d_{ij}) = e^{-\theta_\mathcal{P} \cdot d_{ij}^2} \tag{2.46}$$

here parameter $\theta_\mathcal{P}$ modulates the contrast-sensitive term.

The Potts model without the data dependent term in Equation 2.43 would favor identical class labels at neighboring image sites and, thus, result in a smoothed label image. In the contrast sensitive Potts model in Equation 2.45 this will still be the case if the feature vectors $\mathbf{f}_i(\mathbf{y})$, $\mathbf{f}_j(\mathbf{y})$ are identical, but large differences between the features will reduce the impact of this smoothness assumption and make a class change between neighboring image sites more likely. This results in smooth label maps covering homogenous image regions, while preserving edges, where the classifying objects' borders are more probable.

2.4.2 Data Dependent Approaches

Data dependent interaction potentials incorporate a trained function, whose parameters are estimated from the training data. We start with estimation of the prior probability of labels co-occurrence at adjacent sites, which results in a simple and fast approach, producing reliable results. After that we continue with concatenating a pair of classifiers, similar to those, described in Section 2.3 and designing the interaction model, based on outputs of these two classifiers.

2.4.2.1 Histogram Matrix

For our first interaction model we introduce a training data dependent term, which considers the relative frequency of class transitions [Pra+06]. In classification this term represents the prior probability of two adjacent random variables x_i and x_j taking a pair of labels (l,l'). For this purpose we generate a 2D histogram $H(l,l')$ of the co-occurrence of labels at adjacent image sites in the training data, *i.e.* $H(l,l')$ – is the number of occurrences of the classes (l,l') at adjacent sites. After the training phase, in order to avoid a bias for classes covering a large area in the training data, we normalize the histogram $H(l,l')$, which results in a matrix $h(l,l')$. We distinguish two types of normalization: symmetric and asymmetric, which have own pros and cons. Let us describe them in more detail.

Symmetric Approach. Symmetric normalization approach guaranties that resulting matrix h is symmetric, *i.e.* $h(l,l') = h(l',l)$ and that the diagonal elements are equal to one: $h(l,l) = 1$, $\forall l,l' \in \mathbb{L}$. In case of $H(l,l) > H(l,l')$, $\forall l,l' \in \mathbb{L}$, the largest value in matrix h will be equal to one; and in case of $H(l,l')^2 \geq H(l,l) \cdot H(l',l')$, the largest value in matrix h may exceed one. The latter case is possible to model for academic purpose, but it is almost improbable in real life applications.

Asymmetric Approach. Asymmetric normalization approach takes into account the difference in number of occurrences of classes in the training data (*i.e.* prior probabilities $p(x = l)$ and $p(x = l')$). It guaranties that the largest value in matrix h does not exceed one. The case of the diagonal values of the matrix are not the largest is probable (*i.e.* $h(l,l) \leq h(l,l') = 1$). This may lead to erosion of areas with small class occurrence rate. Please note, that the results of the asymmetric approach could be approximated from the results of symmetric approach, by multiplying the columns of the matrix h with the corresponding to the base columns class, the prior class probability.

Having estimated the training data dependent matrix $h(l,l')$, we combine it with the contrast sensitive Potts model from the Equation 2.45 to achieve our data dependent definition of interaction potential $\langle \psi_{ij}(x_i,x_j;\mathbf{y}),\boldsymbol{\theta}_\psi \rangle \propto p(x_i,x_j \,|\, d_{ij},\boldsymbol{\theta}) \cdot h(l,l') = p(x_i,x_j \,|\, \mathbf{y},\boldsymbol{\theta})$:

$$p(x_i = l, x_j = l' \,|\, \mathbf{y},\boldsymbol{\theta}) = \begin{cases} \theta_l \cdot \mathcal{P}(d_{ij}) \cdot h(l,l') & \text{if } l = l' \\ h(l,l') & \text{otherwise} \end{cases} \tag{2.47}$$

This model differs from the contrast-sensitive Potts model in Equation 2.45 by the use of the normalized histograms $h(l,l')$, which are estimated from the training data. As a consequence, class transitions become more likely, depending on the frequency with which

they occur in the training data.

2.4.2.2 Concatenated Edge Potentials

Our second interaction model is based on an idea of uniting all possible combinations of class transitions together and classify these combinations in a similar was as it was done for unary potentials. For this purpose we introduce a new concatenated set of class labels

$$\mathbb{C} = \mathbb{L} \times \mathbb{L}, \tag{2.48}$$

which encodes all possible combinations of two labels $l, l' \in \mathbb{L}$ with one label $l'' \in \mathbb{C}$. If we also concatenate the corresponding feature vectors $\mathbf{f}_i(\mathbf{y})$, $\mathbf{f}_j(\mathbf{y})$ together, into the vector $\mathbf{f}_{ij}(\mathbf{y}) = \{f_{i1}, f_{i2}, \ldots, f_{im}, f_{j1}, f_{j2}, \ldots, f_{jm}\}$ we can write the potential function in form:

$$p(x_i = l, x_j = l' \,|\, \mathbf{f}_i(\mathbf{y}), \mathbf{f}_j(\mathbf{y}), \boldsymbol{\theta}) \equiv p(\{x_i; x_j\} = l'' \,|\, \mathbf{f}_{ij}(\mathbf{y}), \boldsymbol{\theta}), \tag{2.49}$$

which can be modeled in the same way as unary potentials, from Equation 2.13.

Please note, that our model data dependent interaction model 2.49 allows asymmetricity: $p(x_i, x_j \,|\, \mathbf{f}_i(\mathbf{y}), \mathbf{f}_j(\mathbf{y}), \boldsymbol{\theta}) \neq p(x_j, x_i \,|\, \mathbf{f}_j(\mathbf{y}), \mathbf{f}_i(\mathbf{y}), \boldsymbol{\theta})$.

2.5 Inference and Decoding

There are efficient and exact solutions to inference problem in DAGs and tree-structured graphical models. However, in CRFs which generally allow loops in underlying graphs, exact inference is computationally intractable [Vis+06; KH06] and thus approximate methods should be used. Among such approximations one can select *variational* methods [Jor+99], belonging to the class of deterministic approaches and *sampling* (also called *Monte Carlo*) methods [And+03], belonging to the class of stochastic approaches.

In the scope of this thesis, we will concentrate on the class of iterative *message-passing* algorithms [Pea88]. They were designed to provide efficient and exact solutions to inference problem in tree-structured graphs. But when the message-passing rules are purely local, they may be also applied to the graphs with loops, even though there is no guarantee that they will yield good results. Due to the cycles, information can flow many times around the graph and for some models, the algorithm will converge, whereas for others it will not.

Among message-passing algorithms we will distinguish *sum-product* and *max-product* algorithms, which are closely related. The sum-product algorithm allows us to take a joint distribution and efficiently find marginals over the component variables. The max-product algorithm allows for finding a setting of the variables that has the largest probability and to

find the value of that probability. The max-sum algorithm can be viewed as an application of dynamic programming in the context of graphical models [Cor+01].

2.5.1 Sum-Product Message Passing Algorithm

The sum-product message passing algorithm is also known as *loopy belief propagation* (LBP, [FM98; YFW03]). This algorithm works by passing real valued functions called *messages* along with the graph edges. More precisely, if x_i and x_j are two variable nodes, the messages from x_i to x_j, (denoted by $\mu_{i\to j}(x_j) \in \mathbb{R}^k$) and from x_j to x_i ($\mu_{j\to i}(x_i) \in \mathbb{R}^k$), are real-valued functions – the set of values that can be taken by the random variable associated with node x. These messages say about which value the recipient node should have.

The belief $b_i(x_i) \in \mathbb{R}^k$ at a variable node x_i is proportional to the product of the local evidences ($\varphi_i(x_i)$) at the node i, and all the messages coming into node i:

$$b_i(x_i) = k\ \varphi_i(x_i) \prod_{(i,j)\in\mathcal{E}} \mu_{j\to i}(x_i), \tag{2.50}$$

where k is a normalization constant (the beliefs must sum to 1) and i and j are adjacent nodes.

A message from a variable node x_i to a node x_j is the product of the messages from all neighboring to x_i nodes, except the recipient. The messages are determined self-consistency by the message update rule:

$$\mu_{i\to j}(x_j) = \psi_{i,j}(x_i, x_j)\varphi_i(x_i) \prod_{\substack{(i,k)\in\mathcal{E}\\ k\neq j}} \mu_{k\to i}(x_i). \tag{2.51}$$

In some applications, the loopy belief propagation algorithm can give poor results, whereas in other applications it has proven to be very effective. In particular, state-of-the-art algorithms for decoding certain kinds of error-correcting codes are equivalent to loopy belief propagation [Mac06].

2.5.2 Max-Product Message Passing Algorithm

In Section 2.5.1 we described a simple approach to find latent variable values having high probability. The sum-product algorithm can obtain the marginals $p(x_i)$ for *every single variable*, and then, for each marginal in turn, find the value x_i^* that maximizes that marginal. However, this gives the set of values that are *individually* the most probable. In order to find the set of values that *jointly* have the largest probability we can use the max-product algorithm, wich is a generalization of the *Viterbi algorithm*. It allows us to find the vector $\tilde{\mathbf{x}}$,

that maximizes the joint distribution, so that

$$\tilde{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmax}}\, p(\mathbf{x}) \tag{2.52}$$

for which the corresponding value of the joint probability will be given by

$$p(\tilde{\mathbf{x}}) = \max_{\mathbf{x}} p(\mathbf{x}). \tag{2.53}$$

	$x=0$	$x=1$
$y=0$	0,3	0,4
$y=1$	0,3	0,0

Tab. 2.2: Example of a joint distribution over two binary variables for which the maximum of the joint distribution occurs for different variable values compared to the maxima of the two marginals.

In general, $\tilde{\mathbf{x}}$ is not the same as the set of x_i^* values, as we can easily show using a simple example. Consider the joint distribution $p(x,y)$ over two binary variables $x,y \in 0,1$ given in Table 2.2. The joint distribution is maximized by setting $x=1$ and $y=0$, corresponding the value $0,4$. However, the marginal for $p(x)$, obtained by summing over both values f y, is given by $p(x=0)=0,6$ and $p(x=1)=0,4$, and similarly the marginal for y is given by $p(y=0)=0,7$ and $p(y=1)=0,3$, and so the marginals are maximized by $x=0$ and $y=0$, which corresponds to value of $0,3$ for the joint distribution. In fact, it is not difficult to construct examples for which the set of individually most probable values has probability zero under the joint distribution.

We therefore seek an efficient algorithm for finding the value of $\mathbf{x}$ that maximizes the joint distribution $p(\mathbf{x})$ and that will allow us to obtain the value of the joint distributions at its maximum. To address the second of these problems, we will simply write out the max operator in terms of its components

$$\max_{x} p(\mathbf{x}) = \max_{x_1} \ldots \max_{x_M} p(\mathbf{x}) \tag{2.54}$$

where M is the total number of variables, and then substitute for $p(\mathbf{x})$ using its expansion. In deriving the sum-product algorithm, we made use of the distributive law for the max operator

$$\max(ab, ac) = a\ \max(b,c) \tag{2.55}$$

which holds if $a \geq 0$. This allows us to exchange products in Equation 2.51 with maximization.

Tree-Reweighted Message Passing Algorithm One generalization of the max-product message passing algorithm to higher order clique updates is the tree-reweighted max-product (TRW) algorithm [WJW05] and sequential, monotone update sequence of Kolmogorov [Kol06]. This method can be understood within the dual decomposition framework, in which the dual formulation of the optimization problem can be decomposed into smaller sub-problems that are tied together via Lagrange multipliers.

The TRW dual decomposition method operates by creating a collection of trees which span the full graph, and associating with each tree t a set of parameters τ^t such that the sum $\sum_t \tau^t = \tau$. Each tree is solved separately, providing an upper bound on the MAP, and the algorithm then minimizes this upper bound over the allocation of τ to each tree. Dual decomposition provides a monotone update by sequentially visiting each node and edge and "merging" its copies [Fou+11].

2.6 Parameter Estimation

The control parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_\varphi, \boldsymbol{\theta}_\psi\}$ from Equation 2.7 are learned separately from the association (φ_i) and interaction (ψ_{ij}) potentials and may be obtained using the cross validation technique [Sho+09] or the Powell search method [Pow64; Kra10]. Both methods make use of a real-valued *objective function* Ω of n arguments $\Omega(\boldsymbol{\theta}) : \mathbb{R}^n \to \mathbb{R}$, where $n = |\boldsymbol{\theta}|$ is the number of parameters $\boldsymbol{\theta}$. The problem of control parameters estimation is then turned into the optimization problem, of finding maximum of the objective function:

$$\tilde{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \Omega(\boldsymbol{\theta}). \tag{2.56}$$

The objective function is typically the mean squared prediction error, but within this work we define Ω as the normalized sum of the diagonal elements of the *confusion matrix* [Ste97], obtained by classifying the part of the training data that was not used for training the potentials. A confusion matrix is a specific table layout that allows visualization of the performance of an algorithm. Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa). The diagonal elements represent the rate of correctly classified instances, thus our definition of the objective function is equally sensitive to the large and small classes and is not biased by the amount of entities in a class.

2.6.1 Cross Validation Technique

In a typical K-fold *cross-validation* procedure the data set is randomly and evenly split into K parts (if possible). A candidate model is built based on $K-1$ parts of the data set, called a training set. Prediction accuracy of this candidate model is then evaluated on a test set containing the data in the hold-out part. By respectively using each of the K parts as the test set and repeating the model building and evaluation procedure, we choose the model with the largest value of the objective function $\Omega(\boldsymbol{\theta})$ as the 'optimal' model. Given n independent variables, there are in total of 2^n-1 possible models. In the K-fold cross-validation procedure, each model is in fact evaluated K times. Therefore, a single 'optimal' model is selected via $K(2^n-1)$ times of model evaluation.

The main advantage of the cross-validation technique for the parameter estimation is that it usually produces a reliable results even on small training datasets. The problem of small dataset becomes even more daunting if we recall that for parameter estimation we need to use a separate subset of the training data, different from the raining data, used for the potentials training. From another hand, the cross-validation technique demands an exponential to the number of variables n time for estimating parameters. That prevents us using this technique for problems with large number of parameters.

2.6.2 Powell Search Method

The *Powell search method* [Pow64] is an iterative optimization algorithm that does not require taking the derivative of the objective function and thus it is especially useful for the functions without an underlying mathematical definition. We initialize the algorithm with the initial approximation (search point) $\boldsymbol{\theta}^{(0)}$ and n initial search vectors $\{\boldsymbol{\Delta}_1^{(0)},\dots,\boldsymbol{\Delta}_n^{(0)}\}$, which are simply the normals aligned to each axis [Kos15].

The method maximizes the objective function by a bi-directional search along each search vector, in turn. The bi-directional line search along each search vector can be done by Golden-section search [AW66] or Brent's method [Atk89]. Let the maximum found during each bi-directional line search be:

$$\begin{aligned}
&\boldsymbol{\theta}^{(0)}+\alpha_1\boldsymbol{\Delta}_1^{(0)},\\
&\boldsymbol{\theta}^{(0)}+\textstyle\sum_{i=1}^{2}\alpha_i\boldsymbol{\Delta}_i^{(0)},\\
&\dots,\\
&\boldsymbol{\theta}^{(0)}+\textstyle\sum_{i=1}^{n}\alpha_i\boldsymbol{\Delta}_i^{(0)},
\end{aligned} \tag{2.57}$$

where α_i is the scalar determined during bi-directional search along $\boldsymbol{\Delta}_i$. Then the new

approximation $\boldsymbol{\theta}^{(1)}$ can then be expressed as a linear combination of the search vectors:

$$\boldsymbol{\theta}^{(1)} = \boldsymbol{\theta}^{(0)} + \sum_{i=1}^{n} \alpha_i \boldsymbol{\Delta}_i^{(0)}. \tag{2.58}$$

The new displacement vector $\sum_{i=1}^{n} \alpha_i \boldsymbol{\Delta}_i^{(0)}$ becomes a new search vector, and is added to the end of the search vector list. Meanwhile, the search vector which contributed most to the new direction, *i.e.* the one which was most successful ($\tilde{i} = \text{argmax}_i \, \alpha_i \| \boldsymbol{\Delta}_i^{(0)} \|$), is deleted from the search vector list. The new set of n search vectors is:

$$\begin{aligned}
\boldsymbol{\Delta}_1^{(1)} &= \boldsymbol{\Delta}_1^{(0)} \\
&\dots \\
\boldsymbol{\Delta}_{\tilde{i}-1}^{(1)} &= \boldsymbol{\Delta}_{\tilde{i}-1}^{(0)} \\
\boldsymbol{\Delta}_{\tilde{i}}^{(1)} &= \boldsymbol{\Delta}_{\tilde{i}+1}^{(0)} \\
&\dots \\
\boldsymbol{\Delta}_{n-1}^{(1)} &= \boldsymbol{\Delta}_n^{(0)} \\
\boldsymbol{\Delta}_n^{(1)} &= \textstyle\sum_{i=1}^{n} \alpha_i \boldsymbol{\Delta}_i^{(0)}
\end{aligned} \tag{2.59}$$

In such a way the algorithm iterates an arbitrary number of times until no significant improvement is made [Mat17].

The method is useful for calculating the local maximum of a continuous but complex functions for which estimation of the gradient is problematic or impossible. The basic algorithm is simple; the complexity is in the linear searches along the search vectors, which can be achieved via Brent's method.

2.7 Experiments

In this section we evaluate and compare the techniques, that were presented and described in this chapter. For the evaluation we use three datasets: synthetic *Green Field* dataset, represented via a set of sample points in 2-dimensional feature space and describing 3 classes; and two real-world datasets: *EMDS* and *Vaihingen*. For further information about these datasets, as well as for the total list of extracted features, please refer to the Appendix D. For numerical reasons, all features were scaled linearly into the range $[0; 255]$ and then quantized by 8 bit. Our CRF-classification is based on the *direct graphical models* C++ library [Kos15].

During the evaluation we used different models for the association and interaction potentials. For the association potentials we used the following abbreviations: *Bayes* - naïve Bayes model, described in Section 2.3.1.1; *emGMM* - Gaussian mixture model, estimated

with help of Expectation-Maximization algorithm and *seqGMM* - our sequential Gaussian mixture model, presented in Section 2.3.1.3; *KNN* - K-nearest neighbors model from Section 2.3.2.1; *SVM* - support vector machines model from Section 2.3.2.2; *RF* - random forests model, described in Section 2.3.2.3, using $N_T = 100$ trees (Equation 2.41) of maximum depth 15; *ANN* - artificial neural networks model from Section 2.3.2.4.

For the interaction potentials we used the following abbreviations: *Potts* - the naïve Potts model, given by Equation 2.43 and using $\theta_l = 23/5$, $\forall l \in \mathbb{L}$; *PottsCS* - Potts model, enhanced with the contrast-sensitive term defined in Equation 2.45, with the same θ_l and $\theta_{\mathcal{P}} = 1$; *hMat* - our data-dependent interaction model, described in Equation 2.47 with the same parameters θ and *Concat* - our advanced data-dependent model, described in Section 2.4.2.2 and based on concatenating two unary Bayesian models. In order to assess also the CRF technique we considered additionally *NoEdge* experiment – where the interaction potentials were not implied, *i.e.* interaction potentials were set to $\psi_{ij}(x_i, x_j; \mathbf{y}) \equiv 1$.

Finally, the classification results are compared with the reference. For evaluation of pixel-level segmentation we report the recall and the precision of pixel classification (labeling) as well as the overall accuracy. Additionally, we use *Average Precision* (AP) [AY06] as an evaluation measure of pixel-level classification. An AP is calculated by considering pixel classification results in each image separately. For each EM class, pixels in the image are sorted based on the potentials for being that class. Then, an AP is computed as the average of precisions each of which is computed at the position of a pixel belonging to the class. A larger AP means a better result where pixels for the class are ranked at higher positions. Such APs are computed for test images containing EMs for the class, and averaged to indicate an abstracted pixel-level classification performance. Finally, we take the 'Mean of such averaged APs' (Mean AP) over all the 20 classes to obtain an overall performance.

2.7.1 Impact of The Features on Classification

2.7.1.1 DCNN Features

We start with demonstrating the effectiveness of DCNN features presented in Section 2.2.2 on the task of EM classification. We use 50% of the EMDS dataset images to train DeepLab-VGG-16 for pixel-level feature extraction and unary potentials φ_i. DeepLab-VGG-16 is trained by following the network structure and hyper parameters provided as DeepLab-LargeFOV [7], except that the mini-batch size is changed from 30 to 20 due to the RAM size of our GPU. With respect to RF training on 1024-dimensional pixel-level features, most regions in EM images are backgrounds. This causes the imbalanced problem that pixel-level

[7] `http://liangchiehchen.com/projects/DeepLab-LargeFOV.html`

features for backgrounds (majority class) significantly outnumbers features for 20 EM classes (minority classes) [HG09]. As a result, a meaningless RF that classifies almost all pixels into the background class is favored, because its classification accuracy on training images is high. To overcome this, for each of 21 classes (20 EM classes and the *background* class), an RF is trained by randomly sampling the same number of pixels. Here, this number is chosen as the minimum number of pixels among 21 classes (specifically, 19063 pixels for *Epistilis* (ω_6)).

We compare DCNN features to the following two sets of features:

SIFT: A *Scale-Invariant Feature Transform* (SIFT) feature is one of the most popular local feature, and represents the shape in a local region, reasonably irrespective of changes in illumination, rotation, scaling and viewpoint [Low99]. We densely extract SIFT features by locating interesting points at all pixels. As a result, $\mathbf{f}_i(\mathbf{y})$ for each pixel is characterized by a 128-dimensional SIFT descriptor.

Simple: Here we have gathered 16 common features, which are usually used in image classification: the intensity, calculated as the average of the red, blue and green channels; the saturation component in HSL color space (please see Appendix D.3 for extensive description of the simple features).

We carry out two sets of experiments for every feature: $RF{:}NoEdge$ and $RF{:}Potts$. This comparison should show the impact of different features on the classification with RF and CRF.

	SIFT		Simple		DCNN	
CRF	local	+global	local	+global	local	+global
$RF{:}NoEdge$	2.21 %	10.77 %	18.04 %	35.03 %	54.78 %	62.19 %
$RF{:}Potts$	3.80 %	10.99 %	19.60 %	31.40 %	53.69 %	61.77 %

Tab. 2.3: Mean APs of the results for 3 sets of local features: SIFT, Simple, DCNN; and 2 types of CRFs: local and local-global. The impact of the local features and addition the global features is compared for two classification models: per-pixel RF ($RF{:}NoEdge$) and CRF with Potts pairwise potentials ($RF{:}Potts$).

The Mean AP for local CRF over 20 EM classes in these two experiments are shown in Table 2.3. SIFT features show very poor results, and in spite of they might be useful for solving correspondence problems, they lead to a low classification rate, when used to support CRFs. Finally, we can observe that DCNN features deliver us more than 'twice-as-better' results than Simple features: 54.78%. This validates that DCNN features work more robustly than SIFT and Simple features for classification [Kos+18].

2.7.1.2 Global Features

In this section we demonstrate the effectiveness of the additional global features. As in the previous experiment, we compare three sets of features: SIFT, Simple and DCNN features. Again we carry out two different experiments for every feature set: $RF{:}NoEdge$ and $RF{:}Potts$ (see Section 2.7.1.1 for more details). For the global node, we also train a RF using global features in the 200 training images. Finally, a local-global CRF is implemented in the framework of *Direct Graphical Models* C++ library [Kos15].

The Mean AP over 20 EM classes in the two experiments are shown in Table 2.3. As we can see from Table 2.3, the combination of Simple features with the global features increases the classification rate for them for all experiments near by factor of two. This is done because the additional global node connected to all the local nodes provides a long-range interaction between the local nodes and thus reduces the number of different EM classes present in the image. In other words, the global node supports the decision on the correct EM class and helps to reduce the segments, labeled as wrong EM classes. We can observe it in Figure 2.12, where the incorporation of the global features leads to significant clearance of the resulting label maps from the allogenic segments. We consider this as the consequence of the global smoothness effect infused by the addition of the global node. This validates that the combination of pixel-level features with the global features works more robustly than pixel-level features alone [Kos+18].

The impact of the global features on the classification with the local DCNN features is not so huge: in average they increase already high MAP values by additional 8%. We explain that by the fact, that the DCNN features in comparison to the Simple features are very powerful for EM classification itself, and thus the introduction of the additional global constraints make less effect as for other more weak features.

2.7.1.3 Confidence Features

The proper labeling of classes which are represented by small or even tiny regions in images is a common problem for CRFs because the incorporation of the interaction potentials usually leads to over-smoothing in label maps. This experiment addresses exactly this problem and studies the impact of underlying features on its solution. In the Vaihingen dataset of aerial images, class *car* suffers the most from the over-smoothing effect, because an image region, representing car object is about $0,09$ % of the whole image region.

For extracting our *car confidence feature* we apply first the vehicle detector, described in Section 2.2, which is based on the SVM approach, which in its turn is based on histograms of oriented gradients [Kos+13b]. For calculating HOG, we make use of a sliding window with

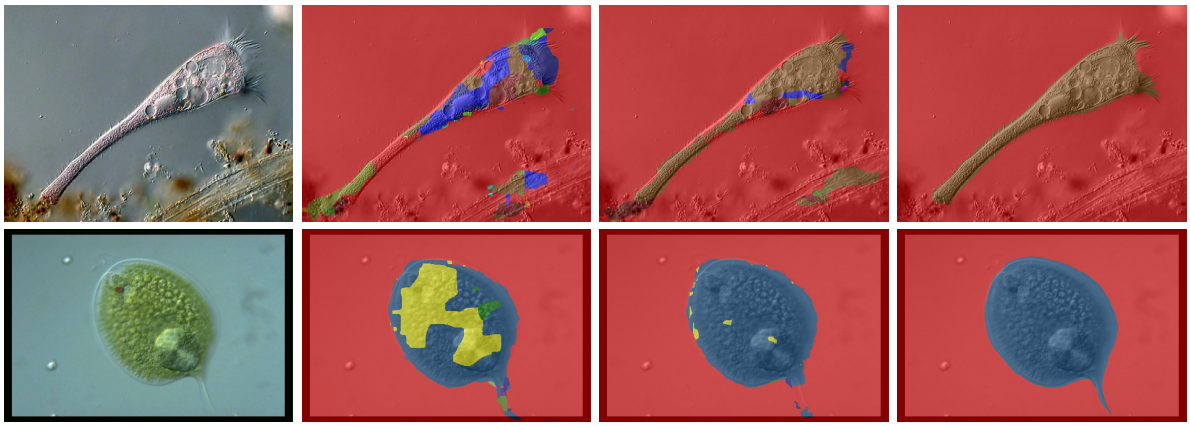


Fig. 2.12: Segmentation results for two EMs: *Stentor* (top row) and *Stylonychia* (bottom row), achieved in *RF*:*Potts* experiment. Red-colored pixels indicate that they are classified (labeled) as *background* while other colors represent pixels classified into different EM classes.

Fig. 2.13: Posterior probability achieved with the SVM approach. **Left**: Input image; **right**: Resulting car confidence image.

size of 80×80 pixels. This window consists of 100 non-overlapping blocks with size of 8×8 pixels each. For each block the gradient vectors are gathered in 9 bins an resulting HOG is achieved.

Platt scaling technique, described in the Section 2.3.2.2 allows us to convert binary output of the SVM-based vehicle detector into posterior probabilities $p(x_i = car\,|\,\mathbf{y})$ for each site i. Our car confidence feature is calculated directly from these probabilities. An example feature map for one scene is illustrated in the Figure 2.13.

In the Figure 2.14 the recall and precision for different thresholds for the estimated vehicle probabilities are shown. For this evaluation, the center point of the connected sites, which have probabilities of being a vehicle larger than the threshold, is compared to the corresponding site in the groundtruth label map (the second column of the Figure 2.15). Connected regions, which cover multiple vehicles (as on the last column of the Figure 2.15) are counted only

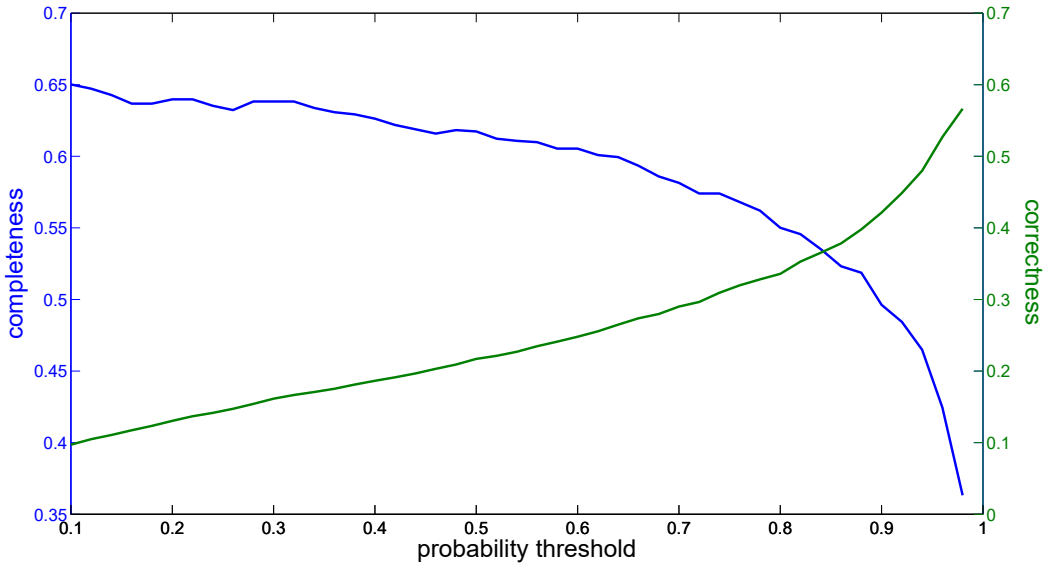


Fig. 2.14: Receiver operating characteristics for varying thresholds of probability.

once, what leads to a significant reduction of recall.

	RF:$NoEdge$		RF:$NoEdge_{car}$		RF:$hMat$		RF:$hMat_{car}$	
Category	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.
asphalt	79,8 %	85,5 %	79,8 %	85,5 %	80,8 %	86,5 %	80,7 %	86,4 %
building	83,8 %	77,8 %	83,7 %	77,6 %	84,6 %	79,0 %	84,5 %	78,9 %
grass	82,9 %	79,0 %	82,9 %	79,4 %	84,9 %	79,1 %	84,7 %	79,6 %
agriculture	52,9 %	61,1 %	54,9 %	62,0 %	52,5 %	68,5 %	54,7 %	68,8 %
tree	86,3 %	56,2 %	85,6 %	56,3 %	87,8 %	56,8 %	87,2 %	57,1 %
car	75,1 %	8,9 %	77,6 %	10,8 %	31,7 %	32,9 %	34,2 %	41,6 %
overall accuracy	**77,9 %**		**79,0 %**		**83,8 %**		**84,1 %**	

Tab. 2.4: Incorporation of the car confidence feature. Recall (Rec.) and Precision (Prec.) of the experimental results.

Here for the evaluation we used two experimental setups: RF:$NoEdge$ and RF:$hMat$ models for association and interaction potentials. In order to estimate the impact of our car confidence feature, we carried out 2 different experiments for both experimental setups: RF:$NoEdge$, RF:$NoEdge_{car}$ and RF:$hMat$, RF:$hMat_{car}$. All the experiments were performed with the use of the same set of features, described in Appendix D, but with one exception: experiments, whose name has the subscription $*_{car}$, incorporate additional car confidence feature. The recall and the precision of the results achieved in these experiments are shown in the Table 2.4. Figure 2.15 presents resulting label maps for two example scenes.

As we can see from the Table 2.4 the overall accuracy of all experiments differs within

five – seven percents, what is explained by strong hit ratio of the random forest classifier. There is also a minor difference between overall accuracies of the experiments $RF{:}hMat_{car}$ and $RF{:}hMat$. This is expected, since there are very few regions, covered by cars, and so, the car confidence feature, which has monotonically low response on the major part of images, does not affect the overall accuracy much [Kos+13b].

The main improvement in the Table 2.4 we can observe for the class *car*. The precision for it in the $RF{:}NoEdge_{car}$ experiment was only 10,8 %, while the use interactions in the $RF{:}hMat$ experiment improves this number till 41,6 %. In comparison with the results, which were achieved with experiment $RF{:}hMat$, *i.e.* without car confidence feature, we achieve about 10 % precision improvement: 41,6 % for $RF{:}hMat_{car}$ versus 32,9 % for $RF{:}hMat$. Here we get also improvement ratio for the recall: 34,2 % for $RF{:}hMat_{car}$ versus 31,7 % for $RF{:}hMat$. The $RF{:}NoEdge$ experiments (with use of car confidence feature and without) produce both better results for the aim class *car* in terms of recall, while producing very poor precision ration. This is explained by too many false positives, produced by random forest model. These false positives are eliminated by the CRF smoothness term in the $RF{:}hMat$ experiments.

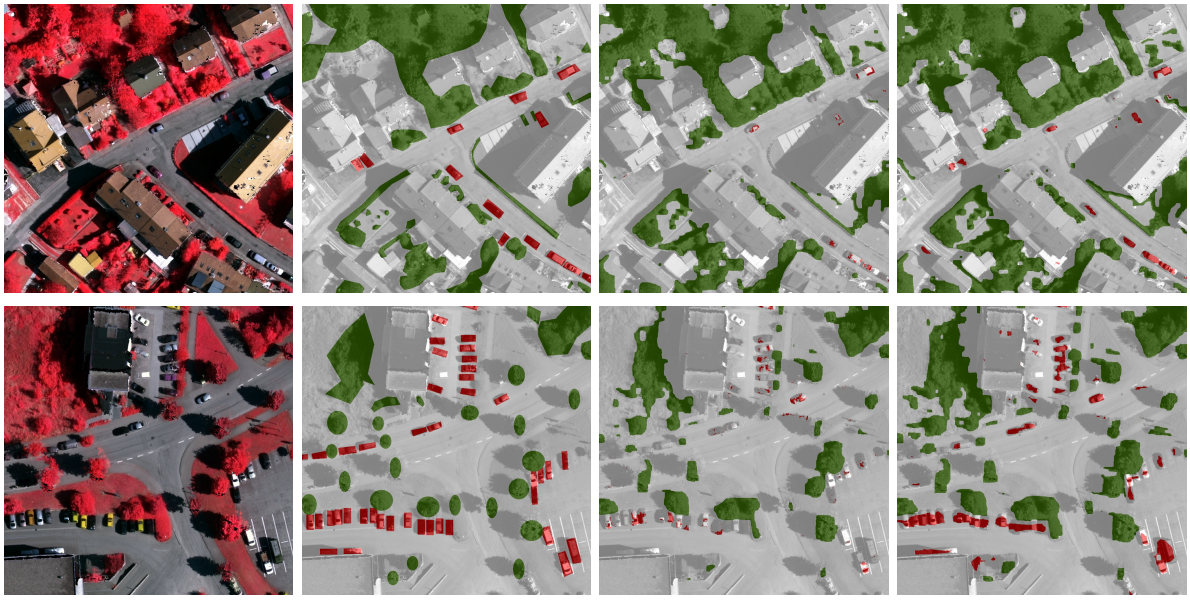


Fig. 2.15: Classification results of the scenes 23 (top row) and 36 (bottom row) of Vaihingen dataset. **First column:** Original images; **Second column:** Groundtruth; **Third column:** $RF{:}hMat$; **Fourth column:** $RF{:}hMat_{car}$ powered with car confidence features. White: *void*; dark-green: *tree*; red: *car*.

Figure 2.15 illustrates two scenes, which are rich of cars. Its third column presents the results of the $RF{:}hMat$ experiment, while the fourth column - results of the $RF{:}hMat_{car}$

experiment. We can see, that the use of the car confidence feature greatly improves the classification rate for cars. In comparison to the groundtruth (second column of the Figure 2.15) cars are over-smoothed and hardly recognizable in the results of the $RF{:}hMat$ experiment. The $RF{:}hMat_{car}$ experiment delivers us the results with the distinct car regions on the right places (almost no false positives).

2.7.1.4 Feature Importance Evaluation

In many classification problems it is important to compare contributions of distinct features to the final result. One way to do so is to build the feature importance vector as described in Section 2.3.2.3. Here, we evaluate the importance of the 16 features, used in the experiments of the Section 2.7.1.3, including the car confidence feature: *car*. Figure 2.16 depicts relative importance of these features. As we can see, our random forest classifier finds features $nDSM_1$, sat_1 and $NDVI_{21}$ as well as var_{sat} most useful, while features $NDVI_1$, *gradient* and var_{grad} less useful.

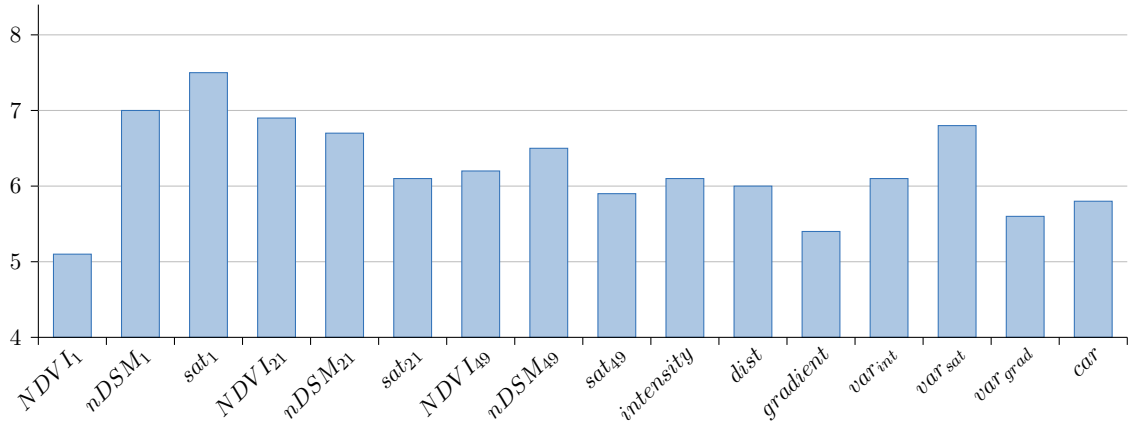


Fig. 2.16: Normalized feature importance vector [%].

2.7.2 Robust Association Potentials

In Section 2.3 we have discussed a variety of possible models for the association potentials. Let us now illustrate their impact on the computation of the label maps. For this purpose we use synthetic *Green Field* data-set, with 3 classes, described by two features. Since all the features are quantized by 8 bit, we can map the whole dataset to the 2-dimensional 256×256 feature space. If we accumulate the sample points in such representation, it will correspond to the probability densities. For the visualization we will mark these densities, belonging to different classes, with three different colors: red, green and blue (see Figure 2.17).

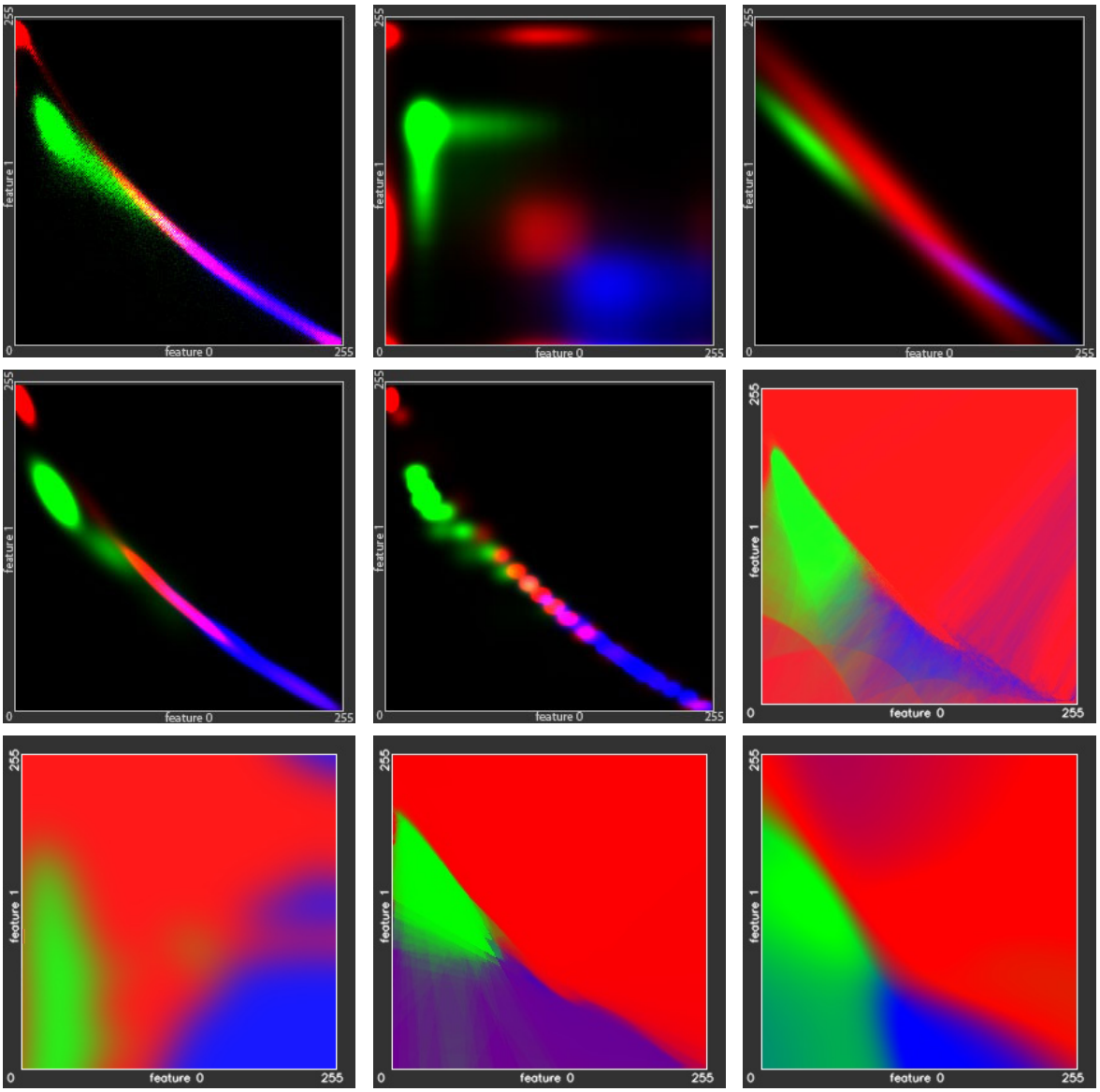


Fig. 2.17: The *Green Field* dataset, defined on a 2-dimensional feature space $\mathbf{f}(\mathbf{y}) \in \mathbb{R}^2$ with 3 classes, depicted by red, green and blue colors. **Top row**, **left:** The original distributions of 160'000 samples from the dataset; **center:** *Bayes* model; **right:** Gaussian Model: the distribution is approximated with a single Gaussian per class; **Middle row**, **left:** $seqGMM$ model; **center:** $emGMM$ model; **right:** k-NN model; **Bottom row**, **left:** SVM model; **center:** RF model; **right:** ANN model.

As we can observe from the Figure 2.17 the generative models try to reproduce the original distributions. In order to do this precisely, a method needs to remember all the $1,6 \times 10^6$ samples from the Green Field dataset. Or, in general, restricting ourself to the 8-bit features, a method needs to remember $k \cdot 256^m$ values, where k is the number of categories and m is the number of features. The main idea of the generative models is to rebuild the original distribution using much less parameters and therefore generalize the model for samples, that

were not observed during training. Bayes model approximates the distribution using only $k \cdot 256 \cdot m$ parameters, and the Gaussian mixture model – $k \cdot G \cdot (m^2 + m)$ parameters, where G is the number of Gaussians in the mixture.

As opposed to the generative models, the discriminative models do not approximate the original distributions, but provide direct predictions for all testing samples (in the Section 2.3.2 we described a number of methods for converting these direct predictions into the potentials). This grants the discriminative models more generalization power: In the areas, where hardly any training sample was met (left bottom and right top corners of the initial distribution image on Figure 2.17) all the generative models show black areas with almost zero potentials, while all the discriminative models show a high confidence about the class labels for these areas.

2.7.2.1 Sequential GMM Model

In the second part of evaluation of the association potentials we compare performance of CRFs with different models in the data term. For the interaction model we took the data-dependent approach $hMat$. Altogether we carried out 3 experiments. In the first experiment, we used the naïve Bayes model 2.20 ($Bayes$:$hMat$), whereas in the second and third experiments we used the GMM model 2.35. The difference between latter two is that in the second experiment ($emGMM$:$hMat$) GMM training is based on the OpenCV implementation of EM [BK08], and in the third – on our sequential GMM model ($seqGMM$:$hMat$) [KRH13]. The precision and the recall of the results achieved in these experiments are shown in Table 2.5.

	$Bayes$:$hMat$		$emGMM$:$hMat$		$seqGMM$:$hMat$	
Category	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.
asphalt	83,7 %	77,1 %	93,8 %	58,0 %	93,7 %	58,2 %
building	65,7 %	92,2 %	72,2 %	92,0 %	72,0 %	92,2 %
grass	51,6 %	75,9 %	58,5 %	81,0 %	58,6 %	81,0 %
agriculture	36,2 %	96,9 %	47,5 %	97,4 %	47,5 %	97,9 %
tree	82,9 %	58,5 %	71,6 %	64,2 %	71,8 %	64,3 %
car	0,5 %	17,8 %	1,2 %	40,5 %	1,3 %	40,7 %
overall accuracy	**84,5 %**		**86,1 %**		**86,0 %**	

Tab. 2.5: Comparison of impact of different association potential models on classification. Recall (Rec.) and Precision (Prec.) of the experimental results.

As we can see, the results achieved with $seqGMM$:$hMat$ are very similar to those achieved for the $emGMM$:$hMat$ and in the same time are better than the $Bayes$:$hMat$ model. In the presented results, GMM outperforms Bayes for all the classes except class *tree*, where it loses in average 10% of recall, result in a small difference in overall accuracies of the methods. The

advantage of our $seqGMM{:}hMat$ method will be more clear if we look at the Table 2.6.

Time	$Bayes{:}hMat$	$seqGMM{:}hMat$	$emGMM{:}hMat$
training:	173 sec	1602 sec	9740 sec
classification:	6,4 sec	12,5 sec	64,0 sec
RAM	1,2 MB	1,5 MB	43500 MB

Tab. 2.6: Timings for Intel® Core™ i7-4820K CPU with 3.70 GHz required for training on 1016 scenes from Vaihingen dataset and classification on 1 scene, and memory consumption (RAM).

The computation times for training our CRF model on 1016 images were 173 and 1602 seconds for the $Bayes{:}hMat$ and $seqGMM{:}hMat$ models, respectively; the time for classification was 6,4 and 12,5 seconds, respectively, per image. The memory consumption was slightly above 1 MB in both cases. For the $emGMM{:}hMat$ experiment the computation times were 9740 seconds for training and 64 seconds for classification one image, with a memory consumption of 42.5 GB. Thus $seqGMM{:}hMat$ is much closer to the $Bayes{:}hMat$ in terms of calculation time and memory requirements, while being close to $emGMM{:}hMat$ in terms of classification accuracy.

2.7.2.2 Efficient KNN Model

The K-nearest neighbors model is the most simple and naïve discriminative model, which provides extremely accurate classification results especially for low-dimensional feature spaces. However, the application of the KNN model in practical applications is problematic because of its low-speed performance for large datasets represented in high-dimensional feature spaces and for the large number of neighbors - K. In this experiment we address exactly this problem of the KNN model.

Our implementation of the KNN model [Kos15] is based on the KD-tree data structure, which is used to store points in k-dimensional space. Leafs of the KD-tree store feature vectors with corresponding groundtruth and every such feature vector is stored in one and only one leaf. Tree nodes correspond to axis-oriented splits of the space. Each split divides space and dataset into two distinct parts. Subsequent splits from the root node to one of the leafs remove parts of the dataset until only small part of the dataset (a single feature vector) is left.

KD-trees allow to efficiently perform searches "K nearest neighbors of N". Considering number of dimensions k fixed, and dataset size N training samples, the time complexity for building a KD-tree is $O(N \cdot \log N)$ and for finding K nearest neighbors – close to $O(K \cdot \log N)$. However, its efficiency decreases as dimensionality k grows, and in high-dimensional spaces KD-trees give no performance over naive $O(N)$ linear search.

In order to evaluate the performance of our KNN model, described in Section 2.3.2.1 we perform a number of experiments: $2r$-KNN, $4r$-KNN, $8r$-KNN, $16r$-KNN and $32r$-KNN – models, where the nearest neighbors enclosed between two spheroids of radii r and $2r$ ($4r$, $8r$, $16r$ and $32r$ respectively) are only taken into account (refer to Figure 2.8 for details). In the ∞r-KNN experiment all the K neighbors were considered. And finally the KNN experiment is the OpenCV implementation of KNN [BK08] based on linear search. The overall accuracies for experiments $2r$-KNN and KNN on Vahingen dataset are given in Table 2.8 and the timings for all 7 experiments are given in Table 2.7.

Time	$2r$-KNN	$4r$-KNN	$8r$-KNN	$16r$-KNN	$32r$-KNN	∞r-KNN	KNN
training:	4659 sec	4659 sec	4659 sec	4659 sec	4659 sec	4659 sec	102 sec
classification:	8,3 sec	22,2 sec	52,8 sec	97,2 sec	134,9 sec	216,1 sec	45,3 sec

Tab. 2.7: Timings for Intel® Core™ i7-4820K CPU with 3.70 GHz required for training on 1016 scenes from Vaihingen dataset and classification of 1 scene.

As we can observe from Tables 2.7 and 2.8 our $2r$-KNN model gives almost the same overall accuracies as the reference KNN model, but needs almost 5.5 times less time. The training time of the xr-KNN models, which includes the building of the KD-tree, takes 78 minutes, what is much more slower then 1,7 minutes for KNN training. However, the training in practical applications is performed only once and could be done offline, when the classification time is more critical for the whole classification engine performance. In Table 2.7 we can also observe almost linear increase of the classification time with increasing the outer spheroid radius to $4r$, $8r$, *etc.* Figure 2.18 shows the classification results on the Green Field dataset for the experiments $2r$-KNN – ∞r-KNN. The classification result of the KNN experiment is depicted at Figure 2.17 middle row, right.

2.7.3 Spatial Regularization

In this section we evaluate and compare 5 interactions models: *NoEdge*, *Potts*, *PottsCS*, *hMat* and *Concat*. For versatile comparison we perform evaluation of these models in combination with all 8 association potentials from Sections 2.7 and 2.7.2. Thus altogether we have performed 40 experiments. The resulting comparison should show the impact of the respective formulation of the interaction terms. Figure 2.19 shows the results for one crossroad [KRH12].

The overall accuracies of the results achieved in the 40 experiments are shown in Table 2.8. In the experiments for the *NoEdge* model, where CRFs were not in use, the average overall accuracy of the classification was 76,78 %. Figure 2.19 also shows the local variation of the class labels, caused by a similar appearance of the classes in the labels that is not compensated by a smoothness term [KRH12]. For the experiments with *Potts* interaction model,

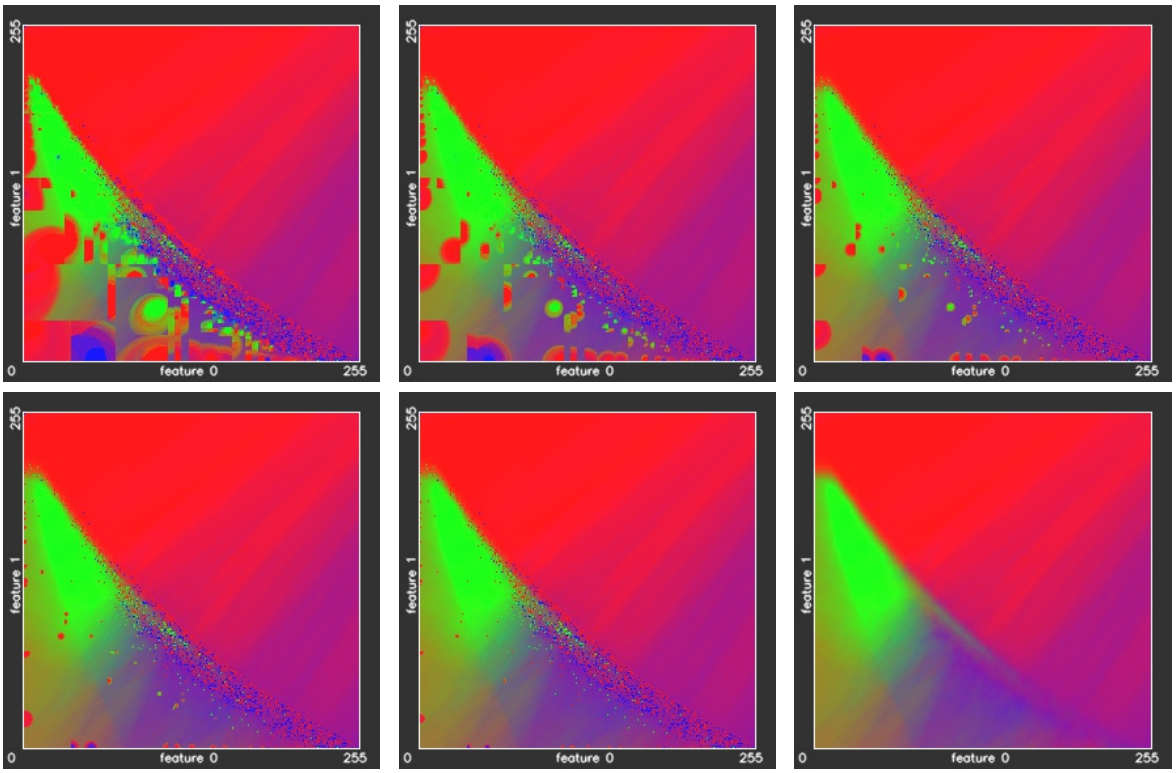


Fig. 2.18: Comparison of different xr-KNN models on Green Field dataset. **Top row:** from left to right: 2r-KNN, 4r-KNN and 8r-KNN models; **Bottom row:** from left to right: 16r-KNN and 32r-KNN and ∞r-KNN models.

the average overall accuracy was 83,4 %, a value that could be increased to 85,4 % in the experiments $hMat$.

Obviously, the smoothing achieved with the Potts models (in *Potts* and *PottsCS* experiments) already had a positive impact on the classification accuracy. Considering the data in the interaction terms in $hMat$ improves the overall accuracy even further by avoiding over-smoothing at region boundaries having sufficient contrast. In particular, the classification rate for class *asphalt* is improved considerably by avoiding confusions with class *car* (See Figure 2.19). The main error source was a confusion of buildings with asphalt and of trees with grass due to errors in the DSM caused by areas with hardly any texture (buildings) or abrupt height changes (trees).

Note that the *PottsCS* model gives in average only 81,68 % of overall accuracy. However, its highest rate (over 85,43 % – 85,59 %) fall on KNN models, making the contrast-sensitive Potts model more preferable to use in couple with K-nearest neighbors model. The *Concat* model in its turn is more suitable for the random forest association potentials. Experiment RF:*Concat* gave 85,76 % overall accuracy, what is 1,66 % better than RF:$hMat$ experiment.

Finally, Table 2.9 shows the time required for training on 1016 scenes and classification 1

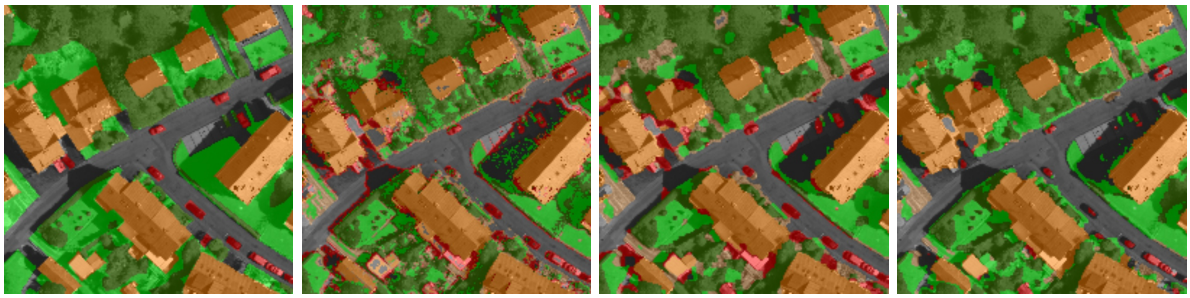


Fig. 2.19: Classification result for scene 23 of Vaihingen dataset. **Left to right:** Reference; *Bayes*:*NoEdge*; *Bayes*:*Potts*; *Bayes*:*hMat*. Grey: *asphalt*; orange: *building*; green: *grass*; beige: *agriculture*; dark-green: *tree*; red: *car*.

		pairwise potentials				
		NoEdge	*Potts*	*PottsCS*	*hMat*	*Concat*
unary potentials	*Bayes*	73,98 %	83,34 %	79,12 %	84,50 %	83,59 %
	emGMM	67,71 %	78,71 %	76,77 %	86,10 %	83,33 %
	seqGMM	67,64 %	81,47 %	77,40 %	86,00 %	83,72 %
	KNN	82,36 %	85,21 %	85,43 %	86,18 %	85,76 %
	2r-KNN	81,39 %	85,80 %	85,59 %	86,28 %	86,01 %
	SVM	79,20 %	83,15 %	81,12 %	83,59 %	83,83 %
	RF	79,00 %	83,37 %	83,10 %	84,10 %	85,76 %
	ANN	82,93 %	86,12 %	84,89 %	86,43 %	86,01 %
	average	76,78 %	83,40 %	81,68 %	85,40 %	84,75 %

Tab. 2.8: Comparison of impact of different interaction and association potential models on classification. Overall accuracies of the experimental results, performed on Vaihingen dataset.

scene on an Intel® Core™ i7-4820K CPU with 3.70 GHz. For sake of clarity, here we compared between all the interaction potentials models in couple with the *Bayes* model, which is the fastest association potential model (See Table 2.6).

2.7.3.1 Local-Global CRF

In Sections 2.7.1.1 and 2.7.1.2 we concentrated on evaluation of pixel-level EM classification results using global features. Here, in addition to the experiments *RF*:*NoEdge* and *RF*:*Potts* (Section 2.7.1.1), we carry out an experiment *RF*:*PottsCS*. Furthermore, to examine the

Time	*NoEdge*	*Potts*	*PottsCS*	*hMat*	*Concat*
training:	113 sec	113 sec	113 sec	173 sec	695 sec
classification:	0,3 sec	5,9 sec	6,2 sec	6,4 sec	6,6 sec

Tab. 2.9: Timings for Intel® Core™ i7-4820K CPU with 3.70 GHz required for training on 1016 scenes from Vaihingen dataset and classification on 1 scene.

effectiveness of our local-global CRF [KRH12], it is compared to an advanced CRF model, *denseCRF*, where pairwise potentials are defined by fully connecting all possible pairs of pixels, in order to capture a variety of spatial relations and especially recover detailed local structures [KK11]. In particular, the original DeepLab approach uses *denseCRF* in the post-processing phrase [Che+16]. Here, DeepLab-VGG-16 is not used as a feature extractor, but used to obtain an initial segmentation result that is then refined by *denseCRF*. We name this approach as $denseCRF_{org}$ and examine its performance, in order to check the effectiveness of our approach that distills outputs at the penultimate layer of DeepLab-VGG-16 as pixel-level features and uses them in other CRFs.

Moreover, our approach is compared to a currently popular DCNN-based image segmentation model, *Fully Convolutional Network* (FCN) [LSD15]. In FCN, a pre-trained DCNN for image classification is re-purposed to segmentation using deconvolution layers, which upsample low-resolution feature maps into the ones that has the original image size and represent pixel-level classification. A version of FCN based on VGG-16 is selected for fair comparison to our approach. In particular, we choose FCN-32s where feature maps after (two customized convolution layers following) the top max-pooling layer in VGG-16 (see the left side in Figure 2.4) are enlarged into image-size feature maps based on a deconvolution layer with a stride size 32[8]. FCN-32s is trained on EM images by following the network structure and hyper parameters defined for voc-fcn32s[9].

Regarding evaluation measures for segmentation results, Mean AP served as a good measure for justifying our choice of features, but it left mainly unclear the impact of applying CRFs. Also, a precision is not considered suitable for the following reason: Figure 2.20 show segmentation results by our local-global CRF (using Potts pairwise potentials) and $denseCRF_{org}$ for three EM images. As can be seen from ground truth images, each EM has a fine-grained structure such as radial lines of ω_1 or thin tails of ω_4 and ω_9. One method like our local-global CRF roughly covers the whole region of the EM, while another like $denseCRF_{org}$ only captures its main part. The latter essentially gets a higher precision than the former. However, in practice, the whole region of an EM is more meaningful for a user than its partial region, so as to avoid missing its appearance or misunderstanding its structural characteristic. Hence, a recall is used as our main evaluation measure for segmentation results, and an overall accuracy (OA) is used as an auxiliary measure to check how similar extracted EM regions are to the ground truth. OA is computed only for the case considering all classes. If one tries to

[8] FCN-32s only analyzes coarse-level feature maps. Some versions of FCN (FCN-16s and FCN-8s) support a 'skip' architecture to fuse coarse- and fine-level feature maps. However, our preliminary experiments showed that this yields no performance improvement (FCN-16s and FCN-8s get (63.63% (average recall), 95.31% (OA)) and (64.34%, 95.48%), respectively), and often causes over-segmentation of EM regions into small meaningless ones.

[9] `https://github.com/shelhamer/fcn.berkeleyvision.org/tree/master/voc-fcn32s`

compute an OA for each class, it is equal to a recall.

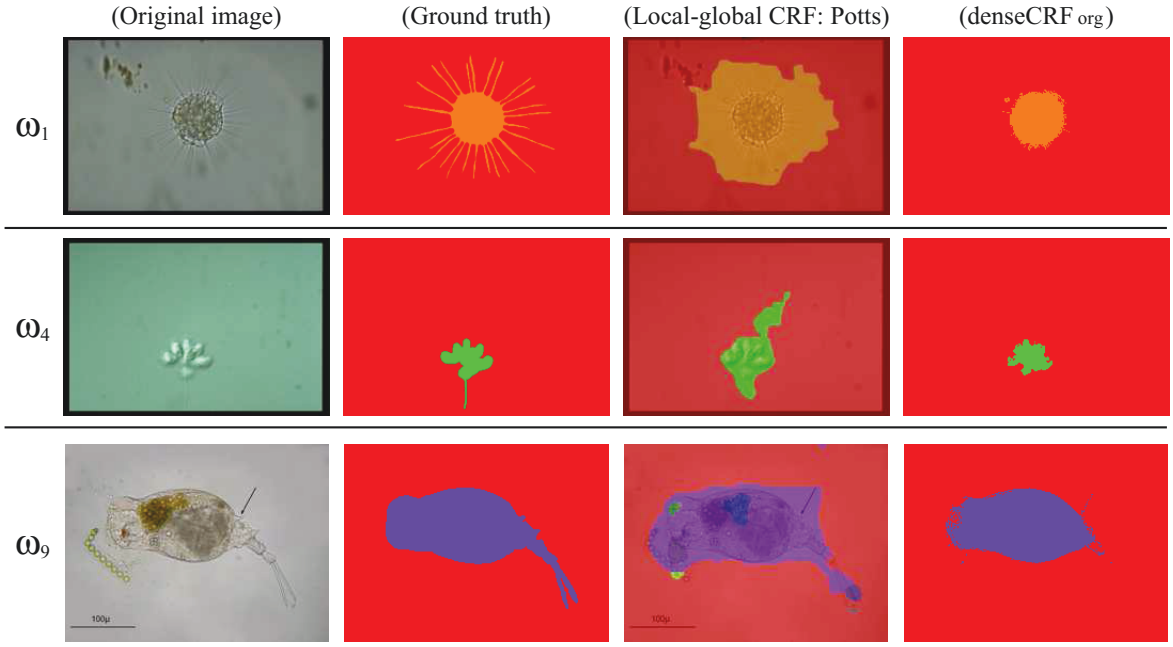


Fig. 2.20: Examples of segmentation results for EMs that have fine-grained structures.

In order to evaluate the segmentation quality in more detail, let us consider the recall for 21 classes as well as the OA values in Table 2.10. We see that the use of only unary potentials in *RF:NoEdge* experiment gives poor OAs, whereas the application of local edge potentials increases the segmentation accuracy more than by 10%. Such a great improvement is conditioned by the fact that the major part of the EM images is covered by the *background* class (which is not considered in Mean AP evaluation), and exactly the improvement on that class leads to the leap in OA values in Table 2.10. This may be more clear by looking at the recall values for the *background* class. In its turn, the average recall values for the local-global CRF models outperform both *denseCRF* models: 74.76% – 79.40% for local-global CRF versus 67.87% – 68.85% for *denseCRF*.

The example segmentations are shown in Figure 2.21, which indicates some difficult cases with semi-transparent EMs, *e.g.*, *Arcella* (ω_2), *Codosiga* (ω_4), *Ceratium* (ω_{12}) and *Synchaeta* (ω_{20}), where it is hard for our method to find their invisible body parts. For example, ω_4 has class-specific antennas, which are labeled correctly, while the semi-transparent inner body is labeled wrongly. The same we observe for ω_{20} where transparent body parts have very similar features as the background and other EMs, so they are labeled wrongly. For the difficult cases RF labels only a particular part of an EM as the relevant object, where the classification result may be still correct, but segmentation is incomplete. Because this particular part contains very specific characteristics, only using this part is regarded as leading the most accurate

Class	local-global CRF			*denseCRF*	*denseCRF*$_{org}$	FCN
	RF:*NoEdges*	*RF*:*Potts*	*RF*:*PottsCS*	*Gaussian*	*Gaussian*	
background	83.02 %	96.24 %	96.28 %	97.13 %	98.94 %	99.22 %
ω_1	91.43 %	81.47 %	79.22 %	75.00 %	53.86 %	45.78 %
ω_2	88.70 %	90.25 %	86.59 %	85.57 %	87.45 %	89.19 %
ω_3	94.13 %	88.86 %	93.11 %	84.49 %	83.50 %	85.10 %
ω_4	87.04 %	64.61 %	62.73 %	46.19 %	41.57 %	37.08 %
ω_5	75.48 %	72.75 %	72.17 %	68.30 %	74.59 %	80.88 %
ω_6	54.77 %	13.80 %	21.09 %	7.36 %	0.50 %	15.80 %
ω_7	79.93 %	81.07 %	76.12 %	71.36 %	80.22 %	83.11 %
ω_8	83.93 %	86.37 %	79.96 %	75.50 %	84.49 %	77.32 %
ω_9	45.82 %	42.93 %	45.74 %	33.08 %	35.64 %	44.22 %
ω_{10}	87.96 %	86.56 %	84.03 %	75.29 %	86.25 %	69.83 %
ω_{11}	92.57 %	93.57 %	91.92 %	90.63 %	90.19 %	83.44 %
ω_{12}	83.82 %	77.28 %	77.16 %	69.57 %	56.87 %	37.13 %
ω_{13}	75.19 %	81.62 %	74.05 %	62.78 %	68.10 %	55.76 %
ω_{14}	93.23 %	86.41 %	87.31 %	85.18 %	82.54 %	59.08 %
ω_{15}	79.20 %	81.39 %	76.07 %	67.46 %	68.45 %	68.06 %
ω_{16}	71.29 %	73.66 %	73.81 %	60.28 %	69.64 %	82.21 %
ω_{17}	65.35 %	64.52 %	65.20 %	60.95 %	65.12 %	65.87 %
ω_{18}	82.65 %	86.98 %	83.76 %	74.63 %	84.53 %	57.59 %
ω_{19}	65.77 %	64.39 %	61.05 %	53.63 %	62.82 %	27.88 %
ω_{20}	86.02 %	82.65 %	82.67 %	80.89 %	70.95 %	71.57 %
average:	79.40 %	76.07 %	74.76 %	67.87 %	68.85 %	63.62 %
OA:	82.63 %	94.19 %	93.98 %	94.05 %	95.85 %	95.48 %

Tab. 2.10: Recall and Overall Accuracy (OA) of the experimental results with DCNN features. Comparison between six classification models: per-pixel RF (*RF*:*NoEdges*), CRF with Potts pairwise potentials (*RF*:*Potts*), CRF with contrast-sensitive Potts model (*RF*:*PottsCS*), fully connected CRF with Gaussian pairwise potentials (*denseCRF*), fully connected CRF on segmentation results by the original DeepLab method [Che+16] (*denseCRF*$_{org}$), fully convolutional network (*FCN*).

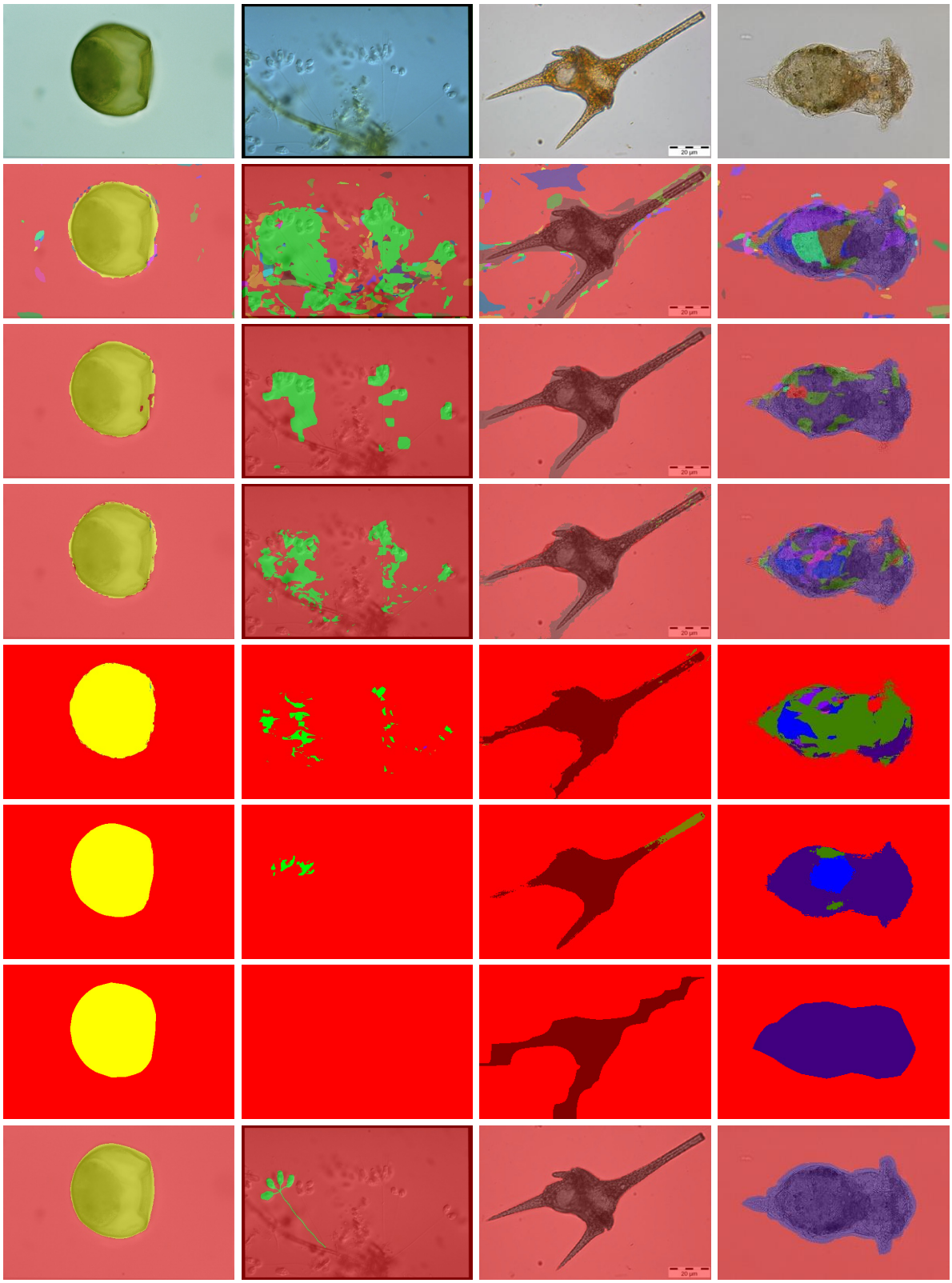


Fig. 2.21: Segmentation results for 4 EMs (ω_2, ω_4, ω_{12}, ω_{20}) with DCNN features. 1^{st} row: input image; 2^{nd} row: local-global CRF RF:$NoEdges$; 3^{rd} row: local-global CRF RF:$Potts$; 4^{th} row: local-global CRF RF:$PottsCS$; 5^{th} row: $denseCRF$; 6^{th} row: $denseCRF_{org}$; 7^{th} row: FCN; 8^{th} row: groundtruth.

classification. Table 2.10 and Figure 2.21 show that our local-global CRF is better than FCN [Kos+18]. The confusion matrix of the result achieved in the experiment *RF*:*Potts* is shown in Figure 2.22.

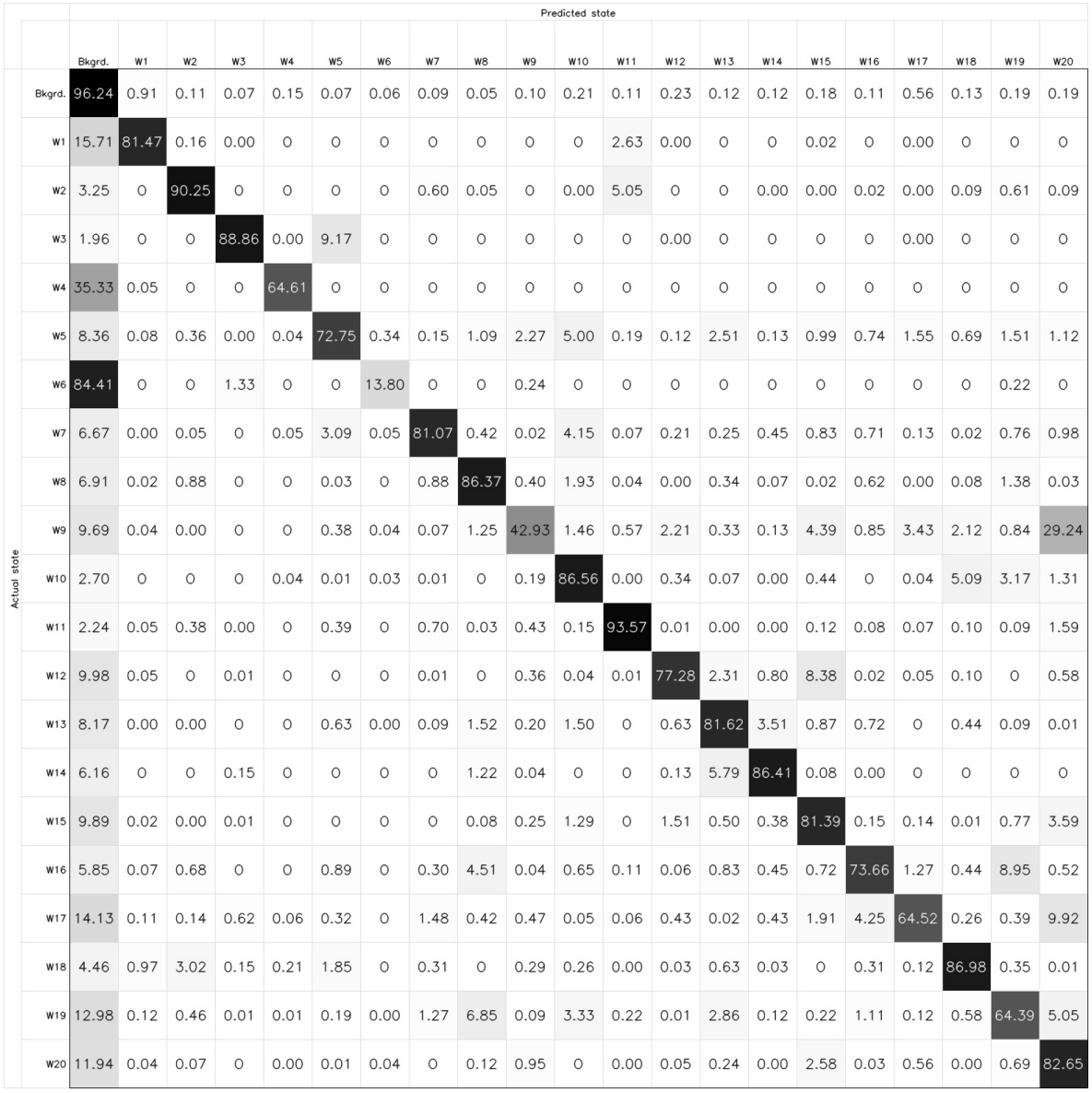


Fig. 2.22: Confusion matrix for the experiment *RF*:*Potts* with use of DCNN features. All values are given in [%]. Overall accuracy: 94.2%.

Comparison to Existing Methods In this section we compare our CRF approach with the *Region-Based Support Vector Machine* (RBSVM) [LSG15b], which can also localize EMs with bounding boxes. The results for RBSVM are available only for the first 15 classes of EMDS dataset and only in terms of Mean AP. We compare the performance of 3 methods: **1**.

RBSVM: RBSVM based on SIFT-BoVW features is trained on 15 classes. *Bag-of-Visual-Words* (BoVW) involves two steps: The first step is to obtain a set of visual words by clustering a large number of SIFT features. Each cluster center represents a characteristic SIFT feature and is regarded as a visual word. Given a region in an image, RBSVM creates a histogram of visual words by assigning each SIFT feature in this region to the most similar visual word. RBSVM localizes an EM to the region from which the histogram maximizing the SVM score is obtained [LSG15b]. **2**. *denseCRF*: denseCRF based on DCNN features is trained on 20 classes; **3**. *local-global CRF*: our CRF approach based on DCNN features which is also trained on 20 classes. Please note, that the CRF-based approaches are trained not on 15, but on 20 classes. Nevertheless, the comparison is fair, because usually the additional classes lead to additional miss-classification, and thus to the reduction of the overall classification accuracy.

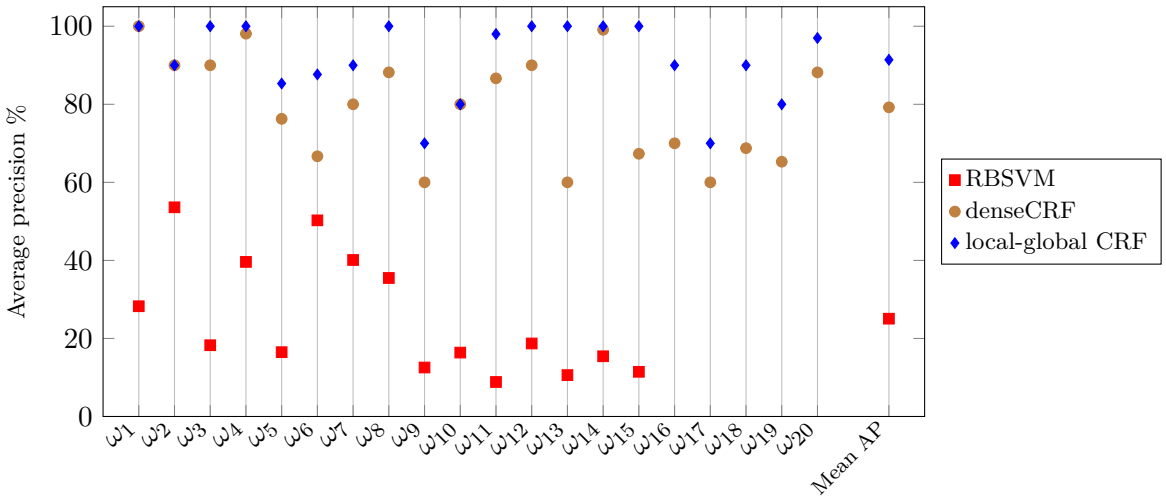


Fig. 2.23: APs and Mean APs of the EM classification results. Comparison between the RBSVM- and CRF-based approaches.

Figure 2.23 represents the AP values for every EM class as well as the Mean AP value. First, we explain how to obtain image-level classification results based on pixel-level segmentation results by our CRF approach. We consider every label $x_i = l$, $l \in \mathbb{L} \setminus \{background\}$ of a pixel y_i of the segmented scene as a vote that the scene represents EM class l. Normalizing all the votes for a scene we achieve a probability of the scene to represent l. We sort all the test scenes in terms of these probabilities and calculate the AP value for l.

As seen from Figure 2.23, the CRF-based classifiers (denseCRF and local-global CRF) significantly outperform the RBSVM classifier. The Mean AP value of RBSVM is 25.06%. Compared to these, the Mean AP of denseCRF reached 79.22% and Mean AP of local-global CRF – 91.40%. Such leap becomes possible because CRFs do not treat image patches independently as in RBSVM. The Mean AP value for local-global CRF is considerably higher

than the Mean AP value for the denseCRF, this is due to the fact that the DCNN features in our model are supported by the global ones.

Discussion about Computation Times Finally, we briefly describe computational times of our EM classification method. Pixel-level feature extraction based on a DCNN were conducted on a workstation equipped with Intel® Core™ i7-7700 CPU with 3.60 GHz, 32GB RAM and GeForce GTX 1080 8GB. By following the original implementation of *DeepLab* [Che+16], the feature extraction phase was run in the Caffe framework [Jia+14]. The re-purposing and fine-tuning step of a pre-trained DCNN (VGG-16) took 8 417 seconds, and pixel-level features for all the 400 EM images were extracted in 3 370 seconds by accessing the penultimate layer of the DCNN with the python interface (pycaffe).

Regarding the training our RF-based unary potentials on the DCNN features, building CRF and conducting inference we used another workstation equipped with Intel® Core™ i7-4820K CPU with 4.50 GHz, 64GB RAM and dual SLI GeForce GTX 780 3GB. Training and classification with our local-global CRF was implemented with the DGM library [Kos15]. Training of our DGM-based classifier on 200 EM images (approx. 4.2×10^5 training samples) took 8.9 seconds, building and initializing the graphical model for one scene – 1.9 seconds and inference for one scene took in average 3.7 seconds.

2.8 Summary

In this chapter, a detailed overview of conditional random fields and the theory behind and related to it is given. I started with a short recapitulation of well-known probabilistic models used for single-position classification. CRF in its turn can be considered as an extension to a structured learning model from the single-position classification model. From the architectural perspective, my formulation consists of three main parts: feature extraction, pixel-level classification and graph modeling.

In this chapter a considerable scientific contribution to the field of feature engineering is made. The introduction of global features and application them to the challenging problem of EM classification increased the classification rate by 7% – 17% (depending on the used local features) in terms of mean APs (see Table 2.3). The global features are used to support the classification and improve the segmentation quality by providing a long-range consistency between pixel labels. Considering the small training dataset problem, an approach was adopted where a DCNN pre-trained on large auxiliary image data is re-purposed and fine-tuned to a pixel-level feature extractor using EM images. In comparison to the state-of-the-art features, used for EM classification, DCNN features allow to increase the classification accuracy from

18% up to 54% in terms of mean APs (see Table 2.3). Finally, the introduced confidence features allow for incorporating of external object detectors into the CRF framework. Having high response on the objects of interest and low response on all other areas, confidence features preserve these objects from over-smoothing. This is extremely useful if the objects of interest have small size. The usage of a car confidence feature allows to rise the precision for class *car* from 32% up to 41% (see Table 2.4).

The next scientific contributions belong to the field of pixel-level classification. Efficient unary and pairwise potentials are essential building blocks of the CRF technique. In this thesis new efficient and high-accurate algorithms for modeling these potential functions are proposed. The sequential Gaussian mixture algorithm is approximately 6 times faster both in training and classification and needs 30 000 times less memory than state-of-the-art expectation maximization algorithm, while being slightly more accurate (see Tables 2.5 and 2.6). The efficient 2r-KNN algorithm, based on KD-tree data structure and takes into consideration only those neighbors, which lie in a small neighborhood of the nearest found neighbor. The 2r-KNN algorithm is approximately 5 times faster in classification then its popular implementation from the OpenCV library [Ope14] (see Table 2.7). The histogram-based matrix and concatenated edge potentials models for data-dependent interaction potentials outperform all existing pairwise models for CRFs (see Table 2.8).

The last but the most significant contribution of this Chapter is made to the field of graph modeling. Adding one global graph node to the classical CRF structure binds all the local graph nodes together with the longest path, consisting of only two edges. Such construction, called local-global CRF and presented in Section 2.1.4 of this thesis allows to outperform some modern classification frameworks as dense CRF [KK11] and fully convolutional network [LSD15] in terms of average recall values (see Table 2.10). Thus, experimental results validate the effectiveness of each of above mentioned scientific contributions.

However, this chapter does not only provide further insights into the design of conditional random fields, it also serves as a useful tool-kit for the construction of new approaches. By introducing a modular notation based on unary and pairwise potentials I set up a general framework that allows for both a simple development and a straightforward implementation of such techniques. The actual generality of the framework becomes obvious in the Chapter 3. There, it is extended to the development of multi-layer conditional random fields for handling occlusions in classification.

3 Multi-Layer Conditional Random Fields

"You will never solve a problem if you think the same way as those who created it."

- Albert Einstein

So far in this thesis only design of single-layer conditional random fields was considered. This technique determines a single class label for each observation, what causes problems if the objects to be classified are partially occluded. For instance, the appearance of streets, sidewalks and buildings may not be clear to a computer if they are largely occluded by objects such as cars or trees. In these cases it may even prevent the approach from estimating the correct classes not only in the occluded regions but also in the non-occluded neighborhood of these regions. Thus single-layer conditional random fields turn from a restriction into a limitation.

In this chapter I introduce the multi-layer conditional random field technique, which can handle this limitation by explicitly modeling *several* class labels for each observation, building a hierarchy of occluding and occluded objects; in this way, the 3D structure of the scene is explicitly considered. Thereby all steps – the modeling, the conditioning and marginalization, the inference and the discretization – are discussed in detail.

Section 3.1 starts with describing the occlusion model which serves as a base for the scientific contributions made in this Chapter. In order to describe occluded objects, the 3D structure of the scene is explicitly represented with a number of depth layers, on which all the objects are arranged. Thus, the probabilistic graphical model from Chapter 2 is extended by adding an extra dimension to it. This results in that for each image site there exist a set of class labels, corresponding to different depth layers of the observed scene. The layers are ordered in the direction of view, such that the lowest layer corresponds to the background (the most distant object), and higher layers – to the occluding objects. A specific class label is used to encode that no occlusion occurs. Labeling might also be supported by depth information obtained from stereo vision.

In Section 3.2 the basic theoretical background of the mixed graphs is given. Mixed graphical models are essential for implementing the described above occlusion model within the multi-layer CRF architecture. They comprise both undirected and directed links, the latter are the main reason why the robust to occlusions CRF model is based on the mixed

graphs. Directed links are better suited for expressing casual relationships between random variables and thus allow for expressing conditional dependencies of the random variables, corresponding to the occluded regions on the random variables, which correspond to the observed objects in the same way as latent variables are conditioned on observed variables in the state-of-the-art random fields. This allows the information from neighboring unoccluded objects as well as information from the occluding layers to contribute to an improved labeling of occluded objects.

The main challenge in using casual relationships between random variables in the multi-layer CRF model is that there is no prior knowledge where occlusion took place. Thus, the mixed graph, corresponding to the proposed model, should have very general and in the same time very flexible structure. In the following the main scientific contributions of this Chapter are listed:

- *Definition of the conditioning and marginalization operations for layered graphs.* In Section 3.2.2 I define operations of conditioning on– and marginalization out a subset of graph nodes, and show that these operators do not contradict the corresponding operations for probability distributions. These graph transformation operators allow to modify the graph on-the-fly during the inference process, while keeping the corresponding probability distribution consistent. Finally, such a mixed graphical model is achieved, which could be smoothly changed in sense of the cliques and its number as understanding the scene and gathering hints about occlusions. The transition from one graph state to the another is put into effect between iterations of inference.

- *Introduction of the multi-layer CRF.* In Section 3.3.1 I introduce the multi-layer graphical model, which is a subclass of the mixed graphs. It is modeled in a such way that latent variables from distant depth layers are conditioned on the latent variable from the near depth layers and all latent variables from different depth layers are directly or indirectly conditioned upon the corresponding data variables. The graph structure adjusts to the scene during the classification process, where certain hints about occlusions are reviled. Using the conditioning and marginalization operations, first I define interrelation between two consequent states of the graph (Theorem 3.2) and then extend it to any two graph states (Theorem 3.3), *i.e.* I describe graph sequential change.

- *Introduction of the inter-layer interaction potentials.* Concerning multi-layer CRF, the relations between any two neighboring class variables per site and the mutual dependencies between class variables at neighboring sites in each of the layers will be explicitly modeled. To this end, a number of prototypes for association potentials was described

in Section 2.3. These prototypes form the data term of the multi-layer conditional random fields. The potential functions which correspond to the graph edges connecting nodes from the same layer will be called as *within-layer* interaction potentials. These are described in Section 2.4. And, since the new graphical model has now 3D structure, one more kind of interaction potentials is needed. These new potential functions which correspond to graph edges, connecting nodes from different layers will be called as *inter-layer* interaction potentials. They as well as the general ideas, used for inter-layer interaction, are described in Section 3.3.2 in detail.

- *Development of the inference algorithm suited for mixed graphs and introduction of the double marginalization technique.* In Section 3.4 I research the application of message-passing algorithms for inference to the mixed graphical models. To the best knowledge of the author, the use of classical sum– and max-product inference algorithms for mixed graphs was not reported in the literature before. Hence, using Theorem 3.4, I represent the mixed graph in a form of a directed graph, keeping the probability distribution unchanged. After that any max-product message-passing algorithm from Section 2.5.2 may be used. In order to decode the occluded regions, the *double marginalization* method is proposed in Section 3.4.1.

- *Introduction of the two-layer CRF.* It is worth to mention, that in many applications two layers are sufficient. In Section 3.5 a special case of multi-layer CRF, with only one extra layer is described. This results in a two-layer CRF model.

After a brief discussion of the two-layer model current Chapter continues with a detailed evaluation of the multi-layer CRF approach in Section 3.6. In order to allow for a direct comparison to the single-layer CRF technique, the same test scenarios are used as in Chapter 2. This qualitative evaluation is followed by an conclusion, where I conclude my second main scientific contribution in Section 3.7.

3.1 Occlusion Model

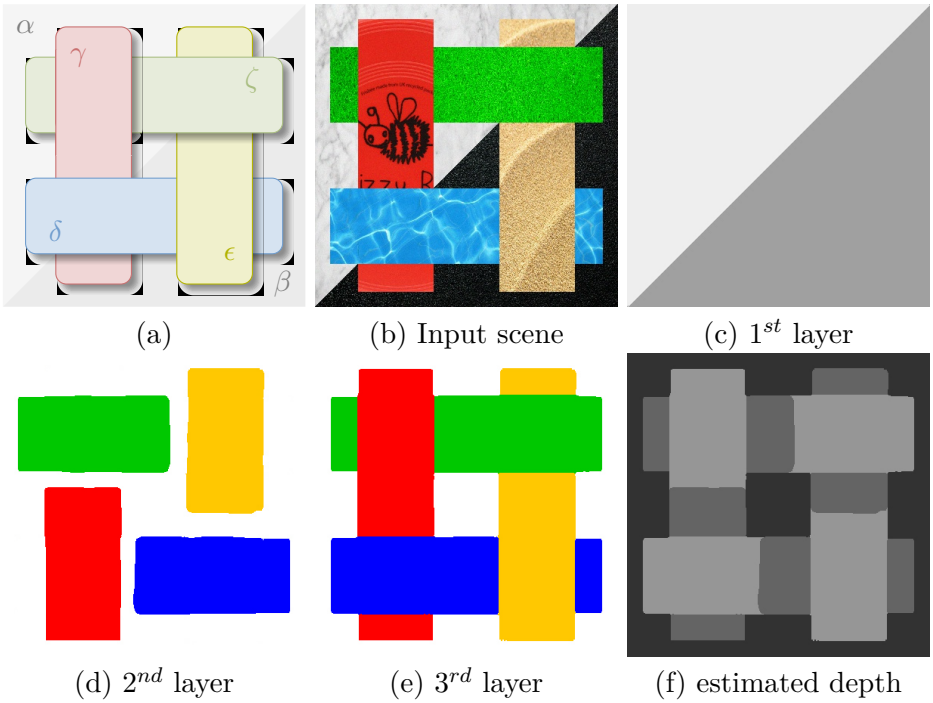
We approach the problem of handling occlusions within the CRF framework first by splitting the set of all labels of interest $\mathbb{L}$ into two non-interesting sets: one, containing labels, which correspond to background, *i.e.* to the objects that cannot occlude other objects but could be occluded, and another, containing labels, which correspond to foreground, *i.e.* all remaining labels. We will call these sets *base* and *occlusion* label sets respectively and designate them through $\mathbb{L}^x$ and $\mathbb{L}^z$, thus $\mathbb{L} = \mathbb{L}^x \dot{\cup} \mathbb{L}^z$.[1] All the following reasoning is based on the following

[1] $\mathbb{L}^x \dot{\cup} \mathbb{L}^z$ denotes the disjoint union of $\mathbb{L}^x$ and $\mathbb{L}^z$.

three assumptions [KSG18b]:

- For each observation there exists one and only one object from the base label set $\mathbb{L}^x$;
- Every object from the occlusion label set $\mathbb{L}^z$ directly or indirectly occludes an object from the base label set $\mathbb{L}^x$;
- Objects from the occlusion label set $\mathbb{L}^z$ may occlude each other.

For absence any of any other assumptions, we design our occlusion model, to handle even complex inter-occlusions as those, depicted in Figure 3.1.



(a) (b) Input scene (c) 1^{st} layer

(d) 2^{nd} layer (e) 3^{rd} layer (f) estimated depth

Fig. 3.1: (a) Example of superposable occlusions. Here objects $\{\alpha, \beta\} \in \mathbb{L}^x$ and $\{\gamma, \delta, \epsilon, \zeta\} \in \mathbb{L}^z$. (b) Synthetic scene. (c) - (e) ML-CRF classification results for 1 base layer and 2 occlusion layers. (f) Coarsely estimated depth from the scene layer decomposition.

Naturally, all the objects in scene may be ordered along the viewing axis, which is orthogonal to the image plane. For convenience, we can split the space between observer and scene background into a number $r+1$ of depth layers, parallel to the image plane. These layers will contain scene objects which are differently distant from the observer. Thus, the most distant background will form the bottom layer of objects, which we will call *base* layer; while the foreground objects will be situated on the top layers, called *occlusion* layers.

Second, we associate with each observation y_i several latent random variables, which correspond to different depth layers of the scene. Variable x_i will correspond to the base

layer, *i.e.* will represent a background object and will take labels from the base label set $\mathbb{L}^x$; variable z_i^1 will correspond to the top layer, *i.e.* to the actually observed data y_i; all other variables $z_i^2, z_i^3, \ldots, z_i^r$ will correspond to the rest of the depth layers in the direction from top to bottom, *i.e.* will represent objects, occluded by z_i^1. All variables z_i^k, $k \in [1; r]$ will have labels from the occlusion label set $\mathbb{L}^z \cup \{void\}$, where additional class *void* is used to model situations where the background is visible. The hierarchy of these variables forms multiple layers of graph nodes for the whole scene $\mathbf{y}$ as depicted at Figure 3.2.

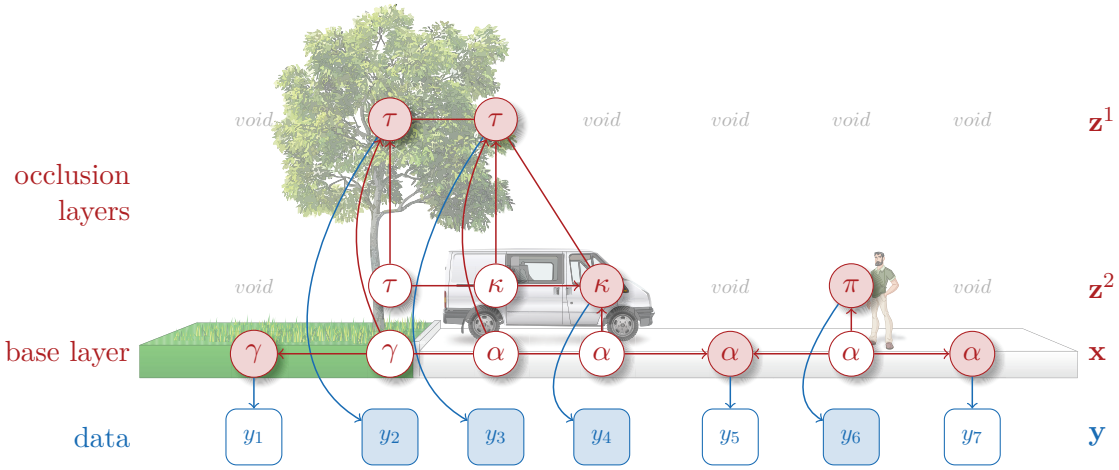


Fig. 3.2: A vertical slice of a scene, observed from a near-vertical direction: classes *grass* and *asphalt* belong to the scene background: $\mathbb{L}^x = \{asphalt(\alpha),\ grass(\gamma)\}$ and $\mathbb{L}^z = \{tree(\tau),\ car(\kappa),\ pedestrian(\pi)\}$. The imposed graph with 3 layers (base layer $\mathbf{x}$ and 2 occlusion layers $\mathbf{z}^1, \mathbf{z}^2$) illustrates our occlusion model: square and round nodes correspond to observations $\mathbf{y}$ and labels $\mathbf{x}, \mathbf{z}^1, \mathbf{z}^2$, respectively. The dark blue nodes correspond to the regions with occlusion, while dark red nodes – to the actually observed objects. Blue and red links correspond to unary and pairwise potentials, respectively.

At the Figure 3.2 occluded label nodes are depicted with bright red circles, while the actually observed label nodes are depicted with dark red circles. The proposed interaction model is also depicted with directed and undirected links. Note that the information in the depicted graph propagates in reverse direction of the arrows. As we can see, resulting occlusion model implies one base layer $\mathbf{x}$ and $r \in [1; \infty)$ occlusion layers $\mathbf{z}^1, \ldots, \mathbf{z}^r$, such that labels are distributed among two mutually exclusive label sets: $\mathbf{x} = l \in \mathbb{L}^x$, $\mathbf{z}^1, \ldots, \mathbf{z}^r = l \in \mathbb{L}^z \cup \{void\}$, where $\mathbb{L}^x \dot{\cup} \mathbb{L}^z = \mathbb{L}$. Such an architecture is capable to handle superposable occlusions, such as depicted at Figure 3.1.

Finally we have to define the interrelation between random variables from different layers of our occlusion model. Here it is important to model the graph links in such a way, that the useful information propagates from the 'non-occluded' and 'occluding' nodes in the direction of the 'occluded' nodes and no ambiguity leaves them. Thus, first we connect the 'non-

occluded' and 'occluding' nodes with the corresponding data sites, whereas 'occluded' nodes will not be linked with data nodes directly but will draw information about correct label from neighborhood, by referring to the actually observed nodes. For this purpose, we will use directed links with arrowheads pointing to the reference nodes. This implies usage of mixed graphs, that will be described further in this Chapter in Sections 3.2 and 3.3 in details.

Since there are no prior knowledge weather occlusion took place, the interaction between random variables will be driven by the ambiguity, arising when interpreting the observation citeKosovMTAP2018. In case of a 'simple' occlusion, like at site 6 at Figure 3.2 this problem is well-posed, because nodes x_6 and z_6^2 (marked with α and π respectively) take labels from different non-intersecting label sets $\mathbb{L}^x$ and $\mathbb{L}^z$. In case of a 'complex' occlusion like at site 3, we meet an ambiguity of interpreting the observation y_3 for variables z_3^1 and z_3^2: in particular it is not clear weather tree occludes car, car occludes tree or there is no occlusion at all and car and tree are neighbors at the same depth layer. We solve such arising ambiguities by applying technique, which we call *double marginalization* and which is described in Section 3.4.1 in detail.

3.1.1 Depth Map Estimation

A prior knowledge about the three dimensional structure of the classifying scene may greatly simplify the application of the multi-layered CRFs. In case, when multiple images of the scene are available, we propose an application of *optical flow techniques* for reconstructing a *depth map* of the scene [Kos08]. Variational methods are among the best performing techniques for the dense optical flow estimation, and allow for a real-time performance [KTS09]. Together with an input image, a corresponding depth map might be used for forming the feature vector that, in its turn, is used in the classification.

3.2 Mixed Graphical Models

A *mixed graph* $\mathcal{G}^m = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ is a graph containing two kinds of links: *undirected* edges ($- \in \mathcal{E}$) and *directed* arcs ($\rightarrow \in \mathcal{A}$). The nodes of such mixed graph are related as follows:

$$If \left\{ \begin{array}{c} v_1 - v_2 \\ v_1 \rightarrow v_2 \\ v_1 \leftarrow v_2 \end{array} \right\} in\ \mathcal{G}^m\ then\ v_1\ is\ a \left\{ \begin{array}{c} neighbor \\ follower \\ leader \end{array} \right\} of\ v_2.$$

Link Sequences and Paths. A *sequence of links* between vertices v_1 and v_n in $\mathcal{G}^m$ is an ordered set of links $\{l_1, l_2, \ldots, l_{n-1}\}$, such that there exists a sequence of vertexes (not necessary

distinct) $\{v_1, v_2, \ldots, v_n\}$, where link l_i has endpoints v_i, v_{i+1}. A sequence of links for which the corresponding sequence of vertexes contains no repetitions is called a *path*.

We will denote paths trough bold Greek ($\boldsymbol{\pi}$) and its parts – *subpathes* – through $\boldsymbol{\pi}(v_i, v_{j+1}) \equiv \{l_i, \ldots, l_j\}$. Note that the result of concatenating two paths with a common endpoint is not necessarily a path, though it is always a sequence. Paths consisting of a single vertex, corresponding to a sequence of no links, are permitted for the purpose of simplifying proofs; such paths will be called *empty* as the set of associated links is empty.

We define a path as a sequence of links rather than vertexes because the latter does not specify a unique path when there may be two links between a given pair of vertexes. A path of the form $v_1 \to \cdots \to v_n$, on which every links is arc of the form $\to$ with the arrowheads pointing towards v_n is a *directed path* from v_1 to v_n.

Leading and Driven Vertexes. A vertex v_1 is said to be *driven* by a vertex v_n if there is a path on which every link l_i is either of the form $v_i - v_{i+1}$, or $v_i \to v_{i+1}$ with $1 \leq i < n$, or $n = 1$; *i.e.* there are no arcs $v_i \leftarrow v_{i+1}$ pointing towards v_1. Such a path is said to be a *driven path* from v_1 to v_n. In its turn, vertex v_n is said to be *leading* for the vertex v_1.

We apply this definition distinctively to arrays:

$$\begin{aligned}\operatorname{drv}(\mathcal{V}) &= \{v_i : \ v_i \text{ is } \textit{driven} \text{ by } v_j \text{ for some } v_j \in \mathcal{V}\}, \text{ and}\\ \overrightarrow{\operatorname{drv}}(\mathcal{V}) &= \{v_i : \ \exists \text{ a } \textit{directed} \text{ path from } v_i \text{ to some } v_j \in \mathcal{V}\}.\end{aligned}$$

From this definition directly follow four properties of mixed graphs:

Proposition 3.1. $\mathcal{V} \subseteq \overrightarrow{\operatorname{drv}}(\mathcal{V}) \subseteq \operatorname{drv}(\mathcal{V})$.

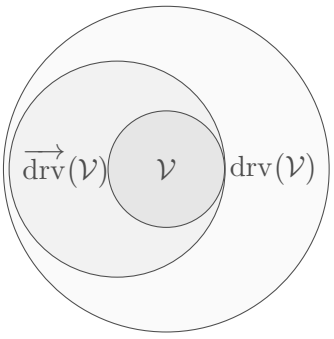


Fig. 3.3: Illustration of Proposition 3.1: an array of leading vertecies $\mathcal{V}$ is included into the array $\overrightarrow{\operatorname{drv}}(\mathcal{V})$, which is in its turn included into the array $\operatorname{drv}(\mathcal{V})$.

Proposition 3.2. $\operatorname{drv}(\mathcal{V}) \equiv \operatorname{drv}(\operatorname{drv}(\mathcal{V}))$.

Proposition 3.3. $\forall \mathcal{U} \subseteq \mathcal{V} : \ \operatorname{drv}(\mathcal{U}) \subseteq \operatorname{drv}(\mathcal{V})$.

Proposition 3.4. $\forall\mathcal{U},\forall\mathcal{V};\ \mathcal{U}\cap\mathcal{V}=\emptyset:\ \mathrm{drv}\big(\mathrm{drv}(\mathcal{V})\setminus\mathcal{U}\big)=\mathrm{drv}(\mathcal{V})$.

Proof. Since $\mathcal{V}\cap\mathcal{U}=\emptyset$, we can write $\mathcal{V}\subseteq\mathrm{drv}(\mathcal{V})\setminus\mathcal{U}$. By Proposition 3.3, $\mathrm{drv}(\mathcal{V})\subseteq\mathrm{drv}\big(\mathrm{drv}(\mathcal{V})\setminus\mathcal{U}\big)$. Vice-versa, $\mathrm{drv}(\mathcal{V})\setminus\mathcal{U}\subseteq\mathrm{drv}(\mathcal{V})$ so $\mathrm{drv}\big(\mathrm{drv}(\mathcal{V})\setminus\mathcal{U}\big)\subseteq\mathrm{drv}\big(\mathrm{drv}(\mathcal{V})\big)=\mathrm{drv}(\mathcal{V})$, by Propositions 3.2 and 3.3. □

The Propositions 3.2 – 3.4 are also valid for $\overrightarrow{\mathrm{drv}}(\cdot)$.

3.2.1 The m-separation Criterion

Now let us extend Pearl's d-separation property [Pea88], defined originally for directed graphs, to the class of mixed graphs.

A non-endpoint vertex u on a path is a *follower on the path* if u is a follower for its predecessor or successor, or both of them on the path. A non-endpoint vertex u on a path is a *neighbor on the path* if u is a neighbor for its predecessor and successor on the path. A non-endpoint vertex u, which is neither a follower nor a neighbor on the path is *manager on the path*, *i.e.* the links at a manager u have one of the following form: $\rightarrow u\leftarrow$, $\rightarrow u-$, $-u\leftarrow$. These definitions are also listed in Table B.1 in Appendix B.

A path between vertices v_1 and v_n in a mixed graph $\mathcal{G}^m$ is said to be *m-connecting given a set* $\mathcal{U}$ (possibly empty), with $v_1,v_n\notin\mathcal{U}$, if

- **(a)** every follower on the path is not in $\mathcal{U}$;
- **(b)** every neighbor on the path is in $\mathcal{U}$;
- **(c)** every manager on the path is in $\overrightarrow{\mathrm{drv}}(\mathcal{U})$.

If there is no path m-connecting v_1 and v_n given $\mathcal{U}$, then v_1 and v_n are said to m-separated by $\mathcal{U}$. Sets $\mathcal{V}_1$ and $\mathcal{V}_2$ are *m-separated* given $\mathcal{U}$, if for every pair v_1, v_n with $v_1\in\mathcal{V}_1$ and $v_n\in\mathcal{V}_2$, v_1 and v_n are m-separated given $\mathcal{U}$ ($\mathcal{V}_1$, $\mathcal{V}_2$, $\mathcal{U}$ are disjoint sets; $\mathcal{V}_1$, $\mathcal{V}_2$ are non-empty). If $\mathcal{V}_1$ is m-separated from $\mathcal{V}_2$ by $\mathcal{U}$, the joint distribution over all variables in graph will satisfy conditional independence property $\mathcal{V}_1\perp\!\!\!\perp\mathcal{V}_2\mid\mathcal{U}$.

This is an extension of Pearl's d-separation criterion to mixed graphs in that in a DAG $\mathcal{D}$, a path is d-connected if and only if it is m-connecting. Here letter 'd' stays for 'directed' and 'm' – for 'mixed'.

3.2.1.1 Properties of m-connecting paths

We now prove two lemmas giving properties of m-connecting paths that we will exploit in Section 3.2.2. Here we follow the reasoning presented first for ancestral graphs in [RS02].

Lemma 3.1. *If* $\boldsymbol{\pi}$ *is a path m-connecting* v_1 *and* v_n *given* $\mathcal{U}$ *in a mixed graph* $\mathcal{G}^m$ *then every vertex on* $\boldsymbol{\pi}$ *is in* $\overrightarrow{\mathrm{drv}}(\mathcal{U}\cup\{v_1,v_n\})$.

Proof. Suppose that a vertex u, laying on the path $\boldsymbol{\pi}$ is not connected with v_1 or v_n via a directed path. In this case in order to prove the Theorem it is necessary and sufficient to show that $u \in \overrightarrow{\text{drv}}(\mathcal{U})$.

According to our assumption that u is not in $\overrightarrow{\text{drv}}(v_1)$ nor in $\overrightarrow{\text{drv}}(v_2)$, there must exist undirected edges or arcs pointing towards u on $\boldsymbol{\pi}$. Let vertexes $u_1 \in \boldsymbol{\pi}(v_1, u)$ and $u_n \in \boldsymbol{\pi}(u, v_n)$ be the closest vertexes to u, where such situations occur. Subpaths $\boldsymbol{\pi}(u_1, u)$ and $\boldsymbol{\pi}(u, u_n)$ contain no edges and no arcs pointing towards u, which means that they are directed paths from u to u_1 and u_n; and that vertexes u_1, u_n are also managers on $\boldsymbol{\pi}$. Since $\boldsymbol{\pi}$ is m-connecting given $\mathcal{U}$, both vertexes $u_1, u_n \in \overrightarrow{\text{drv}}(\mathcal{U})$, and subsequently $u \in \overrightarrow{\text{drv}}(\mathcal{U})$. □

Lemma 3.2. *Let $\mathcal{G}^m$ be a mixed graph containing disjoint sets of vertices $\mathcal{V}_1, \mathcal{V}_2$ and $\mathcal{U}$ ($\mathcal{U}$ may be empty). If there are vertices $v_1 \in \mathcal{V}_1$ and $v_n \in \mathcal{V}_2$ joined by a path $\boldsymbol{\pi}$ on which no followers is in $\mathcal{U}$, all neighbors are in $\mathcal{U}$, and every manager is in $\overrightarrow{\text{drv}}(\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{U})$ then there exist vertices $v_1' \in \mathcal{V}_1$, $v_n' \in \mathcal{V}_2$ such that v_1' and v_n' are m-connected given $\mathcal{U}$ in $\mathcal{G}^m$.*

Proof. Let $\boldsymbol{\pi}'$ be a path which contains the minimum number of managers of any path between some vertex $v_1' \in \mathcal{V}_1$ and some vertex $v_n' \in \mathcal{V}_2$ on which no followers but all neighbors are in $\mathcal{U}$ and every manager is in $\overrightarrow{\text{drv}}(\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{U})$. Path $\boldsymbol{\pi}'$ is guaranteed to exist since the path $\boldsymbol{\pi}$ described in the Lemma has this form. In order to show that $\boldsymbol{\pi}'$ m-connects v_1' and v_n' given $\mathcal{U}$ it is sufficient to show that every manager on $\boldsymbol{\pi}'$ is in $\overrightarrow{\text{drv}}(\mathcal{U})$.

Suppose for a contradiction that there is a manager u on $\boldsymbol{\pi}'$ and $u \notin \overrightarrow{\text{drv}}(\mathcal{U})$. Then $u \in \overrightarrow{\text{drv}}(\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{U}) \setminus \overrightarrow{\text{drv}}(\mathcal{U})$. So either $u \in \overrightarrow{\text{drv}}(\mathcal{V}_1) \setminus \overrightarrow{\text{drv}}(\mathcal{U})$ or $u \in \overrightarrow{\text{drv}}(\mathcal{V}_2) \setminus \overrightarrow{\text{drv}}(\mathcal{U})$. Suppose the former, then there is a directed path $\boldsymbol{\pi}''$ from u to some vertex $v_1'' \in \mathcal{V}_1$. Let u' be the vertex closest to v_n' on $\boldsymbol{\pi}'$ which is also on $\boldsymbol{\pi}''$. By construction the paths $\boldsymbol{\pi}'(u', v_n')$ and $\boldsymbol{\pi}''(u', v_1')$ do not intersect except at u'. Hence concatenating these subpaths forms a path which satisfies the conditions on $\boldsymbol{\pi}'$ but, since u' is a follower on the concatenated path, it has fewer managers than $\boldsymbol{\pi}'$, which is a contradiction. The case where $u \in \overrightarrow{\text{drv}}(\mathcal{V}_2) \setminus \overrightarrow{\text{drv}}(\mathcal{U})$ is symmetric. □

Corollary 3.1. *In a mixed graph $\mathcal{G}^m$, there is a path $\boldsymbol{\pi}$ between v_1 and v_n on which no follower and all neighbors are in $\mathcal{U}$ ($v_1, v_n \notin \mathcal{U}$) and every manager is in $\overrightarrow{\text{drv}}(\{v_1, v_n\} \cup \mathcal{U})$ if and only if there is a path m-connecting v_1 and v_n given $\mathcal{U}$ in $\mathcal{G}^m$.*

Proof. One direction is immediate and the other is a special case of Lemma 3.2 with $\mathcal{V}_1 = \{v_1\}, \mathcal{V}_2 = \{v_n\}$. □

This corollary shows that condition **(c)** in the definition of m-separation can be weakened to:

(c)' every manager on the path is in $\overrightarrow{\text{drv}}(\{v_1, v_2\} \cup \mathcal{U})$.

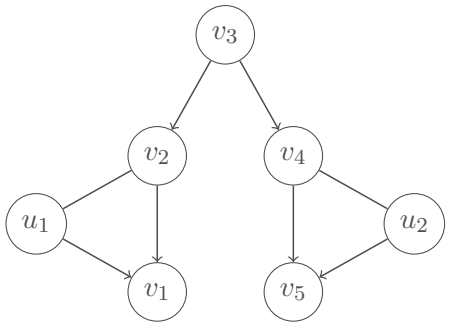


Fig. 3.4: Illustration of Lemma 3.2: a mixed graph $\mathcal{G}^m$, where every vertex is in $\overrightarrow{\text{drv}}(\{v_1, v_2\})$. Path $\boldsymbol{\pi}$, passing through vertices v_1, v_2, v_3, v_4 and v_5 m-connects v_1 and v_5 given $\emptyset$.

3.2.2 Marginalization and Conditioning

Independence Model. An independence model $\mathcal{J}$ over a set $\mathcal{V}$ is a set of triples $\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U}$ where $\mathcal{V}_1, \mathcal{V}_2$ and $\mathcal{U}$ are disjoint subsets of $\mathcal{V}$; $\mathcal{V}_1$ and $\mathcal{V}_2$ are non-empty. The triple $\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U}$ is interpreted as saying that $\mathcal{V}_1$ is independent of $\mathcal{V}_2$ given $\mathcal{U}$.

An independence model $\mathcal{J}$ with vertex set $\mathcal{V}$ after conditioning out a subset $\mathcal{Y}$, is simply the subset of triplets which do not involve any vertices in $\mathcal{Y}$. More formally we define:

$$\mathcal{J}\big[^{\mathcal{Y}} \equiv \Big\{(\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U}) : \ (\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U} \cup \mathcal{Y}) \in \mathcal{J};\ (\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{U}) \cap \mathcal{Y} = \emptyset\Big\}. \tag{3.1}$$

An independence model $\mathcal{J}$ with vertex set $\mathcal{V}$ after marginalizing out a subset $\mathcal{Z}$, is the set of triplets, defined as follows:

$$\mathcal{J}\big[_{\mathcal{Z}} \equiv \Big\{(\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U}) : \ (\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U}) \in \mathcal{J};\ (\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{U}) \cap \mathcal{Z} = \emptyset\Big\}. \tag{3.2}$$

Combining these two definitions, we obtain:

$$\mathcal{J}\big[^{\mathcal{Y}}_{\mathcal{Z}} \equiv \Big\{(\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U}) : \ (\mathcal{V}_1 \perp\!\!\!\perp \mathcal{V}_2 \mid \mathcal{U} \cup \mathcal{Y}) \in \mathcal{J};\ (\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{U}) \cap (\mathcal{Y} \cup \mathcal{Z}) = \emptyset\Big\}. \tag{3.3}$$

3.2.2.1 Graph Transformation

Graph $\mathcal{G}^m\big[^{\mathcal{Y}}_{\mathcal{Z}}$ has vertex set $\mathcal{V} \setminus (\mathcal{Y} \dot\cup \mathcal{Z})$ and edges specified as follows:
If two vertices v_1 and v_2 are conditionally dependent given $\mathcal{U} \cup \mathcal{Y}$, were $\mathcal{U} \subseteq \mathcal{V} \setminus (\mathcal{Y} \cup \mathcal{Z} \cup \{v_1, v_2\})$:

$$\{v_1\} \not\perp\!\!\!\perp \{v_2\} \mid \mathcal{U} \cup \mathcal{Y},$$

$$\textit{and} \left\{ \begin{array}{l} v_1 \in \mathrm{drv}(v_2) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y});\ v_2 \in \mathrm{drv}(v_1) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y}) \\ v_1 \in \mathrm{drv}(v_2) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y});\ v_2 \notin \mathrm{drv}(v_1) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y}) \\ v_1 \notin \mathrm{drv}(v_2) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y});\ v_2 \in \mathrm{drv}(v_1) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y}) \\ v_1 \notin \mathrm{drv}(v_2) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y});\ v_2 \notin \mathrm{drv}(v_1) \cup \overrightarrow{\mathrm{drv}}(\mathcal{Y}) \end{array} \right\} \textit{in } \mathcal{G}^m \textit{ then } \left\{ \begin{array}{l} v_1 - v_2 \\ v_1 \to v_2 \\ v_1 \leftarrow v_2 \\ v_1 - v_2 \end{array} \right\} \textit{in } \mathcal{G}^m\lceil_{\mathcal{Z}}^{\mathcal{Y}}.$$

In words, $\mathcal{G}^m\lceil_{\mathcal{Z}}^{\mathcal{Y}}$ is a graph containing the vertices that are not in $\mathcal{Y}$ or $\mathcal{Z}$. Two vertices v_1 and v_2 are adjacent in $\mathcal{G}^m\lceil_{\mathcal{Z}}^{\mathcal{Y}}$ if they are m-connected in $\mathcal{G}^m$ given any subset that contains all vertices in $\mathcal{Y}$ and no vertices in $\mathcal{Z}$. If vertices v_1 and v_2 are adjacent in $\mathcal{G}\lceil_{\mathcal{Z}}^{\mathcal{Y}}$ then they are linked with an edge if they are driven by each other or both lie on the directed paths to some vertices in $\mathcal{Y}$; or if these two conditions are violated. Otherwise they are linked with an arc, pointing towards v_1 if and only if v_1 is a leading vertex for v_2, or v_2 is linked via a directed path with a vertex from $\mathcal{Y}$, whereas v_1 is not; and an arc pointing towards v_2 otherwise.

A path $\boldsymbol{\pi}$ between vertices v_1 and v_n on which every follower is in $\mathcal{Z}$, every neighbor is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{v_1, v_n\}) \setminus \mathcal{Z}$ and every manager is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{v_1, v_n\})$, is called an *inducing path with respect to* $\mathcal{Y}$ *and* $\mathcal{Z}$. This is a generalization of the definition introduced in [VP91].

Theorem 3.1. *If* $\mathcal{G}^m = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ *is an mixed graph, and* $\mathcal{Y} \dot\cup \mathcal{Z} \subset \mathcal{V}$ *then*

$$\mathcal{J}(\mathcal{G})\lceil_{\mathcal{Z}}^{\mathcal{Y}} = \mathcal{J}(\mathcal{G}\lceil_{\mathcal{Z}}^{\mathcal{Y}}).$$

The proof of this theorem for mixed graphical models may be found in [RS02].

3.3 Multi-Layer Graphical Models

In Section 3.2 we have defined the mixed graph $\mathcal{G}^m = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ as a structure, build upon nodes $\mathcal{V}$, undirected edges $\mathcal{E}$ and directed arks $\mathcal{A}$. In probabilistic graphical models, graph nodes represent random variables while edges and arcs designate conditional dependencies between them. In contrast to the classical CRFs, where no arcs were supported, in the multi-layer approach, directed arcs designate 'one-way', casual dependencies. The class of mixed graphs is much larger than required for our purposes, hence we introduce now the subclass of multi-layer graphs.

The *multi-layer* graph $\mathcal{G}^{ml} = (\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \mathcal{E}, \mathcal{A})$ is a mixed graph $\mathcal{G}^m = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ in which the array of nodes consists of the data nodes $y \in \mathcal{Y}$, base layer class nodes $x \in \mathcal{X}$ and occlusion layers class nodes $z \in \mathcal{Z}$: $\mathcal{V} = \mathcal{X} \dot\cup \mathcal{Y} \dot\cup \mathcal{Z}$; and the following conditions hold for all vertexes y, x and z in $\mathcal{G}^{ml}$:

(a) $\forall v \in \mathcal{V},\ \exists y \in \mathcal{Y} :\ v \in \overrightarrow{\mathrm{drv}}(y)$;

(b) $\forall v \in \mathcal{V},\ \forall y \in \mathcal{Y} :\ v \in \mathrm{drv}(y)$;

In words, condition **(a)** states that every vertex in a multi-layer graph is linked via directed path with at least one data vertex, *i.e.* every latent random variable, represented with multi-layer graph and no matter from which layer, is conditioned on corresponding observation. Condition **(b)** states that every graph vertex is driven by every data vertex, what guaranties that the information from all observations affects every single latent random variable. In addition, according to our occlusion model, we condition the base and the top occlusion layers directly on the data, *i.e.* $\mathcal{X} \subset \overrightarrow{\text{drv}}(\mathcal{Y})$, $\mathcal{Z} \supseteq \mathbf{z}^1 \subset \overrightarrow{\text{drv}}(\mathcal{Y})$. This is the motivation for terming such graphs 'multi-layer'.

From this definition directly follow two important properties of multi-layer graphs:

Proposition 3.5. *In a multi-layer graph all the vertexes are driven by every data site:*

$$\mathcal{V} \subseteq \text{drv}(y),\ \forall y \in \mathcal{Y}.$$

Proposition 3.6. *In a multi-layer graph all vertexes $x \in \mathcal{X}$ are not neighbors.*

In order to keep our model computationally light, we also restrict the number of links in an inducing path to 2.

Theorem 3.2. *If $\mathcal{G}^m = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ is a multi-layer graph, with vertex set $\mathcal{V} = \mathcal{X} \dot\cup \mathcal{Y} \dot\cup \mathcal{Z}$, and $x_1, x_2 \in \mathcal{X}$ then the following five conditions are equivalent:*

1. *There is an edge or arc between x_1 and x_2 in $\mathcal{G}^m \lceil^{\mathcal{Y}}_{\mathcal{Z}}$.*
2. *There is an inducing path between x_1 and x_2 with respect to $\mathcal{Y}$ and $\mathcal{Z}$ in $\mathcal{G}^m$.*
3. *The verticies in $\text{drv}(\mathcal{Y} \cup \{x_1, x_2\})$ that are not in $\mathcal{Z} \cup \{x_1, x_2\}$ do not m-separate x_1 and x_2 in $\mathcal{G}^m$:*

$$\{x_1\} \not\perp\!\!\!\perp \{x_2\} \mid \text{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus (\mathcal{Z} \cup \{x_1, x_2\}).$$

4. $\forall \mathcal{U},\ \mathcal{U} \subseteq \mathcal{V} \setminus (\mathcal{Y} \cup \mathcal{Z} \cup \{x_1, x_2\}),\ (\{x_1\} \not\perp\!\!\!\perp \{x_2\} \mid \mathcal{U} \cup \mathcal{Y})$ *in* $\mathcal{J}(\mathcal{G}^m)$.
5. $\forall \mathcal{U},\ \mathcal{U} \subseteq \mathcal{V} \setminus (\mathcal{Y} \cup \mathcal{Z} \cup \{x_1, x_2\}),\ (\{x_1\} \not\perp\!\!\!\perp \{x_2\} \mid \mathcal{U})$ *in* $\mathcal{J}(\mathcal{G}^m) \lceil^{\mathcal{Y}}_{\mathcal{Z}}$.

Proof. In order to simplify the proof, let us designate through $\mathcal{U}'$ the right-hand side of the expression in (3):

$$\mathcal{U}' = \text{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus (\mathcal{Z} \cup \{x_1, x_2\}).$$

By Proposition 3.4:

$$\begin{aligned}\overrightarrow{\mathrm{drv}}(\mathcal{U}' \cup \{x_1, x_2\}) &= \overrightarrow{\mathrm{drv}}((\mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus (\mathcal{Z} \cup \{x_1, x_2\})) \cup \{x_1, x_2\}) \\ &= \overrightarrow{\mathrm{drv}}(\mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{Z}) \\ &= \overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}).\end{aligned} \tag{3.4}$$

Now let us proceed to the proof point by point:

(2)⇒(3) If there is an inducing path $\boldsymbol{\pi}$ in $\mathcal{G}^m$ with respect to $\mathcal{Y}$ and $\mathcal{Z}$, then

- every follower on $\boldsymbol{\pi}$ is in $\mathcal{Z}$, and since $\mathcal{U}' \cap \mathcal{Z} = \emptyset$ none of the followers is in $\mathcal{U}'$;
- every neighbor on $\boldsymbol{\pi}$ is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{Z} \subseteq \mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{Z}$ by Proposition 3.1. Since $x_1, x_2 \in \mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\})$ and $x_1, x_2 \notin \mathcal{Z}$ we can write that $\mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{Z} = \mathcal{U}' \cup \{x_1, x_2\}$ and consequently every neighbor on $\boldsymbol{\pi}$ is in $\mathcal{U}'$; and
- every manager on $\boldsymbol{\pi}$ is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) = \overrightarrow{\mathrm{drv}}(\mathcal{U}' \cup \{x_1, x_2\})$ by Equation 3.4.

Hence by Corollary 3.1 there exists a path $\boldsymbol{\pi}'$ which m-connects x_1 and x_2 given $\mathcal{U}'$ in $\mathcal{G}^m$, which means that $\{x_1\} \not\perp\!\!\!\perp \{x_2\} \mid \mathcal{U}'$.

(3)⇒(2) Let $\boldsymbol{\pi}$ be a path which m-connects x_1 and x_2 given $\mathcal{U}'$. By Lemma 3.1 and Equation 3.4, every vertex on $\boldsymbol{\pi}$ is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\})$, thus:

- every follower is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{U}' \subseteq \mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{U}' = \mathcal{Z} \cup \{x_1, x_2\}$ by Proposition 3.1, so every follower is in $\mathcal{Z}$;
- every neighbor is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \cap \mathcal{U}' = \overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus (\mathcal{Z} \cup \{x_1, x_2\}) \subseteq \overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{Z}$ by Proposition 3.1; and
- every manager is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \cap \overrightarrow{\mathrm{drv}}(\mathcal{U}') \subseteq \overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\})$.

Hence $\boldsymbol{\pi}$ is an inducing path with respect to $\mathcal{Y}$ and $\mathcal{Z}$ in $\mathcal{G}^m$.

(3)⇒(4) Again, by Lemma 3.1 and Equation 3.4, every vertex on $\boldsymbol{\pi}$ is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\})$, thus:

- every follower is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{U}' \subseteq \mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus \mathcal{U}' = \mathcal{Z} \cup \{x_1, x_2\}$ by Proposition 3.1. Since $(\mathcal{Z} \cup \{x_1, x_2\}) \cap (\mathcal{U} \cup \mathcal{Y}) = \emptyset$, every follower is not in $\mathcal{U}$ or $\mathcal{Y}$;
- every neighbor is in $\overrightarrow{\mathrm{drv}}(\mathcal{Y} \cup \{x_1, x_2\}) \cap \mathcal{U}' \subseteq \mathrm{drv}(\mathcal{Y} \cup \{x_1, x_2\}) \setminus (\mathcal{Z} \cup \{x_1, x_2\}) = \mathcal{V} \setminus (\mathcal{Z} \cup \{x_1, x_2\})$ by Propositions 3.1 and 3.5. Thus, for every possible choice of $\mathcal{U}$ the union $\mathcal{U} \cup \mathcal{Y}$ will lie inside $\mathcal{V} \setminus (\mathcal{Z} \cup \{x_1, x_2\})$; and

- every manager is in $\overrightarrow{\text{drv}}(\mathcal{Y}\cup\{x_1,x_2\})\cap\overrightarrow{\text{drv}}(\mathcal{U}')\subseteq\overrightarrow{\text{drv}}(\text{drv}(\mathcal{Y}\cup\{x_1,x_2\})\setminus(\mathcal{Z}\cup\{x_1,x_2\}))=\overrightarrow{\text{drv}}(\mathcal{V}\setminus(\mathcal{Z}\cup\{x_1,x_2\}))$ by Proposition 3.5. As we have shown above $(\mathcal{U}\cup\mathcal{Y})\subseteq\mathcal{V}\setminus(\mathcal{Z}\cup\{x_1,x_2\})$, by Propsition 3.3 follows that for every choice of $\mathcal{U}$ holds $\overrightarrow{\text{drv}}(\mathcal{U}\cup\mathcal{Y})\subseteq\overrightarrow{\text{drv}}(\mathcal{V}\setminus(\mathcal{Z}\cup\{x_1,x_2\}))$.

Hence, if none of the vertices from $\mathcal{V}\setminus(\mathcal{Z}\cup\{x_1,x_2\})$ separate x_1 and x_2, then there is no such $\mathcal{U}\subseteq\mathcal{V}\setminus(\mathcal{Y}\cup\mathcal{Z}\cup\{x_1,x_2\})$ that fulfills $(\{x_1\}\perp\!\!\!\perp\{x_2\}\mid\mathcal{U}\cup\mathcal{Y})$ in $\mathcal{J}(\mathcal{G}^m)$.

(4)⇒(3) This follows trivially taking $\mathcal{U}=\mathcal{U}'\setminus\mathcal{Y}$.

(4)⇔(1) Definition of $\mathcal{G}^m\lfloor_{\mathcal{Z}}^{\mathcal{Y}}$.

(4)⇔(5) Definition of $\mathcal{J}(\mathcal{G}^m)\lfloor_{\mathcal{Z}}^{\mathcal{Y}}$.

□

An important consequence of condition (3) in Theorem 3.2 is that a single test of m-separation in $\mathcal{G}^m$ is sufficient to determine whether or not a given adjacency is present in $\mathcal{G}^m\lfloor_{\mathcal{Z}}^{\mathcal{Y}}$; it is not necessary to test every subset of $\mathcal{V}\setminus(\mathcal{Y}\cup\mathcal{Z}\cup\{x_1,x_2\})$.

3.3.1 Multi-Layer CRFs

Multi-layer conditional random fields are based on classical conditional random fields, described in Chapter 2 with one main difference: rather than having one class variable x_i per image site y_i, now we determine $r+1$ such class variables: x_i and $z_i^1,\ldots,z_i^r$ (thus r is the number of occlusion layers), corresponding to the base and occlusion layers of our occlusion model, respectively. Consequently, the multiple class variables form a *multi-layer* conditional random field [KSG18b].

According to our reasoning from Section 3.1, we split the label set into two subsets $\mathbb{L}=\mathbb{L}^x\dot\cup\mathbb{L}^z$, where $\mathbb{L}^x$ corresponds to objects at the base layer and $\mathbb{L}^z$ corresponds to objects at the occlusion layer, hence x_i takes values $l\in\mathbb{L}^x$ and $z_i^k,\ k\in[1;r]$ – values $l\in\mathbb{L}^z\cup\{\mathit{void}\}$. Sets $\mathbb{L}^x$ and $\mathbb{L}^z$ are mutually exclusive and the special class *void* is added in order to model situations where the base layer is not occluded (refer to Figure 3.2).

More formally, we address the general problem of learning a mapping from input observations $y\in\mathbb{Y}$ to a number of discrete response variables $(x,z^1,\ldots,z^r)^\top\in\mathbb{X}^{r+1}$, based on a training sample of input-output pairs $((x_1,z_1^1)^\top,y_1),\ \ldots,\ ((x_n,z_n^1)^\top,y_n)\in\mathbb{X}^2\times\mathbb{Y}$ drawn from some fixed but unknown probability distribution. We assume an input image $\mathbf{y}$ to consist of n data sites $i\in\mathcal{Y}=\{1,2,\ldots,n\}$ with observed data y_i, *i.e.*, $\mathbf{y}=(y_1,y_2,\ldots,y_n)^\top$, where $\mathcal{Y}$ is the array of all sites, corresponding to the data nodes of an associated graph $\mathcal{G}^{ml}$. With each

site i we associate $r+1$ discrete class variables $(x_i, z_i^1, \dots, z_i^r)^\top \in \mathbb{X}^{r+1}$, which take values from a given sets of classes $\mathbb{L}^x \dot\cup \mathbb{L}^z \cup \{void\}$.

The goal of classification is to determine the most probable values for $x_i, z_i^1, \dots, z_i^r$ given the data $\mathbf{y}$. Thus, our aim is to compute the posterior probability $p(\mathbf{x}, \mathbf{z}^1, \dots, \mathbf{z}^r \,|\, \mathbf{y})$ of a possible output $\mathbf{x} = ((x_1, z_1^1, \dots, z_1^r)^\top, \dots, (x_n, z_n^1, \dots, z_n^r)^\top)^\top \in \mathbb{X}^{(r+1)\cdot n}$ given the input $\mathbf{y} = (y_1, \dots, y_n)^\top \in \mathbb{Y}^n$. Finally, according to [KSG18b] we model the posterior probability $p(\mathbf{x}, \mathbf{z}^1, \dots, \mathbf{z}^r \,|\, \mathbf{y})$ directly, expanding the model in Equation 2.5:

$$p(\mathbf{x}, \mathbf{z}^1, \dots, \mathbf{z}^r \,|\, \mathbf{y}) = \frac{1}{Z} \cdot P_{data} \cdot P_{smooth} \cdot P_{occl}. \tag{3.5}$$

Here, the data and smoothness terms correspond to the CRF model, given in Equation 2.5; P_{occl} is the new occlusion term, which brings the causal properties to the model and correspond to the directed graph arcs $\mathcal{A}$. Our definitions of these terms and underlying potential functions are described below.

Figure 3.5 shows the structure of our multi-layer graphical model. Squares and circles correspond to observations and labels, respectively. The dark blue nodes (data sites 2, 3, 4 and 6) correspond to the regions with occlusion, *i.e.* where only the occluding object is visible; and thus, the dark red nodes – to the labels, corresponding to actually visible objects. The graph edges represent dependencies between the nodes.

The reason why we have split the layers is to increase the accuracy of the labeling of occluded regions: to reveal the labels of the bright label nodes, where the association potentials could not provide the corresponding base layer nodes with reliable information because the data is not observable.

Data term. The data term has three components:

$$P_{data} = \prod_{i \in \mathcal{Y}} \varphi_i^x(x_i; \mathbf{y}) \varphi^z(z_i^1; \mathbf{y}) \prod_{k=2}^{r} \varphi_i^a(z_i^k), \tag{3.6}$$

where the association potentials $\varphi_i^x \in \mathbb{R}^{|\mathbb{L}^x|}$ and $\varphi_i^z \in \mathbb{R}^{|\mathbb{L}_+^z|}$[2] associate the data $\mathbf{y}$ with the class variables x_i, z_i^1 of image site i (blue links in Figure 3.5). Omitting the superscript indicating the layer, the association potentials φ_i are related to the probability of a label x_i (or z_i^1) taking a value l given the data $\mathbf{y}$ by $\varphi_i(x_i; \mathbf{y}) = p(x_i{=}l \,|\, \mathbf{f}_i(\mathbf{y}))$, where the image data are represented by site-wise feature vectors $\mathbf{f}_i(\mathbf{y})$ that may depend on all observations $\mathbf{y}$.

The auxiliary data-independent potential $\varphi^a \in \mathbb{R}^{|\mathbb{L}_+^z|}$ is used to initialize the variables $z_i^k, k \in [2; r]$ with a uniformly distributed constant values: $\varphi^a(z_i^k) \equiv 1/|\mathbb{L}^z|$. That makes these

[2] We designate $\mathbb{L}_+^z \equiv \mathbb{L}^z \cup \{void\}$.

variables to agree with the corresponding z_i^1 variables after inference. We will describe the role of these variables in Section 3.4.1.

Smoothness term. The smoothness term is the product of interaction pairwise potentials over neighboring pixels, defined by graph edges $\mathcal{E}$ (red links in Figure 3.5). In order to preserve the geometrical boundaries between classes and prolongate these boundaries into the occluded regions we split the set of edges $\mathcal{E}$ into a number of non-overlapping edge groups $\mathcal{E}_g, g \in \mathcal{G}$. According to *contour completion* CRF [Sil+14] we can write the smoothness term from Equation 2.5 in form:

$$\prod_{(i,j)\in\mathcal{E}} \psi_{ij}(x_i, x_j; \mathbf{y}) \equiv \prod_{g\in\mathcal{G}} \prod_{(i,j)\in\mathcal{E}_g} \psi_{ij}^g(x_i, x_j; \mathbf{y}). \tag{3.7}$$

Initially, we assign all graph edges to one group g=0. We use Hough transform [DH72] to detect objects' boundaries in the scene, specifically we detect lines and 2^{nd}-order curves. For each detected line and curve we add one group $g \in \mathcal{G}$ and define a group-specific class-transition matrix $\mathcal{T}_g$ ($\mathcal{T}_0$ is a unit matrix). All graph edges, intersecting the line we re-assign to the group $\mathcal{E}_g$. Finally, we can write down our smoothness term with group-specific interaction potentials:

$$P_{smooth} = \prod_{i\in\mathcal{Y}} \prod_{g\in\mathcal{G}} \prod_{(i,j)\in\mathcal{E}_g} \psi_{ij}^g(x_i, x_j; \mathbf{y}) \prod_{k=1}^{r} \psi_{ij}^g(z_i^k, z_j^k; \mathbf{y}), \tag{3.8}$$

here the potentials $\psi_{ij}^g \in \mathbb{R}^{|\mathbb{L}|\times|\mathbb{L}|}$ describe how likely a pair of adjacent sites i and j is to take the labels $(x_i, x_j) = (l, l')$ given the data $\mathbf{y}$: $\psi_{ij}^g(x_i, x_j; \mathbf{y}) = p(x_i{=}l, x_j{=}l' \,|\, \mathbf{f}_i(\mathbf{y}), \mathbf{f}_j(\mathbf{y}), g)$. Since $\mathbb{L}^x \cap \mathbb{L}_+^z = \emptyset$ we may use the same potential ψ_{ij}^g at all layers. The main difference between the model in Equation 3.8 and [Sil+14] is that the we use non-overlapping groups $\mathcal{E}_g$ and group-specific potentials ψ_{ij}^g.

In order to model our interaction potentials, we first initialize all matrices $\mathcal{T}_g(l, l')$ with a unit matrix and add them to the contrast-sensitive data-dependent model, defined in Equation 2.47:

$$p(x_i{=}l, x_j{=}l' \,|\, \mathbf{y}, g) = \begin{cases} \theta_l \cdot \mathcal{P}(d_{i,j}) \cdot h(l, l') \text{ if } l{=}l' \\ \mathcal{T}_g(l, l') \cdot h(l, l') \text{ if } l{\neq}l' \end{cases} \tag{3.9}$$

where the regularization function $\mathcal{P}(d_{i,j})$ is defined in Equation 2.46. Matrices $\mathcal{T}_g$ bring specific class transition behavior for sites, connected with edges, which belong to groups $\mathcal{E}_g$. We consider this on more detail in Section 3.4.1. This model differs from the contrast-sensitive data-dependent model from Section 2.4.2 by the use of the prior probabilities $\mathcal{P}$ and class-transition matrices $\mathcal{T}_g$. As a consequence, class transitions become more likely, depending on the frequency with which they occur in the training data and belonging to a specific edge

group.

Occlusion term. The occlusion term consists of two *inter-layer pairwise potentials*, which bind all layers of our model:

$$P_{occl} = \prod_{i \in \mathcal{Y}} \prod_{(i,t) \in \mathcal{A}} \xi_i^x(x_i, z_i^1; \mathbf{y}) \prod_{k=2}^{r} \xi_i^z(z_i^{k-1}, z_t^k). \tag{3.10}$$

The first component $\xi_i^x \in \mathbb{R}^{|\mathbb{L}^x| \times |\mathbb{L}_+^z|}$ is the innovative *occlusion* potentials, which is the key to successful layer decomposition (purple links in Figure 3.6). It represents a measure of how likely one object at visible layer occludes other object from base layer and binds random variables x_i and z_i^1 from different domains and consequently the potentials are represented by non-square and non-symmetric matrices of $|\mathbb{L}^x| \times |\mathbb{L}_+^z|$ entries. They learn how likely a base class variable x_i is to take value $l \in \mathbb{L}^x$ given the data $\mathbf{y}$ and being 'occluded' by an occlusion class variable z_i^1, which takes value $l' \in \mathbb{L}_+^z$: $\xi_i^x(x_i, z_i^1; \mathbf{y}) = p(x_i{=}l \,|\, z_i^1{=}l', \mathbf{f}_i(\mathbf{y}))$. Since $\mathbb{L}^x \cap \mathbb{L}_+^z = \emptyset$ we can not model the occlusion potential using classical CRF techniques. In Section 3.3.2 we overcome this problem by proposing a novel model for these pairwise potentials.

The second component $\xi_i^z \in \mathbb{R}^{|\mathbb{L}_+^z| \times |\mathbb{L}_+^z|}$ is the *inter-layer smoothness potentials* corresponding to the directed graph arcs (green links in Figures 3.5 and 3.6), which connect occlusion variables z_i^k at neighboring layers. Since these variables take values from the set $\mathbb{L}_+^z$, we can model potentials ξ_i^z with a data-independent model from Section 2.4.1. This component allows for directing the reliable information from the visible layer into the occlusion area and prevents any ambiguity to leave it. The inter-layer smoothness potentials are also described in more details in Section 3.3.2.

We derive the parametric version of Equations 3.5, 3.6, 3.8 and 3.10 as expansion of the model in Equation 2.7:

$$\begin{aligned} p(\mathbf{x}, \mathbf{z}^1, \ldots, \mathbf{z}^r \,|\, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z} & \prod_{i \in \mathcal{Y}} \left\langle \varphi_i^x(x_i; \mathbf{y}), \boldsymbol{\theta}_\varphi^x \right\rangle \left\langle \varphi_i^z(z_i^1; \mathbf{y}), \boldsymbol{\theta}_\varphi^z \right\rangle \prod_{k=2}^{r} \left\langle \varphi_i^a(z_i^k), \boldsymbol{\theta}_\varphi^a \right\rangle \cdot \\ & \prod_{i \in \mathcal{Y}} \prod_{(i,j) \in \mathcal{E}} \left\langle \psi_{ij}^x(x_i, x_j; \mathbf{y}), \boldsymbol{\theta}_\psi^x \right\rangle \prod_{k=1}^{r} \left\langle \psi_{ij}^z(z_i^k, z_j^k; \mathbf{y}), \boldsymbol{\theta}_\psi^z \right\rangle \cdot \\ & \prod_{i \in \mathcal{Y}} \left\langle \xi_i^x(x_i, z_i^1; \mathbf{y}), \boldsymbol{\theta}_\xi^x \right\rangle \prod_{k=2}^{r} \left\langle \xi_i^z(z_i^{k-1}, z_i^k), \boldsymbol{\theta}_\xi^z \right\rangle . \end{aligned} \tag{3.11}$$

In Equation 3.11, $\boldsymbol{\theta} = \left\{ \boldsymbol{\theta}_\varphi^x, \boldsymbol{\theta}_\varphi^z, \boldsymbol{\theta}_\varphi^a, \boldsymbol{\theta}_\psi^x, \boldsymbol{\theta}_\psi^z, \boldsymbol{\theta}_\xi^x, \boldsymbol{\theta}_\xi^z \right\}$ are model control parameters, including weights, which modulate the influence of the individual terms in the classification and those,

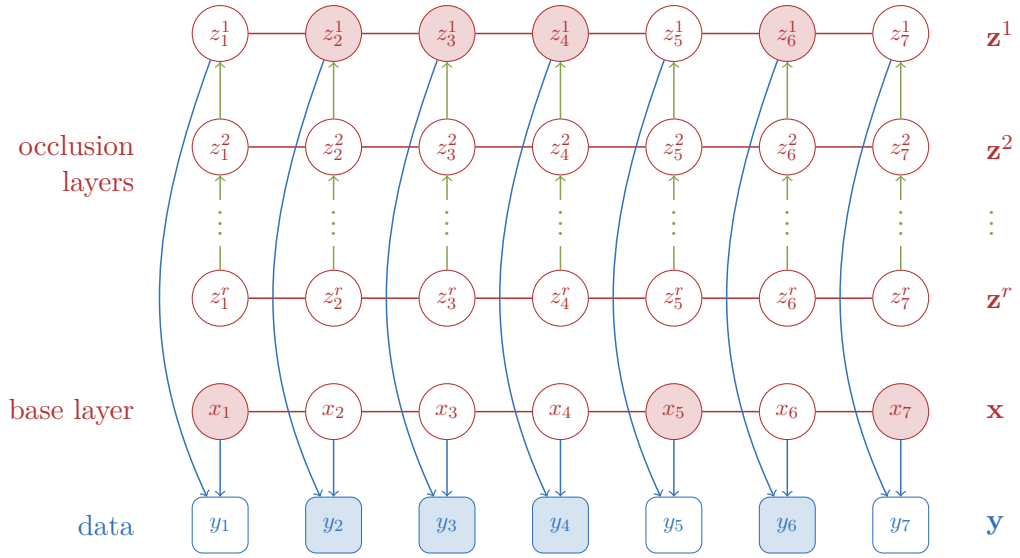


Fig. 3.5: Structure of the multi-layer graph $\mathcal{G}^{ml}$. Squares and circles correspond to observations $\mathbf{y}$ and labels $\mathbf{x}, \mathbf{z}^1, \dots \mathbf{z}^r$, respectively; dark blue nodes represent occluded regions, corresponding to the Figure 3.2, while dark red nodes – to the actually observed objects. The second dimension and additional links between data and labels are omitted for simplicity.

which are included into the potential functions.

Figure 3.6 shows the structure of our multi-layer CRF model, which corresponds to the graph $\mathcal{G}^{ml}$, depicted at Figure 3.5 after conditioning on the data $\mathcal{Y}$. At the figure, we designate the association potentials with blue color, within-layer pairwise potentials with red color, inter-layer pairwise potentials with the green color and occlusion potentials – with purple color. The reason why we have split the layers is to increase the accuracy of classification for occluded regions, *e.g.* to reveal the labels x_2, x_3, x_4 and x_6, where the association potentials could not provide the corresponding base layer nodes with reliable information because the data corresponding to the base layer are not observable.

Theorem 3.3. *If $\mathcal{G}^m = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ is a multi-layer graph, and $\mathcal{Y}, \mathcal{Z}$ are disjoint subsets of $\mathcal{V}$, then*

$$(\mathcal{G}^m \lceil^{\mathcal{Y}}) \lfloor_{\mathcal{Z}} = \mathcal{G}^m \lceil^{\mathcal{Y}}_{\mathcal{Z}}.$$

Proof. We prove this theorem in 3 steps: 1. we show that the set of vertexes in the left-hand side and the right-hand side of the theorem's statement are the same; 2. then we prove that the adjacencies are the same; 3. and finally – that the type of adjacencies (edges and arcs) are the same.

1. From the definition of the graph transformation operator in Section 3.2.2.1 and from the properties of the sets [3] directly follows that the graphs $(\mathcal{G}^m \lceil^{\mathcal{Y}}) \lfloor_{\mathcal{Z}} \equiv (\mathcal{G}^m \lceil^{\mathcal{Y}}_{\emptyset}) \lfloor^{\emptyset}_{\mathcal{Z}}$ and $\mathcal{G}^m \lceil^{\mathcal{Y}}_{\mathcal{Z}}$ have

[3] $\mathcal{V} \setminus \emptyset = \mathcal{V}$

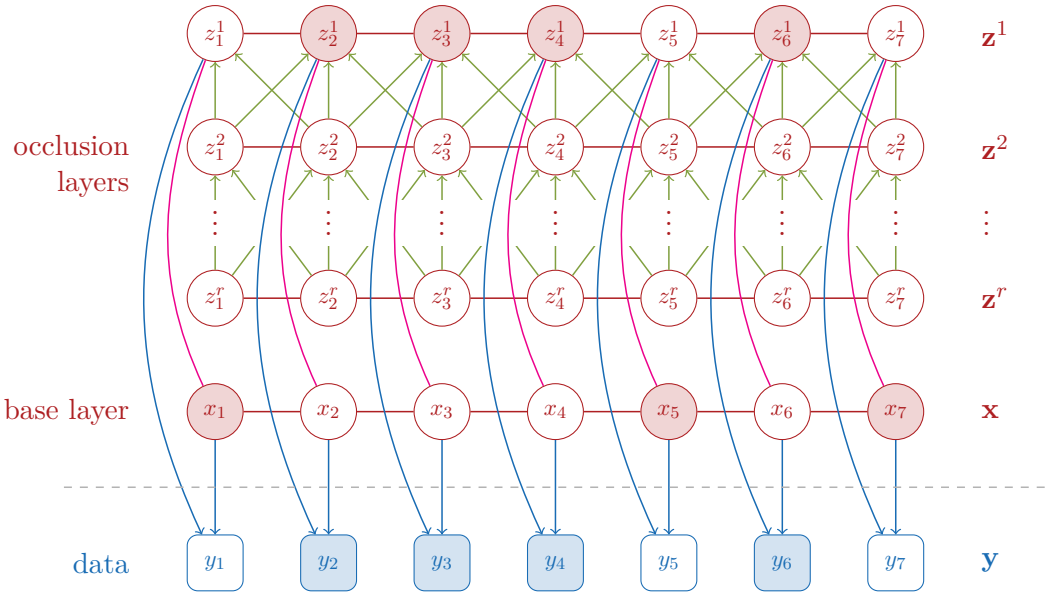


Fig. 3.6: Graph $\mathcal{G}^{ml}[^{\{\mathcal{Y}\}}$: structure of the multi-layer CRF model. Squares and circles correspond to observations $\mathbf{y}$ and labels $\mathbf{x}, \mathbf{z}^1, \dots \mathbf{z}^r$, respectively; blue edges - unary (association) potentials; red and green edges - within-layer and inter-layer pairwise (interaction) potentials respectively; purple edges - occlusion potentials. In spite of the data nodes $\mathbf{y}$ are not included in the graph $\mathcal{G}^{ml}[^{\{\mathcal{Y}\}}$, they are still present at the Figure in order to visualize the dependencies, corresponding to the unary potentials; dark blue nodes represent occluded regions, corresponding to the Figure 3.2, while dark red nodes - to the actually observed objects. The second dimension and additional links between data and labels are omitted for simplicity.

the same sets of vertexes.

2. Let x_1 and x_2 be two vertexes in $\mathcal{V} \setminus (\mathcal{Y} \cup \mathcal{Z})$. There is an edge or arc between x_1 and x_2 in $\mathcal{G}^m[^{\mathcal{Y}}_{\mathcal{Z}}$ if and only if $\forall \mathcal{U} \subseteq \mathcal{V} \setminus (\mathcal{Y} \cup \mathcal{Z} \cup \{x_1, x_2\}) : (x_1 \perp\!\!\!\perp x_2 \mid \mathcal{U} \cup \mathcal{Y}) \notin \mathcal{J}(\mathcal{G}^m)$ [4]. Using definition of independence model in Equation 3.3 and Theorem 3.1 we can rewrite this condition in form: $\forall \mathcal{U} \subseteq (\mathcal{V} \setminus \mathcal{Y}) \setminus (\mathcal{Z} \cup \{x_1, x_2\}) : (x_1 \perp\!\!\!\perp x_2 \mid \mathcal{U}) \notin \mathcal{J}(\mathcal{G}^m)[^{\mathcal{Y}} \overset{Thm.\ 3.1}{\equiv} \mathcal{J}(\mathcal{G}^m[^{\mathcal{Y}})$. From the latter follows that the vertexes x_1 and x_2 are adjacent in $(\mathcal{G}^m[^{\mathcal{Y}})[_{\mathcal{Z}}$.

3. Here we assume that vertexes $x_1, x_2 \in \mathcal{V} \setminus (\mathcal{Y} \cup \mathcal{Z})$ are adjacent in both $(\mathcal{G}^m[^{\mathcal{Y}})[_{\mathcal{Z}}$ and $\mathcal{G}^m[^{\mathcal{Y}}_{\mathcal{Z}}$. Let $x_1 \in \mathrm{drv}(x_2)$ in $\mathcal{G}^m[^{\mathcal{Y}}_{\mathcal{Z}}$, what means that $x_1 \in \mathrm{drv}(x_2 \cup \mathcal{Y})$ in $\mathcal{G}^m$. If we perform conditioning out $\mathcal{Y}$ from graph $\mathcal{G}^m$ we can write that x_1 is either in $\mathrm{drv}(x_2)$ or x_1 is a neighbor in $\mathcal{G}^m[^{\mathcal{Y}}$. If we subsequently marginalize out $\mathcal{Z}$ from graph $\mathcal{G}^m[^{\mathcal{Y}}$ we can still write that x_1 is either in $\mathrm{drv}(x_2)$ or x_1 is a neighbor but in $(\mathcal{G}^m[^{\mathcal{Y}})[_{\mathcal{Z}}$. Since we assumed that x_1 and x_2 are also adjacent in $(\mathcal{G}^m[^{\mathcal{Y}})[_{\mathcal{Z}}$, x_1 will not be a neighbor by Proposition 3.6 and thus $x_1 \in \mathrm{drv}(x_2)$ in $(\mathcal{G}^m[^{\mathcal{Y}})[_{\mathcal{Z}}$.

We have shown that if $x_1 \in \mathrm{drv}(x_2)$ in $\mathcal{G}^m[^{\mathcal{Y}}_{\mathcal{Z}}$ then $x_1 \in \mathrm{drv}(x_2)$ in $(\mathcal{G}^m[^{\mathcal{Y}})[_{\mathcal{Z}}$. Now in order to

[4] A paraphrasing of the definition of graph transformation in Section 3.2.2.1

finish the proof we need to show that if $x_1 \notin \mathrm{drv}(x_2)$ in $\mathcal{G}^m\lceil_{\mathcal{Z}}^{\mathcal{Y}}$ then $x_1 \notin \mathrm{drv}(x_2)$ in $(\mathcal{G}^m\lceil^{\mathcal{Y}})\lfloor_{\mathcal{Z}}$. If $x_1 \notin \mathrm{drv}(x_2)$ in $\mathcal{G}^m\lceil_{\mathcal{Z}}^{\mathcal{Y}}$ then $x_1 \notin \mathrm{drv}(x_2)$ or $x_1 \in \mathcal{Y}$ in $\mathcal{G}^m$ by Proposition 3.5. By our assumption that $x_1 \in \mathcal{V} \setminus (\mathcal{Y} \cup \mathcal{Z})$ and by subsequent application of the conditioning and marginalization operation to graph $\mathcal{G}^m$ in the same way as in the first part of the proof, we derive that $x_1 \notin \mathrm{drv}(x_2)$ in $(\mathcal{G}^m\lceil^{\mathcal{Y}})\lfloor_{\mathcal{Z}}$.

□

3.3.2 Inter-Layer Pairwise Potential

The inter-layer pairwise potential $\xi_i^x(x_i, z_i^1; \mathbf{y})$ in Equation 3.10 describes how likely class variables x_i and z_i from base and occlusion layers and corresponding to the same data site i are to take values l and l' given the data $\mathbf{y}$, while the inter-layer pairwise potential $\xi_i^z(z_i^k, z_i^{k-1}; \mathbf{y})$ in Equation 3.10 describes how likely class variables z_i^k and z_i^{k-1}, $k \in [2; r]$ from occlusion layers and corresponding to the same data site i are to take values l' and l'' given the data $\mathbf{y}$:

$$\xi_i^x(x_i, z_i^1, \mathbf{y}) \propto p(x_i = l, z_i^1 = l' \,|\, \mathbf{f}_i(\mathbf{y})) \tag{3.12}$$

$$\xi_i^z(z_i^k, z_i^{k-1}) \propto p(z_i^k = l', z_i^{k-1} = l'') \tag{3.13}$$

where $l \in \mathbb{L}^x$, $l', l'' \in \mathbb{L}^z$ and $k \in [2; r]$.

In order to model the posterior probability from the right-hand side of Equations 3.12 and 3.13 we obviously cant use the contrast-sensitive approaches, which is relatively simple and gives reliable results, when using for the within-layer potentials. Nevertheless, we can model the posterior probability from right hand side of Equation 3.13 with the naïve Potts models from Equation 2.43, what reflects the independence of function $\xi_i^z(z_i^k, z_i^{k-1})$ from the data. The only possibility to model the posterior probability from right hand side of Equation 3.12 is to apply one of the data dependent approaches, proposed in the Section 2.4.2. Let us consider both of them more closely.

Histogram Matrix. The data dependent approach to model within-layer pairwise potential, described in the Section 2.4.2.1 may be adapted also for modeling an inter-layer pairwise potential. For this purpose, from the training data we also generate a 2D histogram $H_\xi : \mathbb{L}^x \times \mathbb{L}^z \to \mathbb{N}$ of the co-occurrence of labels at different layers but the same image site. Thus, $H_\xi(l, l')$ is the number of image sites in the training data with $x_i = l$ and $z_i^1 = l'$. Then the matrix $H_\xi(l, l')$ is normalized, using either symmetric, or asymmetric approaches, resulting in a matrix $h_\xi(l, l')$ that is the basis for the potential $\xi_i^x(x_i, z_i^1; \mathbf{y})$:

$$p(x_i = l, z_i^1 = l' \,|\, \mathbf{f}_i(\mathbf{y})) = h_\xi(l, l') \tag{3.14}$$

Note that for the inter-layer pairwise potential function in Equation 3.14, we do not multiply the histogram matrix $h_{\xi}(l,l')$ with a contrast-sensitive term, as it was done in the original model in Equation 2.47.

Concatenated Edge Potentials. In contrast to the concatenated edge potentials described in Section 2.4.2.2, which were used for the within-layer pairwise potentials, here we redefine the concatenated set of class labels from Equation 2.48 as

$$\mathbb{C}_{\xi} = \mathbb{L}^x \times \mathbb{L}^z \tag{3.15}$$

and the potential function becomes

$$p(x_i = l, z_i^1 = l' \,|\, \mathbf{f}_i(\mathbf{y})) \equiv p(\left\{x_i; z_i^1\right\} = l'' \,|\, \mathbf{f}_i(\mathbf{y})), \tag{3.16}$$

where $l \in \mathbb{L}^x$, $l' \in \mathbb{L}^z$ and $l'' \in \mathbb{C}_{\xi}$. This data dependent model differs from the original one 2.49 in the usage of non-concatenated feature vector $\mathbf{f}_i(\mathbf{y})$. One more important difference concerning the training procedure: for the association potentials φ^x and φ^z we use training pairs (x_i, y_i), (z_i, y_i) respectively and for the occlusion potentials ξ_i^x – (x_i, z_i, y_i), thus the training data must have two separate layers of labels: one for the base and one for the occlusion layers.

3.4 Inference in Multi-Layer Graphs

In Section 2.5 we gave a short introduction to inference methods. The considered 'classical' message-passing inference methods (*e.g.* LBP, Viterbi) were originally developed for directed graphical models and afterward were adopted for undirected graphical models as well. The application of message-passing algorithms to mixed graphical models was not studied yet.

We start this section with rewriting the message update rule, defined by Equation 2.51 in a matrix form[5]:

$$\mu_{i\to j}(v_j) = \psi_{ij}^2(v_i, v_j)^{\top} \times \left[\varphi_i(v_i) \cdot \prod_{\substack{(i,k)\in\mathcal{E} \\ k\neq j}} \mu_{k\to i}(v_i) \right]. \tag{3.17}$$

A diagrammatic representation of Equation 3.17 on example of an undirected graph with 8 nodes is depicted in Figure 3.7:

As we can see from Figure 3.7 every two graph nodes v_i and v_j bound with an undirected

[5]In this thesis, without loss of generality, we will model the undirected edges via two directed arcs associated with the same potential $\sqrt{\psi(\cdot,\cdot)}$, what will affect only the inference stage.

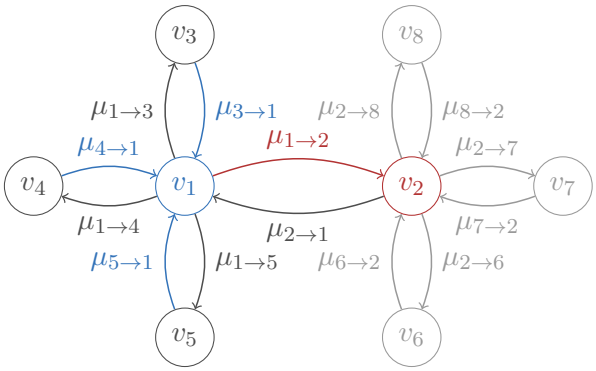


Fig. 3.7: Diagrammatic representation of the message update rule: $\mu_{1\to2}(v_2) = \psi_{12}^2(v_1,v_2)^\top \times [\varphi_1(v_1)\cdot\mu_{3\to1}(v_1)\cdot\mu_{4\to1}(v_1)\cdot\mu_{5\to1}(v_1)]$. Red color represents the message to be updated and the blue color - messages needed for the update.

edge $(i,j)\in\mathcal{E}$ (we still consider a message-passing algorithm for undirected graphical models) induce sending two opposite-directed messages $\mu_{i\to j}$ and $\mu_{j\to i}$. Before proceeding any further we prove one theorem:

Theorem 3.4. *For a message-passing inference algorithm and for any two graph nodes $v_1,v_2\in\mathcal{V}$, the following linking is equivalent till the power of the corresponding pairwise potential function:*

$$v_1 - v_2 \quad\equiv\quad v_1 \leftrightarrows v_2.$$

Proof. According to the graph construction rules B.5 given in Appendix B the left-hand side clique represents the joint probability $p(v_1,v_2)$, whereas the right-hand side clique represents two conditional probabilities: $p(v_1\,|\,v_2)\cdot p(v_2\,|\,v_1)$. Taking into account the approximation of these probabilities with the potential functions as given in Section B.4 we can rewrite the statement of the theorem in the following form:

$$\varphi(v_1)\cdot\psi(v_1,v_2)\cdot\varphi(v_2) = \varphi(v_1)\cdot\psi(v_1,v_2)\cdot\varphi(v_2)\cdot\psi(v_1,v_2)$$
$$\varphi(v_1)\cdot\psi(v_1,v_2)\cdot\varphi(v_2) = \varphi(v_1)\cdot\psi(v_1,v_2)^2\cdot\varphi(v_2)$$

Thus, we have shown that when using the set of rules from Section B.4 for decomposing the probabilities into the potential functions, the statement of the theorem holds till the power of pairwise potential function $\psi(v_1,v_2)$. □

From Theorem 3.4 follows a very important property of the pairwise potential functions:

Proposition 3.7. *A pairwise potential function, associated with an edge must be a symmetric function, whereas a pairwise potential function, associated with an arc may be an arbitrary function.*

$v_1 \rightarrow v_2$ *Asymmetry between leader and follower*

$v_1 - v_2$ *Symmetry*

Please, refer also to Appendix B for more details.

Finally, we use modified tree-reweighted message-passing algorithm from Section 2.5.2. According to the Theorem 3.4 in order to apply it to the mixed graphs, we represent each undirected graph edge as two opposite-directed arcs. This results in a graph having directed arcs only. We allow the messages to be passed only in opposite arc directions. This algorithm works by formulating the energy minimization problem as an integer programming problem and then solving its linear relaxation. Our inference algorithm is briefly summed up in Algorithm 4.

Algorithm 4: Message-passing inference

Data: Unary and pairwise potentials: $\varphi_i(v_i)$, $\psi_{i,j}(v_i, v_j)$, $\forall i, j \in \mathcal{V}, (i,j) \in \mathcal{A}$.
Result: Marginal probabilities $p_i(v_i), \forall i \in \mathcal{V}$.

repeat
  forall $i \in \mathcal{V}$ **do** // for all graph vertices v_i
    forall $j : (i,j) \in \mathcal{A}$ **do** // for all outgoing arcs
      $tmp \leftarrow \varphi_i(v_i)$;
      forall $k : (k,i) \in \mathcal{A} \setminus (j,i)$ **do** // for all incoming arcs, except (j,i)
        $tmp \leftarrow tmp \cdot \mu_{k \rightarrow i}$;
      $\hat{\mu}_{i \rightarrow j} \leftarrow \psi^2_{i,j}(v_i, v_j)^\top \times tmp$;
      $Normalize(\mu_{i \rightarrow j})$;
  forall $i, j : (i,j) \in \mathcal{A}$ **do** // for all graph arcs
    $\mu_{i \rightarrow j} \leftarrow \hat{\mu}_{i \rightarrow j}$;
until *converged*;
forall $i \in \mathcal{V}$ **do** // for all graph vertices v_i
  forall $k : (k,i) \in \mathcal{A}$ **do** // for all incoming arcs
    $p_i(v_i) \leftarrow \varphi_i(v_i) \cdot \mu_{k \rightarrow i}$;
  $Normalize(p_i(v_i))$;

3.4.1 Double Marginalization

As we have stated in Section 3.1, the main challenge of successful application of the ML-CRF model to the problem of handling occlusions is that we do not know beforehand where the occlusions are. We approach this challenge by splitting the inference process into 3 distinct phases [KSG18b]:

1. 'Simple' occlusion phase. First we classify visible regions and analyze the visible layer $\mathbf{z}^1$ in order to determine which image sites are likely to be occluded. The layers $\mathbf{z}^2, \ldots, \mathbf{z}^r$ and the

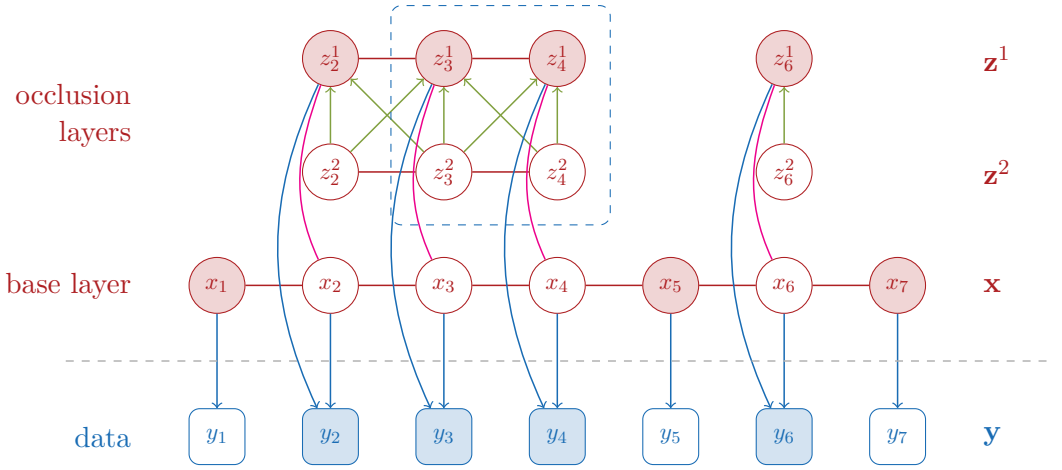


Fig. 3.8: Graph $\mathcal{G}^{ml}_{[\{z_1^1,z_1^2,z_5^1,z_5^2,z_6^2,z_7^1,z_7^2\}}$ with two occlusion layers: $\mathbf{y}$ – data; $\mathbf{x},\mathbf{z}^1,\mathbf{z}^2$ – labels; blue edges - unary (association) potentials; red and green edges - within-layer and inter-layer pairwise (interaction) potentials respectively; purple edges - occlusion potentials; dark blue nodes represent occluded regions, corresponding to the Figure 3.2, while dark red nodes - to the actually observed objects. The ambiguous section of the graph is marked out with dashed rectangle.

potentials ξ_i^z are data-independent, hence for the first phase the equality $\mathbf{z}^k = \mathbf{z}^1, \forall k \in [2;r]$ holds by construction. If a variable z_i^1 of the image site i does not take label *void*, we determine an occlusion for this site. Thus, we can estimate position and form of occluded regions in the scene. After that, we marginalize out the graph nodes, corresponding to the random variables, having class *void*. Hence we may end up with a graph structure, depicted in Figure 3.8. This graph corresponds to one, which depicted at the Figure 3.2.

Having decoded the labels for visible regions, we can now modify the matrices $\mathcal{T}_g$ from Equation 3.9 in such a way, that they make the desired class transitions on the boundary line or curve more likely. For that we read class labels l and l' of the regions, separated by that line and set the positions $\mathcal{T}_g(l;l')$ and $\mathcal{T}_g(l';l)$ to some large parameter. The performance of this approach is illustrated in Figure 3.9, where we see, that among other methods, our ML-CRF preserved the inherent region structure for all scenes.

2. 'Complex' occlusion phase. In order to detect *complex* occlusions (as those depicted in Figure 3.1) we continue analyzing the remaining random variables in $\mathbf{z}^1$. If a distinct group $\mathcal{O} \subset \mathbf{z}^1$ of random variables $z_i^1,\ i \in \mathcal{O} \subset \mathcal{Z}$ has the same class, we assume that this is a *simple* occlusion and all the underlying occlusion random variables $z_i^k = z_i^1,\ k \in [2;r],\ i \in \mathcal{Y}' \subset \mathcal{Y}$. In case if, the group $\mathcal{O}$ has variables with different labels, we meet the *problem of ambiguity*.

For example, let us consider Figure 3.8 in respect to the Figure 3.2, thus the variables z_3^1 and z_4^1 have values $z_3^1 = \tau$ and $z_4^1 = \kappa$. Both τ and κ labels belong to $\mathbb{L}^z$ and appear together at the visible layer. If the scene depth information is not available, it is not clear whether

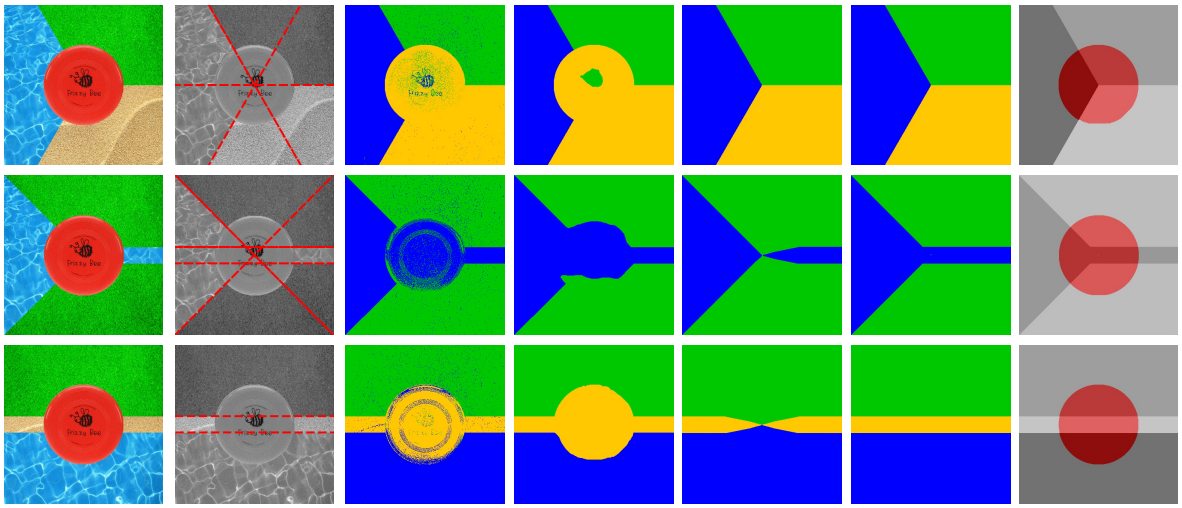


Fig. 3.9: Classification error on our synthetic dataset: red Frisbee disc occludes the structures at the base layer. Examples showing the classification results for occluded pixels using four methods.

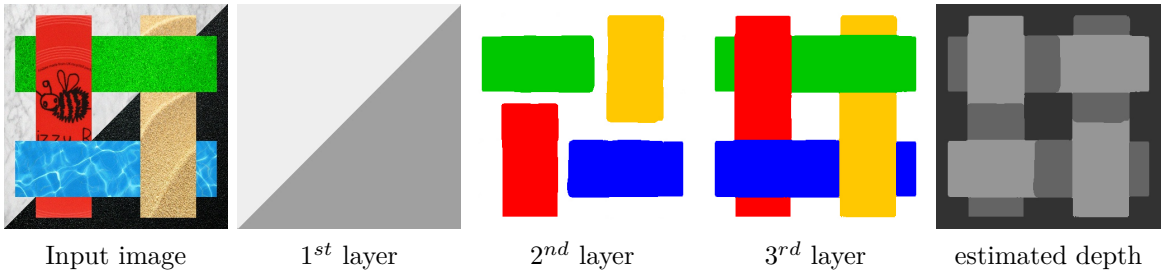


Fig. 3.10: Our synthetic dataset, illustrating a superposable occlusions. **From left to right:** Synthetic scene; ML-CRF classification results for the base layer, occlusion layer and visible layer; coarsely estimated depth from the scene layer decomposition.

object τ occludes κ or vice versa – object κ occludes object τ, *i.e.* there is not enough proof to make decision about which labels should take variables in $\mathbf{z}^2$. Both scenarios are possible (the third possible scenario when objects τ and κ are neighbors, *i.e.* not occlude each other, is trivial). The problematic section of the graph in Figure 3.8 is marked out with a dashed rectangle and considered more deeply in the Figure 3.11.

3. Layer decomposition phase. We overcome this problem by using two probe inference steps: In the first we marginalize out variables z_i^2, that stay behind variables z_i^1 with a class label κ, so the neighboring to z_i^2 variables z_j^2, $(i,j) \in \mathcal{E}$ are more likely to take label κ (mind the new directed arcs in Figure 3.11 (left)); In the second we marginalize out variables z_k^2, that stay behind those visible z_k^1 having class label τ. Note, that after marginalizing occlusion nodes out from the graph, the equality $\mathbf{z}^k = \mathbf{z}^1, k \in [2;r]$ may not hold anymore.

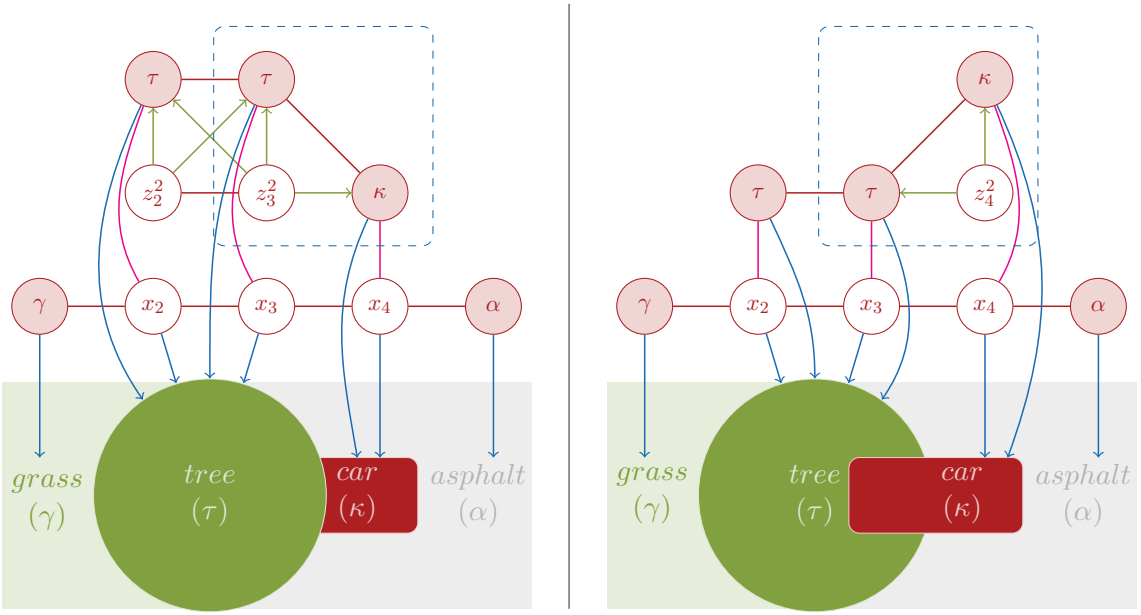


Fig. 3.11: Problem of ambiguity and its solution: double marginalization method. Neighboring sites have two different objects from occlusion layer $\tau, \kappa \in \mathbb{L}^z$ – two scenarios are possible: **Left:** object τ occludes object κ scenario: variables z_1^2 and z_2^2 are marginalized out; **Right:** object κ occludes object τ scenario: variables z_3^2 and z_4^2 are marginalized out.

After each marginalization we estimate the label confidences for the random variables of occlusion layers, using the formula:

$$conf(z) = 1 - \frac{\max(\varphi^a(z) \setminus \max(\varphi^a(z)))}{\max(\varphi^a(z))}, \tag{3.18}$$

where $\varphi^a(z)$ are from Equation 3.6. Then we chose the most probable variant, which random variables z_i^2 or z_k^2 should be marginalized out. We call this procedure *double marginalization* and apply it for each region of the graph with *complex* occlusions. Additionally, after the inference process, the scene depth for patch i may be coarsely estimated from the number of remaining nodes $z_i^{\cdot}$ (please see Figure 3.10, which depict a synthetic scene, corresponding to the Figure 3.1).

3.5 Two-Layer Conditional Random Fields

Here we describe a special case of multi-layer CRF, with only one occlusion layer, *i.e.* $r = 1$, what results in a *two-layer* CRF model (tCRF, [Kos+13a]). It is worth to mention, that in many applications two layers are sufficient. In general, one occlusion layer is sufficient for separating foreground from background.

We model the posterior probability $p(\mathbf{x}, \mathbf{z} \,|\, \mathbf{y})$ directly, simplifying the model in Equa-

tion 3.5:

$$p(\mathbf{x},\mathbf{z}\,|\,\mathbf{y}) = \frac{1}{Z}\prod_{i\in\mathcal{Y}} \varphi_i^x(x_i;\mathbf{y})\ \varphi_i^z(z_i;\mathbf{y})\cdot \prod_{i\in\mathcal{Y}}\prod_{(i,j)\in\mathcal{E}} \psi_{ij}^x(x_i,x_j;\mathbf{y})\ \psi_{ij}^z(z_i,z_j;\mathbf{y})\cdot \prod_{i\in\mathcal{Y}} \xi_i^x(x_i,z_i;\mathbf{y}). \tag{3.19}$$

its parametric version as simplification of the model in Equation 3.11:

$$p(\mathbf{x},\mathbf{z}\,|\,\mathbf{y},\boldsymbol{\theta}) = \frac{1}{Z}\prod_{i\in\mathcal{Y}} \left\langle \varphi_i^x(x_i;\mathbf{y}),\boldsymbol{\theta}_\varphi^x\right\rangle \left\langle \varphi_i^z(z_i;\mathbf{y}),\boldsymbol{\theta}_\varphi^z\right\rangle \cdot \prod_{i\in\mathcal{Y}}\prod_{(i,j)\in\mathcal{E}} \left\langle \psi_{ij}^x(x_i,x_j;\mathbf{y}),\boldsymbol{\theta}_\psi^x\right\rangle \left\langle \psi_{ij}^z(z_i,z_j;\mathbf{y}),\boldsymbol{\theta}_\psi^z\right\rangle \cdot \prod_{i\in\mathcal{Y}} \left\langle \xi_i^x(x_i,z_i;\mathbf{y}),\boldsymbol{\theta}_\xi\right\rangle. \tag{3.20}$$

In Equation 3.20, $\boldsymbol{\theta} = \left\{\boldsymbol{\theta}_\varphi^x,\boldsymbol{\theta}_\varphi^z,\boldsymbol{\theta}_\psi^x,\boldsymbol{\theta}_\psi^z,\boldsymbol{\theta}_\xi\right\}$ are model control parameters, including weights, which modulate the influence of the individual terms in the classification and those, which are included into the potential functions.

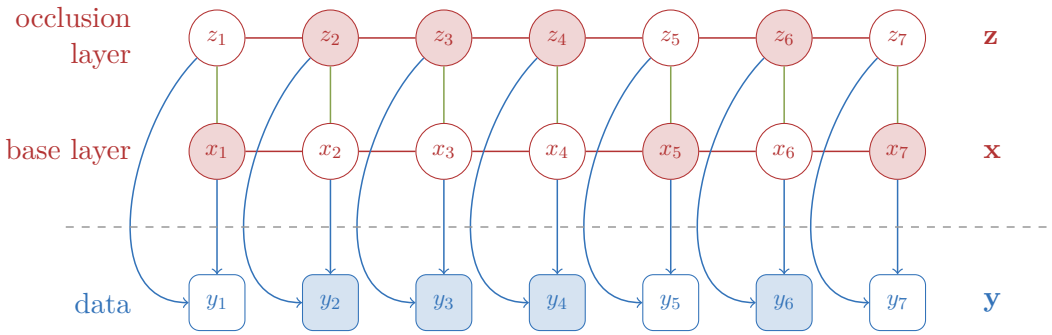


Fig. 3.12: Structure of the two-layer CRF model. The second dimension and additional links between data and labels are omitted for simplicity. Squares and circles correspond to observations $\mathbf{y}$ and labels $\mathbf{x},\mathbf{z}$, respectively. The dark blue nodes correspond to a region with occlusion. The red and green edges – within-layer and inter-layer pairwise potentials respectively.

Figure 3.12 shows the structure of our two-layer CRF model. Squares and circles correspond to observations and labels, respectively. Figure 3.12 corresponds to the example, depicted at Figure 3.2, so the dark blue nodes (data sites 2 – 4 and 6) correspond to a region with occlusion, *i.e.* where only the occluding object is visible; and thus, the dark red nodes – to the labels, corresponding to actually visible objects. The graph edges represent dependencies between the nodes. At such figures, we will also designate the association potentials with blue color, within-layer pairwise potentials with red color and inter-layer pairwise potentials

– with the green color. Note that after conditioning on observed data $\mathcal{Y}$ the graph $\mathcal{G}^{tl}[^{\{\mathcal{Y}\}}$ is undirected.

3.6 Experiments

In this section the presented in this chapter two-layer and multi-layer CRF models are evaluated and compared with the classical single-layer CRF technique. In particular, the experiments cover the following methods developed by me:

- Two-layer conditional random field model (*tCRF*, [Kos+13a], Section 3.5), based on undirected graphs;
- Multi-layer conditional random field model (*ML-CRF*, Section 3.3.1), based on mixed graphs with underlying methods:
  - Concatenated model for inter-layer pairwise potentials which correspond to the new directed graph arcs in ML-CRF (Section 3.3.2);
  - Double marginalization technique for decoding occluded regions (Section 3.4).

It is a prerequisite of the proposed methods that the training data also have two separate layers of labels: one for the base and one for the occlusion layers. Hence only the datasets providing such information may be used. For the evaluation two suitable data-sets with two layers of references were selected: *Vaihingen* and *StreetScenes*. For further information about these data-sets, please refer to the Appendix D.

The lists of extracted features are also given in Appendix D. Here all of them except the *car confidence* feature are used. In spite of the fact that in Chapter 2 the car confidence feature was proven to be a very useful feature, in the following experiments I want to show that the ML-CRF model can reveal the occluded regions without application of any object detectors. For numerical reasons, all features are scaled linearly into the range $[0;255]$ and then quantized by 8 bit. The CRF-classification is based on the *direct graphical models* C++ library [Kos15]. Each image site i in Equations 3.5 and 3.19 to be connected to its four nearest neighbors j in the data grid, thus the red edges of the graphical model in Figures 3.5 and 3.12 are defined.

In each experiment, 62,5 % of the available samples were used for training (50 % for potential functions and 12,5 % for control parameters $\boldsymbol{\theta}$); the remaining 37,5 % of samples were used for evaluation. For the association potentials the random forest model, described in the Section 2.3.2.3 and consisting of N_T=100 trees of maximal depth 15 was used, whereas for the within-layer interaction potentials – the data dependent model, based on the histogram

matrix from Equation 2.47. This combination of unary and pairwise models recommended itself as one of the most reliable in Section 2.7.3. Finally, the inter-layer interaction potentials were modeled with concatenated edge potentials from Equation 3.16. The control parameters $\boldsymbol{\theta}$ were estimated with the help of Powell search method described in Section 2.6.2. The resulting label maps were compared with the ground-truth; hence the precision and recall of the results per class as well as the overall classification accuracy [Lew90] are reported.

Baselines. The following baselines were chosen to be compared with the proposed ML-CRF and tCRF models:

- boosted decision trees-based classifier of *Guo & Hoiem* [GH12].[6]

Other general-purpose methods to infer labels of occluded regions do not exist in the literature, so I provide several baselines. Each method attempts to predict the labels of the underlying surfaces, given the label confidences for the visible surfaces:

- *Most confident background* assigns each foreground pixel to the most confident background label (what corresponds to the classical CRFs);
- *Nearest* method assigns occluded background pixels to the nearest (in image location) visible background pixel;

3.6.1 Two-Layer Conditional Random Fields

Existing road extraction methods are far from being practically relevant in challenging environments, like in suburban regions, where model assumptions about roads are violated [May+06]. The main reasons for failure of existing methods were occlusion of the road surface by cars, trees and a complex 3D geometry, *e.g.* at motorway interchanges. This was the main motivation for the two-layer CRF model. Thus, our evaluation will be especially concentrated on the labeling of the occluded areas.

To assess our two-layer CRF model we carried out a number of different experiments. At the first stage we used the Vaihingen data-set and performed two experiments: in the first experiment CRF, each layer was processed independently, thus the inter-layer interaction potentials ξ were not considered. In the second experiment $tCRF$ we use the two-layer CRF model with the inter-layer interaction potentials. For all the experiments in this section, we used the same models for association and interaction potentials.

Figure 3.13 shows the convergence behavior of the Powell method for training the parameters $\boldsymbol{\theta}$ in Equation 3.20 for both cases. It shows that originally the procedure converges

[6] Accuracies and figures for this baseline are taken from the corresponding publication.

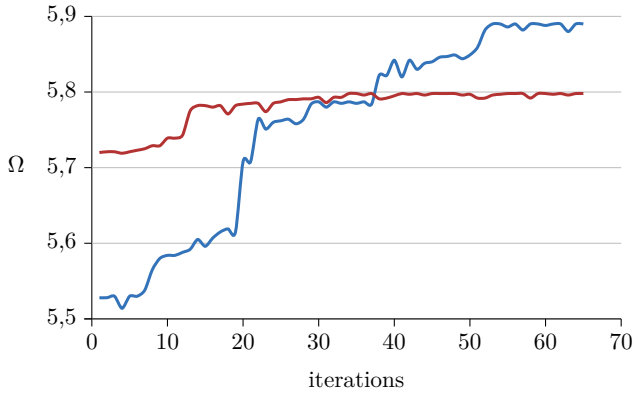


Fig. 3.13: Convergence of the Powell search method: Red curve: CRF; blue curve: $tCRF$.

more slowly for the $tCRF$ method, probably due to a relatively poor initialization of some parameters, but in the end-iterated state, a larger value of the objective function Ω from Equation 2.56 can be achieved. For the Vaihingen dataset the parameters were: θ_d=0,5, θ_s=0,3, θ_o=18,5, θ_1=1086, θ_2=0,01 and θ_3^g=100, $\forall g \in \mathcal{G}$.

		CRF		$tCRF$	
	Category	Rec.	Prec.	Rec.	Prec.
base layer	*asphalt*	80,2 %	90,3 %	85,0 %	87,7 %
	building	86,5 %	78,3 %	85,9 %	82,5 %
	grass	82,7 %	85,5 %	88,3 %	87,8 %
	agriculture	84,1 %	64,4 %	85,4 %	84,2 %
	overall accuracy	**82,6 %**		**86,6 %**	
occl. layer	*void*	78,1 %	96,9 %	86,8 %	95,7 %
	tree	90,4 %	58,0 %	86,3 %	65,4 %
	car	72,7 %	11,5 %	47,7 %	19,4 %
	overall accuracy	**80,4 %**		**86,3 %**	

Tab. 3.1: Comparison between two-layer ($tCRF$) and single-layer CRF (CRF) models on Vaihingen dataset. Recall (Rec.) and Precision (Prec.) of the experimental results for Vaihingen dataset.

Figure 3.14 shows the results of the experiments for two Vaihingen scenes (22 and 43). In the figure we can observe that our two-layer model considerably improves the road classification in comparison to the state-of-the-art single-layer model. For example, in the right part of the scene 22, the $tCRF$ model successfully extracts a road part that is completely occluded with a tree, while CRF wrongly labels this area as grass. This improvement is possible because the $tCRF$ models explicitly considers occlusion, the results of the base layer receiving information from spatially neighboring image sites, multi-scale features, and the second layer of labels. The scene 43 also shows how an occluded road can be correctly classified by the

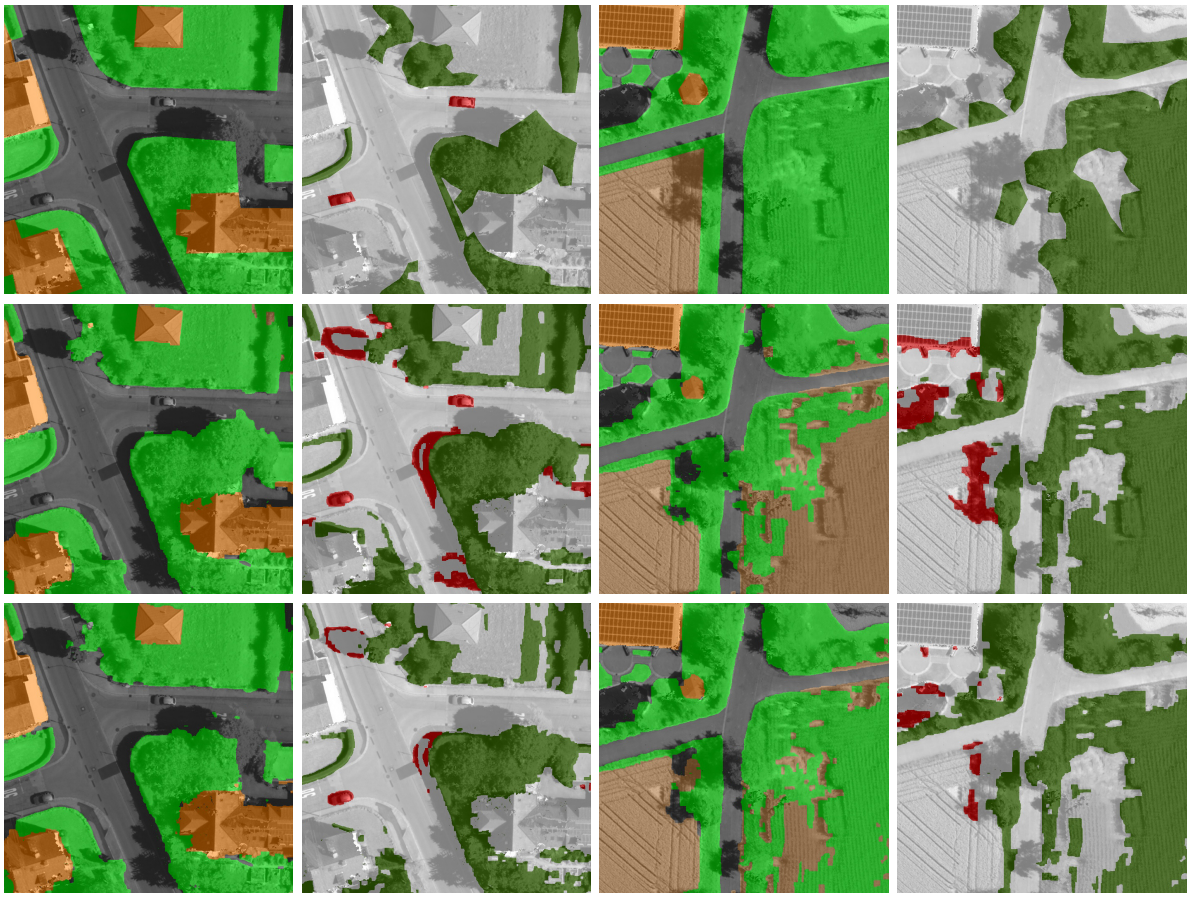


Fig. 3.14: Comparison between classical CRF and two-layer CRF approach. **Left to Right:** scene 22 base and occlusion layers, scene 43 base and occlusion layers. **Top to Bottom:** groundtruth, CRF, $tCRF$. Gray: *asphalt*; orange: *building*; green: *grass*; beige: *agriculture*; white: *void*; dark-green: *tree*; red: *car*.

$tCRF$. In addition, the grass area in the right lower part of the scene is labeled as agricultural by the CRF model, in spite of the occlusion layer saying that this region is covered by trees. Agricultural regions are rarely covered by a forest, and the $tCRF$ model can use this knowledge (derived from the training data) in order to classify this area correctly. For both scenes we can observe many false positives for the class *car*. Their number is reduced considerably by the $tCRF$ model, though at the cost of a few false negative cars. This is also reflected in the quality numbers in Table 3.1.

The recall and precision as well as the overall accuracy of the results achieved in these two experiments are shown in Table 3.1. Using the CRF model, the overall accuracy of the classification was 82,6% for the base layer and 80,4% for the occlusion layer. In the second ($tCRF$) experiment the overall accuracy for the base layer was 86,6%. The improvement can be attributed by more accurate classification in the occlusion areas (Figure 3.14). From

	CRF	*MC*	*GH*	*tCRF*
road	90,9 %	92,5 %	93,0 %	**94,5 %**
sidewalk	0,5 %	28,5 %	**52,5 %**	0,3 %
building	54,3 %	**90,5 %**	90,0 %	46,6 %
store	0,0 %	0,5 %	**11,0 %**	0,1 %
tree	92,3 %	69,5 %	73,5 %	**92,9 %**
sky	77,6 %	68,0 %	79,0 %	**80,4 %**

Tab. 3.2: Completeness of the classification results for *StreetScene* dataset. *CRF*: single-layer CRF; *MC*: Most confident background method and *GH*: method of Guo & Hoiem; *tCRF*: our method.

the Table 3.1, we can also observe that both the recall and precision of class *car* are still very low. This is due to the fact that cars are relatively small regions and so are described with our features not well enough. The outcome of additional car-detector may correct this situation: please refer to Section 2.7.1.3. Nevertheless our *tCRF* model has almost double precision value for cars, than *CRF* model, while having smaller recall value. As far as recall and precision are concerned, the major improvement is an increased precision for *asphalt* and an improved recall for *grass*. The class *agriculture* has a rather low correctness in the model CRF. For the occlusion layer, we observe the best performance when using *tCRF*.

Fig. 3.15: StreetScenes: examples of base layer classification. **Top row:** Reference; **Bottom row:** *tCRF* classification result. Gray: *road*; blue: *sidewalk*; orange: *building*; green: *tree*; cyan: *sky*.

At the second stage of experiments we used the *StreetScene* data-set and also performed two experiments: *CRF* and *tCRF* but this time we also compare our method with *Most confident background* (*MC*) and *Guo & Hoiem* (*GH*) baseline methods. The results are

presented in Table 3.2 and some classification examples are depicted at Figure 3.15. Table 3.3 depicts a confusion matrix for the base layer, achieved in the $tCRF$ experiment.

	asphalt	*building*	*grass*	*agriculture*
asphalt	27,50	1,13	4,10	0,07
building	2,03	13,47	1,12	0,02
grass	2,17	0,81	36,38	0,98
agriculture	0,10	0,11	1,73	8,26

Tab. 3.3: Confusion matrix for the base layer, achieved on Vaihingen dataset [%].

As we can see from Table 3.2 neither CRF nor $tCRF$ can distinguish *sidewalk* and *store* classes. Nevertheless, our $tCRF$ method beats the baseline GH method in terms of classification accuracy for 3 of 6 classes: *road*, *tree*, *sky*.

3.6.2 Multi-Layer Conditional Random Fields

To assess our multi-layer model we carried out a number of experiments. First we used Vaihingen dataset to compare our ML-CRF approach with the *Most confident background* (MC) baseline method. Figure 3.16 shows the classification results for 3 scenes with massive occlusions, caused by trees. We can observe that our model (5^{th} row) considerably improves the road classification in comparison to the baseline model (4^{th} row). ML-CRF model successfully extracts road parts that are completely occluded with trees, while MC wrongly labels this area as grass. This improvement is possible because the ML-CRF models explicitly considers occlusion, the results of the base layer receiving information from spatially neighboring image sites, multi-scale features, and the additional layers of labels. Additionally the label-maps, achieved with the ML-CRF appeared to be more "smooth", which is caused by additional links connecting the base layer with the occlusion layers.

Figure 3.18 (c) present the percentage of correctly labeled pixels for different classes of the base layer. Note that improvements of 4-5% over the baselines do not fully convey the large qualitative improvement that can be seen in Figure 3.18 (d), where only occluded pixels were evaluated. We can observe that the layered models definitely outperform baseline methods for classes *road* and *grass*. We explain low rates for classes *house* and *agro* by the fact that buildings and agricultural areas are very seldom occluded and such cases may be related to the reference mistakes.

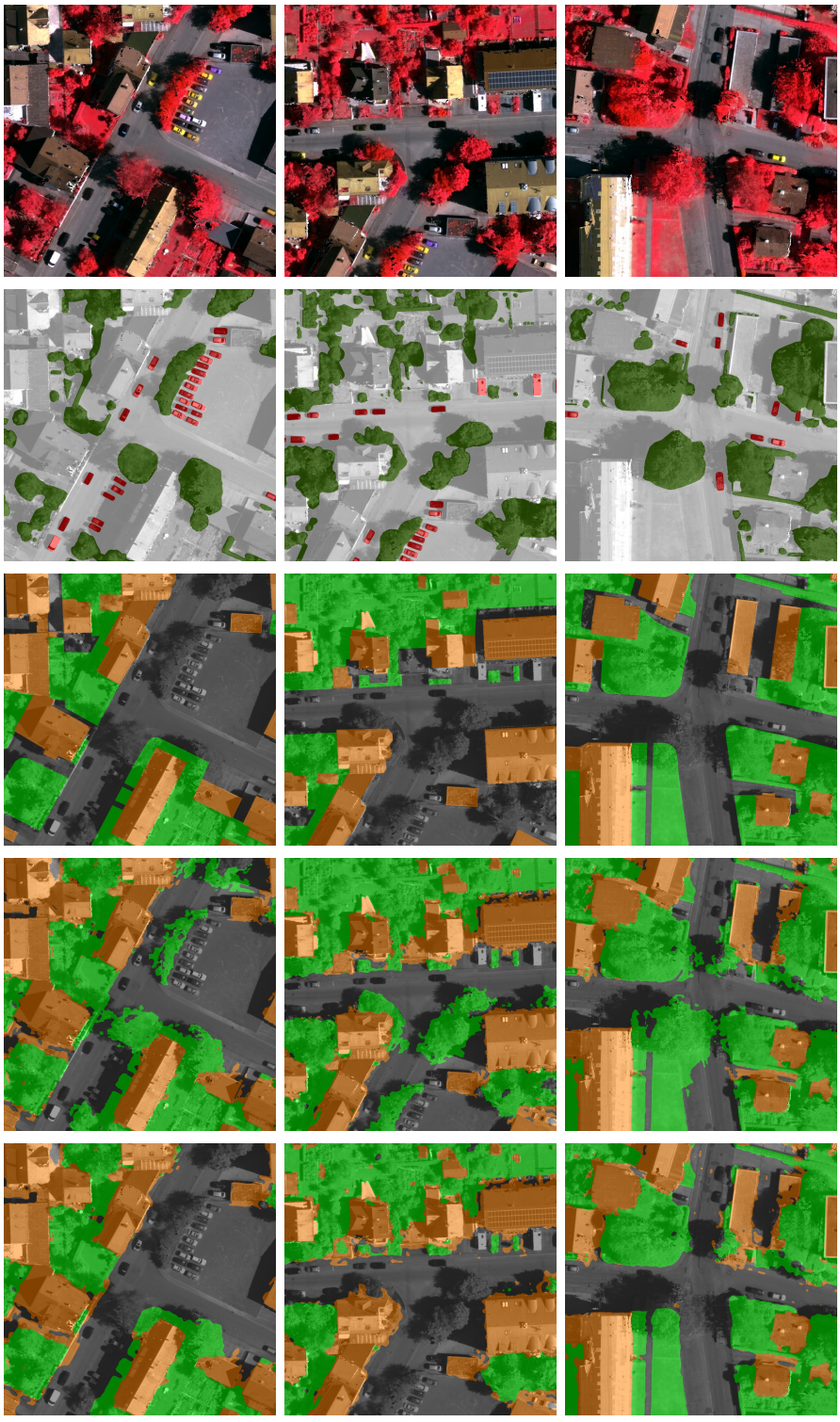


Fig. 3.16: Classification results on Vaihingen dataset: Top to Bottom: Original image; Reference for occlusion layer; Reference for base layer; Most Confident; ML-CRF. Gray: *asphalt*; orange: *building*; green: *grass*; white: *void*; dark-green: *tree*; red: *car*.

At the second stage of experiments we used StreetScene dataset and compared our ML-CRF approach with baselines, including the classifier of Guo & Hoiem. The results are presented at Figure 3.18 (a,b) the classification examples are depicted at Figure 3.17.

Fig. 3.17: Classification results on StreetScenes dataset: Left to right: Original image; Reference for base layer; Guo & Hoiem method with polygon fitting; ML-CRF. Green: *road*; yellow: *sidewalk*; red: *building*; orange: *tree*; blue: *sky*; gray: *unknown*.

As we can see from Figure 3.18 (a) none of the methods can distinguish *store* class. This is because the stores are indistinguishable from usual buildings in sense of the used features. Additional shop advertisement detector may fix that. Class *sky* in Figure 3.18 (b) has zero values, because in the dataset sky was never occluded. We can observe that our approach

produces comparable results to the baseline methods and even significantly outperforms them for some classes *e.g.* class (*tree*) up to 23%.

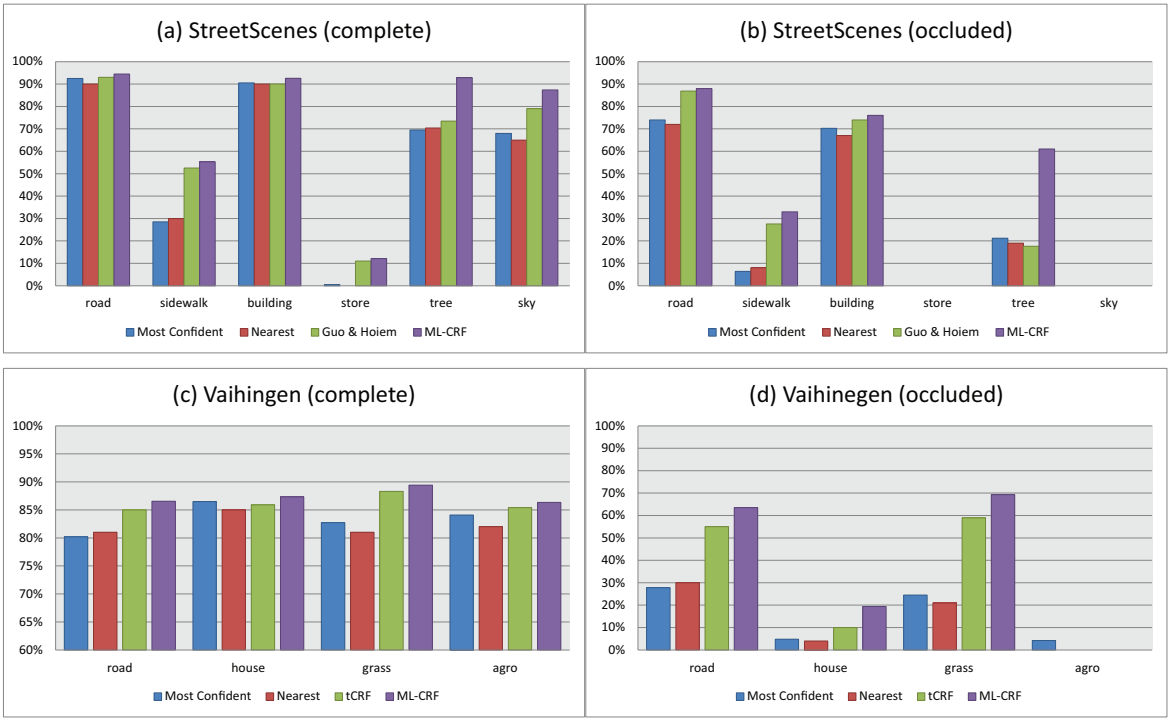


Fig. 3.18: Per-class pixel accuracy on both datasets we experimented with.

3.7 Summary

In this chapter a solid image classification framework for handling occlusions with Markov- and conditional random fields was presented. This framework especially targets the challenging problem of classifying the occluded parts of the 3D scene depicted in a 2D image. Due to the layered structure the presented approach is capable to improve the classification rate for partially occluded objects without applying object detectors or polygon fitting techniques.

Specifically, I proposed a novel "Multi-Layer-CRF" framework that allows for the integration of sophisticated occlusion potentials into the model and enables the automatic inference of the layer decomposition. Unlike other image labeling techniques where a single label is determined for each pixel, layered model assigns multiple labels to pixels: one for the visible object and others – for occluded objects, if they exist. The presented framework is based on the mixed graphical models, which explicitly encode causal relationship between the visible and occluded regions.

In order to decode the complex occlusions with the presented model I proposed the *double marginalization* method, which is capable to reveal 3D structure of the scene from a 2D

image only. For the double marginalization method the operations of conditioning on– and marginalization out a subset of nodes in a mixed graph were re-defined. By proving Theorem 3.1 it was shown that these operations do not contradict the corresponding operations for probability distributions. These graph transformation operators make possible to modify the graph on-the-fly during the inference process, while keeping the corresponding probability distribution consistent.

Finally a special message-passing algorithm to perform maximum a posterior inference on mixed graphs was used and it's ability to infer the correct labels of occluded regions in real-world scenes was proven. Thus, I demonstrated the generality and performance of my method on the problem of occlusions arising in airborne- and street-view images. It is shown to increase the classification accuracy in occluded areas by up to 23%.

4 Conclusion and Outlook

4.1 Conclusion

The main topic of this thesis was to introduce a systematic approach for the accurate design and the efficient implementation of layered conditional random fields methods, that could handle occlusions. This was done successively in two steps:

Efficient CRF. The scientific contributions in Chapter 2 start with the investigation of how different features affect the classifier performance. With help of the car detector, which uses SVM and rotation-invariant features, car confidence values were achieved. The car confidence feature, which is based on the output of the car detector and has a minor number of false positives is shown to increase the accuracy of classification especially for aiming class car.

The important conclusion made from the successful application of the confidence feature is that *the feature responses on different categories should bring maximum inter-category separability while keeping low variabilities within distinct categories.* A similar requirement is also set for the potentials, produced by classifiers. This encourages trying *DCNN features* which can be extracted with help of an auxiliary classifier, based on convolutional neural networks. These features were tested on a challenging EMDS dataset from a microbiology domain and showed both outstanding accuracy and efficiency.

Next, I concentrated on more efficient classification models used for the potential functions. For the unary potentials the sequential algorithm for GMM training was presented. This sequential approach is 6 times faster and needs far less memory than classical GMM implementation that is based on a Expectation Maximization algorithm. The method was evaluated on a set of airborne images and also showed overall classification accuracy improvement in comparison to the classical GMM approach.

An optimization scheme for the KNN classifier was also proposed. My KNN implementation is based on KD-tree data structure and takes into consideration only those neighbors, which lie in a small neighborhood of the nearest found neighbor. The experimental results showed that the introduced algorithm is more than 5 times faster than the original KNN implementation from the OpenCV library [Ope14] and produces nearly the same classification accuracy.

For modeling the pairwise potential functions, the novel concatenated model was pre-

sented. This model in fact a general data-dependent interaction model that uses unary potential functions as underlying models. Experimental results showed that together with the histogram-based matrix model the concatenated model outperformed all previously existing pairwise models for CRFs.

Multi-Layer CRF. In Chapter 3 a novel approach, called *multi-layer CRF* for considering occlusions in classification was presented. This approach is inspired by an idea of CRFs, but it is based on mixed graphical models, thus the operation of marginalization and conditioning for mixed graphs must be re-defined and a suitable message-passing algorithm for inference must be offered.

Due to the layered structure of the ML-CRF model, it is capable to improve the accuracy of area classification for partially occluded objects. The method was evaluated on the set of airborne- as well as on street-view images and showed a considerable improvement of the overall accuracy in comparison to the classical CRF approach.

With the work that was done in this thesis I have demonstrated that *conditional random fields are not only a method for structured prediction, which considers 'neighboring' samples*: with the freedom of modeling the interactions, which CRF provides, it becomes a powerful tool, which allows to 'see' *unobserved* entities.

4.2 Future Work

Although the proposed framework allows the construction of very fast and highly accurate conditional random fields techniques, there remain a lot of things that can be done. Moreover, the introduced and proposed approaches in this thesis gave birth to new questions and ideas. In the following some of the most promising of these ideas are sketched:

CRF Modeling Ideas Let me start with four aspects that concern an improved modeling or the integration of additional concepts in the existing framework.

- *Deep Learning.* Currently the deep learning techniques for semantic image segmentation tasks became extremely popular because of their high accuracy. DCNNs, for example, substitute classical CRFs in many areas of computer vision and pattern recognition. Despite the DCNNs' main disadvantages (*e.g.* low learning speed and high demands to the volumes of training data), they outperform the models that use classical classifiers, which were considered in Chapter 2. However, the combination of layered CRF and deep learning techniques may produce more efficient and accurate results.

There were already works to represent a CRF as recurrent neural network [Zhe+15], and within this thesis the DCNN features were used as underlying features for the RF model (Section 2.2.2). In Section 2.3.2.4 it was shown that the ANN model may be successfully used as the association potential for CRF. Nevertheless, the use of deep learning techniques as specific association or interaction potentials was not studied and in the future, I would investigate the combination of CRF and DCNN more in detail.

- *Problem-Specific potentials.* In this thesis I have compared 8 models for the association potentials and 5 models for the interaction potentials. For some specific problems, or even for some specific categories the best results were achieved by one model, for another problem – by another model. The proposed within this thesis idea of association several graph nodes with the single observation, makes it possible to classify observations using several association potentials, built upon different models, simultaneously. This fits the ideas of *ensemble learning* [OM99] and *boosting* [Bre96]. Consequently, the new data-dependent pairwise potentials, connecting these several graph nodes, should be developed. These pairwise potentials should manage the contribution of every potential depending on the observed data and neighboring nodes.

- *Data Fusion.* Probabilistic methods used for modeling the association potentials of CRFs are also commonly used for data fusion. However, incorporating multiple layers into the CRF technique allows for combining data from completely different domains, which is impossible to project to one unified calculation grid. For example, in geodesy a higher order CRF model may be used for simultaneous classification of land cover and land use; in photogrammetry – for multitemporal and multiscale classification of optical satellite imagery; in non-destructive testing – for detection and localization of flaws in metals by classification of ultrasonic and X-Ray data.

- *Optical Flow Estimation.* Another interesting aspect that exceeds the current modeling framework is the use of CRFs for estimation of the optical flow (and subsequently stereo). For this task, the nodes' association potentials should be given in form of the 'energy functionals' getting high when pixels of both images match. The interaction potentials should take the role of the 'smoothness' term, regularizing the optical flow field. This idea is not completely new in computer vision: [KZ01; SP07; Kos15], however the use of multiple layers, representing the 3D scene structure will allow to handle occlusion – the main problem in accurate stereo estimation, in a more natural way.

Numerical Ideas. The main efficiency questions to the framework are related to three processes: training, classification and inference. Since the training may be performed offline,

classification and inference remain the most acute efficiency questions. Apart from the development of new and more efficient classification and inference methods, there are also one general numerical idea that may be worth being investigated.

- *Direct Parallelization.* In order to improve the performance of the ML-CRF method even further, one may think of a direct parallelisation of the proposed inference and decoding schemes. In contrast to domain decomposition techniques that tackle the problem indirectly by decomposing the graph, such a direct strategy only distributes the computation itself and thus requires a much lesser computational overhead. This in turn allows for higher speedups with similar numbers of CPUs. Recent results for a direct parallelisation of the message-passing algorithms presented in [ASP11] confirm the usefulness of the latter approach.

Semantic 3D from 2D. The proposed multi-layer approach has much more broader specter of applications if thinking about its capacity to classify occluded areas as the ability to assign correct labels to the data, which is not directly observed. In domains as RGB-D imagery, spatio-temporal reconstruction, *etc.* this ability is even more important since the labels must be assigned to the graph nodes, for which no data is available. My multi-layer approach drains necessary information for such 'no data' regions from its surrounding in 3D. However, additional temporal or depth information, supporting the introduced ML-CRF technique may increase the classification rate of the occluded regions heavily.

- *Features Based on Motion Analysis.* When in addition to the scene image, a short temporal sequence of images (a few seconds before and after the scene image) is available, it should be possible to extract special features for the association potentials. Without considering 'structure-from-motion' techniques, the features based on motion analysis may be extremely useful for assigning the moving objects to the foreground layer and, thus, for separating foreground from the background. Such temporal data supporting the presented ML-CRF model might increase the classification accuracy significantly. It is worth mentioning, that the groundtruth for a single frame of the sequence should be enough for the model to train.

- *Features Based on Stereo and Laser Scanning.* Secondly, the additional depth information, achieved from stereo or laser scanning might be extremely useful. As it was shown in Chapter 3, depth information may be consistently incorporated into the presented framework as an additional feature. In combination with the *Data Fusion* idea from above, the depth information can be encoded not only as the features and graph struc-

ture, but also into the additional CRF layer, where the depth would be estimated in the same stochastic way as categories.

All these ideas give just a small impression on the realm of things that can still be done. However, one thing is for sure: *There will always be a need for fast and accurate algorithms in computer vision.*

A Notations

p	probability (formally an array of probabilities)
$\mathcal{J}$	independence model
$\mathcal{V} = \{1, 2, \ldots, n\}$	array of vertexes in a graph (indexes)
$\mathcal{Y} \subset \mathcal{V}$	array of data vertexes
$\mathcal{X} \subset \mathcal{V}$	array of base layer class vertexes
$\mathcal{Z} \subset \mathcal{V}$	array of occlusion layer class vertexes
$\mathcal{E} = \{(1,2), (1,3), \ldots, (\cdot, \cdot)\}$	array of edges, *i.e.* undirected links (pairs of indexes); here $(i,j) \equiv (j,i)$
$\mathcal{A} = \{(1,2), (1,3), \ldots, (\cdot, \cdot)\}$	array of arcs (directed links) (pairs of indexes); here $(i,j) \neq (j,i)$
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	(undirected) graph
$\mathcal{D} = (\mathcal{V}, \mathcal{A})$	directed graph
$\mathcal{G}^m = (\mathcal{V}, \mathcal{E}, \mathcal{A})$	mixed graph
$\mathcal{G}^{ml} = (\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \mathcal{E}, \mathcal{A})$	multi-layer graph
$\mathcal{C} \subset \mathcal{G}$	array of cliques
$\boldsymbol{\pi} = \{l_1, l_2, \ldots, l_{n-1}\}$	path

$\boldsymbol{\pi}(v_i, v_{j+1}) = \{l_i, \ldots, l_j\}$	subpath
$\mathbb{X} = \{x_1, x_1, \ldots\}$	set of latent random variables (all possible classifications)
$\mathbb{Y} = \{y_1, y_2, \ldots\}$	set of observed random variables (all possible observations)
$\mathbb{L} = \{l_1, l_2, \ldots, l_k\}$	set of categories
$\mathbb{L}^x \dot{\cup} \mathbb{L}^z = \mathbb{L}$	sets of base layer– and occlusion layer categories, respectively
$\lvert\mathbb{L}\rvert \equiv k$	number of categories
$\mathbf{f}(\mathbf{y}) = (f_1, f_2, \ldots, f_m)^\top$	feature vector
$\lvert\mathbf{f}(\mathbf{y})\rvert \equiv m$	number of features
$\varphi(x)$	unary potential function
$\psi(x_1, x_2)$	(within-layer) pairwise potential function
$\xi(x_1, x_2)$	inter-layer pairwise potential function
$\mathcal{N}(\mathbf{y}) \equiv \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian function
G	number of Gaussian functions in a Gaussian mixture model
$\mathbb{E}(\mathbf{y}_1, \mathbf{y}_2)$	Euclidean distance
$\Delta(\mathbf{y}, \mathcal{N})$	Mahalanobis distance
$\mathbb{D}_{KL}(\mathcal{N}_1 \,\lvert\rvert\, \mathcal{N}_2)$	Kullback-Leibler divergence
$\mathbf{0}$	zero matrix, *i.e.* a matrix with all elemets equal to zero
$I \equiv diag(1, 1, \ldots, 1)$	identity matrix

L	loss matrix
$\boldsymbol{\theta}$	control parameters vector
$\mathcal{P}(\cdot)$	penalization function

B Graphical Models

B.1 Statistical Independence

Two random variables v_1 and v_2 are *statistically independent* (also *unconditional independent*) if and only if $p(v_1 \,|\, v_2) = p(v_1)$. As consequence we have:

$$p(v_1, v_2) = p(v_1 \,|\, v_2) \cdot p(v_2) = p(v_1) \cdot p(v_2). \tag{B.1.1}$$

B.2 Conditional Independence

Two random variables v_1 and v_2 are *conditional independent* given a third random variable v_3 if and only if $p(v_1 \,|\, v_2, v_3) = p(v_1 \,|\, v_3)$[1] [Daw80]. As consequence we have:

$$p(v_1, v_2 \,|\, v_3) = p(v_1 \,|\, v_2, v_3) \cdot p(v_2 \,|\, v_3) = p(v_1 \,|\, v_3) \cdot p(v_2 \,|\, v_3), \tag{B.2.1}$$

which must be hold for every possible value of v_3, and not just for some values.

B.3 General Product Rule

By application of the product rule of probability $p(v_i, v_j) = p(v_i \,|\, v_j) \cdot p(v_j)$, we can write the joint distribution for an arbitrary number n of random variables $\mathbf{v} = (v_1, \ldots, v_n)^\top$, in the form:

$$p(\mathbf{v}) = \prod_{i=1}^{n-1} p(v_i \,|\, v_{i+1}, \ldots, v_n) \cdot p(v_n); \text{ or} \tag{B.3.1}$$

$$p(\mathbf{v}) = p(v_1) \prod_{i=2}^{n} p(v_i \,|\, v_{i-1}, \ldots, v_1). \tag{B.3.2}$$

Note, that this decomposition holds for any choice of the joint distribution.

[1] This notion is equivalent to $v_1 \perp\!\!\!\perp v_2 \,|\, v_3$

B.4 Potential Approximation

unary potentials

$p(v_i)$	$\propto$	$\varphi(v_i)$
$p(\mathbf{v})$	$\propto$	$\prod_{i=1}^{n} \varphi(v_i)$

pairwise potentials

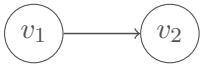
$p(v_i \mid v_j)$	$\propto$	$\varphi(v_i) \cdot \psi(v_i, v_j)$ [2]
$p(v_i \mid v_j, \ldots, v_m)$	$\propto$	$\varphi(v_i) \cdot \prod_{t=j}^{m} \psi(v_i, v_t)$
$p(v_i, v_j)$	$\propto$	$\varphi(v_i) \cdot \psi(v_i, v_j) \cdot \varphi(v_j)$ [3]
$p(\mathbf{v})$	$\propto$	$\prod_{i=1}^{n-1} \varphi(v_i) \prod_{j=i+1}^{n} \psi(v_i, v_j) \cdot \varphi(v_n)$

[2] Here we assume the pairwise potential function $\psi(v_i, v_j)$ to be an arbitrary (also non-symmetric) function.

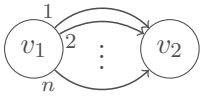
[3] Since the probability distribution $p(v_i, v_j)$ is symmetric, we assume here, that the pairwise potential function $\psi(v_i, v_j)$ holds the same property, *i.e.*: $\psi(v_i, v_j) \equiv \psi(v_j, v_i)$, $\forall i, j$.

B.5 Graph Construction Rules

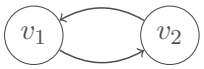
$$p(v_1, v_2) \overset{B.1.1}{\propto} \varphi(v_1) \cdot \varphi(v_2)$$



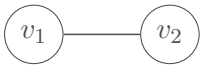
$$p(v_1 \,|\, v_2) \propto \varphi(v_1) \cdot \psi(v_1, v_2)$$



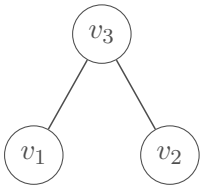
$$p(v_1 \,|\, v_2) \propto \varphi(v_1) \cdot \psi(v_1, v_2)^n$$



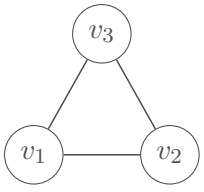
$$\begin{aligned} p(v_1 \,|\, v_2) \cdot p(v_2 \,|\, v_1) &\propto \varphi(v_1) \cdot \psi(v_1, v_2) \cdot \varphi(v_2) \cdot \psi(v_2, v_1) \\ &= \varphi(v_1) \cdot \psi(v_1, v_2) \cdot \psi(v_1, v_2)^\top \cdot \varphi(v_2) \end{aligned}$$



$$p(v_1, v_2) \;\propto\; \varphi(v_1) \;\cdot\; \psi(v_1, v_2) \;\cdot\; \varphi(v_2)$$



$$\begin{aligned} p(v_1, v_2, v_3) &= p(v_3 \,|\, v_2, v_1) \cdot p(v_2, v_3) \\ &= \frac{p(v_1, v_2 \,|\, v_3) \cdot p(v_3)}{p(v_2, v_3)} \cdot p(v_2, v_3) \\ &\overset{B.2.1}{=} p(v_1 \,|\, v_3) \cdot p(v_2 \,|\, v_3) \cdot p(v_3) \\ &\propto \varphi(v_1) \cdot \psi(v_1, v_3) \cdot \varphi(v_2) \cdot \psi(v_2, v_3) \cdot \varphi(v_3) \end{aligned}$$



$$\begin{aligned} p(v_1, v_2, v_3) &\overset{B.3.1}{=} p(v_1 \,|\, v_2, v_3) \cdot p(v_2 \,|\, v_3) \cdot p(v_3) \\ &\propto \varphi(v_1) \cdot \psi(v_1, v_2) \cdot \psi(v_1, v_3) \cdot \varphi(v_2) \cdot \psi(v_2, v_3) \cdot \varphi(v_3) \end{aligned}$$

B.6 Mixed Graphs

$\mathcal{G}^m$	u is a	$\mathcal{G}^m \left[\begin{smallmatrix}\{u\}\\ \emptyset\end{smallmatrix}\right.$	$\mathcal{G}^m \left[\begin{smallmatrix}\emptyset\\ \{u\}\end{smallmatrix}\right.$
$v_1 - u - v_2$	neighbor on the path	$v_1 - v_2$	$v_1 \quad v_2$
$v_1 \to u \leftarrow v_2$	manager on the path	$v_1 - v_2$	$v_1 \quad v_2$
$v_1 - u \leftarrow v_2$	manager on the path	$v_1 \leftarrow v_2$	$v_1 \quad v_2$
$v_1 \to u - v_2$	manager on the path	$v_1 \to v_2$	$v_1 \quad v_2$
$v_1 - u \to v_2$	follower on the path	$v_1 \quad v_2$	$v_1 \to v_2$
$v_1 \to u \to v_2$	follower on the path	$v_1 \quad v_2$	$v_1 \to v_2$
$v_1 \leftarrow u - v_2$	follower on the path	$v_1 \quad v_2$	$v_1 \leftarrow v_2$
$v_1 \leftarrow u \leftarrow v_2$	follower on the path	$v_1 \quad v_2$	$v_1 \leftarrow v_2$
$v_1 \leftarrow u \to v_2$	follower on the path	$v_1 \quad v_2$	$v_1 - v_2$

Tab. B.1: The set of examples of mixed graphs over three variables v_1, v_2 and u used to discuss conditional independence properties as well as marginalizing and conditioning for multi-layer graphs.

C Algorithms

C.1 Function *addPoint*

Algorithm 5: addPoint($\mathbf{f}$)

Data: Sample point $\mathbf{f}$

Result: Updated Gaussian function $\dot{\mathcal{N}}$

if $numPoints = 0$ **then**
  $\dot{\boldsymbol{\mu}} \leftarrow \mathbf{f}$;
  $\dot{\boldsymbol{\Sigma}} \leftarrow \mathbf{0}$;
else
  $\dot{\boldsymbol{\mu}} \leftarrow \frac{numPoints \cdot \boldsymbol{\mu} + \mathbf{f}}{numPoints+1}$;
  $\dot{\boldsymbol{\Sigma}} \leftarrow \frac{numPoints \cdot (\Sigma + \mu\mu^\top) + ff^\top}{numPoints+1} - \dot{\mu}\dot{\mu}^\top$;
$numPoints \leftarrow numPoints + 1$;

C.2 Function *distance*

Algorithm 6: distance($\mathcal{N}$, $\mathbf{f}$)

Data: Sample point $\mathbf{f}$; Gaussian function $\mathcal{N}$; minimal number of samples $\hat{N}_{min}$

Result: Distance dst

if $Mahalanobis\_dst_\theta$ **then**
  if $\mathcal{N}.numPoints \geq \hat{N}_{min}$ **then**
    $dst \leftarrow \Delta(\mathbf{f}, \mathcal{N})$;
  else
    $dst \leftarrow \frac{Mahalanobis\_dst_\theta}{Euclidian\_dst_\theta} \cdot \mathbb{E}(\mathbf{f}, \mathcal{N}.\boldsymbol{\mu})$;
else
  $dst \leftarrow \mathbb{E}(\mathbf{f}, \mathcal{N}.\boldsymbol{\mu})$;

C.3 Function *divergence*

Algorithm 7: divergence($\mathcal{N}_1$, $\mathcal{N}_2$)

Data: Gaussian functions $\mathcal{N}_1$ and $\mathcal{N}_2$; minimal number of samples $\hat{N}_{min}$

Result: Divergence *div*

```
if N1.numPoints ≥ N̂min then
    div ← 𝔻KL(N1,N2);
else
    div ← ∞;
```

D Datasets and Experimental Setup

D.1 Datasets

The methods, presented in this thesis were evaluated using 3 datasets from different areas of pattern recognition. The first dataset is the *Environmental Microorganism Data Set* (EMDS) [Zou+16], containing microscopic images from microbiology and used intensively in Chapter 2. Our novel method, presented in Chapter 3 requires test data to consist of two separate layers of class labels: one for the base and one for the occlusion layer. Thus we are limited by using only those datasets, which provide such information. So for evaluation we have chosen the following datasets: *Vaihingen* [Cra10] and the *StreetScene* [Bil06].

***Environmental Microorganism* Dataset** The EMDS dataset consists of 400 microscopic images of size 445×304 and contains 20 classes of environmental organisms (plus 1 *background* class) and each microorganism class is represented by 20 scenes. Here, we chose the nodes of the graphical model to correspond to single pixels. Thus, the graphical model for one layer and one scene consists of 445×304 nodes.

***Vaihingen* Dataset** The Vaihingen dataset consists of 2032 scenes with resolution of 250×250 pixels and *Ground Sampling Distance* (GSD) of 8 *cm*. Each scene is represented by a *Color Infra-Red* (CIR) image (*orthophoto*) and a height-map image (*Digital Surface Model*; DSM) generated with the variational stereo matching algorithm [KTS09] from wide baseline multiple overlapping airborne images. Both the CIR and the DSM images are defined on the same grid. Here, we also chose the nodes of the graphical model to correspond to single pixels.

***StreetScene* Dataset** The StreetScene dataset consists of 3544 color images with resolution of 1280×960 pixels. In this case data sites are represented by image patches of 5×5 pixels. So, the graphical model for one layer and one scene consists of 256×192 nodes.

D.2 Reference Data

The reference data for the EMDS dataset is given as a single layer map of labels, achieved by manually labeling the microorganism images. And the two-layer reference for the Vaihingen

	EMDS	Vaihingen	StreetScenes
number of scenes:	400	2032	3544
scene resolution [pixels]:	445×304	250×250	1280×960
downscale ratio:	$1:1$	$1:1$	$1:5$
layer resolution [nodes]:	445×304	250×250	256×192
red channel:	✓	✓	✓
green channel:	✓	–	✓
blue channel:	✓	✓	✓
infra-red channel:	–	✓	–
height map:	–	✓	–

Tab. D.1: Datasets overview.

and SteetScenes datasets is generated by labeling the images, using the assumption about the continuity of objects' shapes in occluded areas, to define the reference of the base layer.

***EMDS* Dataset** The reference of the EMDS dataset contains 20 classes of environmental microorganisms $\{\omega_1, \ldots, \omega_{20}\}$. In the thesis, for simplicity, a microorganism name is sometimes represented by the symbol ω_i $(1 \leq i \leq 20)$, as depicted in Fig. D.1. The background constitutes the 21-st class and covers $87,99$ % of the whole amount of samples, which is $55\,314\,923$. The percentages of the samples belonging to the microorganism classes varies from $0,06$ % for Epistylis till $1,56$ % for Arcella.

***Vaihingen* Dataset** The reference of the Vaihingen dataset has 6 classes: *asphalt*, *building*, *grass*, *agriculture*, *tree* and *car*, so that $\mathcal{L}^b = \{asphalt, building, grass, agriculture\}$ and $\mathcal{L}^o = \{tree, car, void\}$, where special class *void* represents the situation when there is no occlusions.

***StreetScene* Dataset** The reference of the StreetScenes dataset contains a reference in the form of polygons that also consider parts of hidden objects and hence could be used to define the two-layered reference. It has 9 classes: *road*, *sidewalk*, *building*, *store*, *tree*, *sky*, *car*, *pedestrian* and *bicycle*. The reference for this dataset is given by polygons, and it occurs that some image areas are not covered by any polygon. In order to keep our model consistent, we introduce here class *unknown* and mark with it all the uncovered areas at the base layer; at the occlusion layer, such areas are marked as *void*. During the evaluation class *unknown* was ignored. Moreover, in order to assess the presented methods more impartial, we merge classes *road* and *sidewalk* together into more general class *asphalt* and include class *store* into the class *building*, so that $\mathcal{L}^b = \{asphalt, building, sky, unknown\}$ and $\mathcal{L}^o = \{tree, car, bicycle, pedestrian, void\}$.

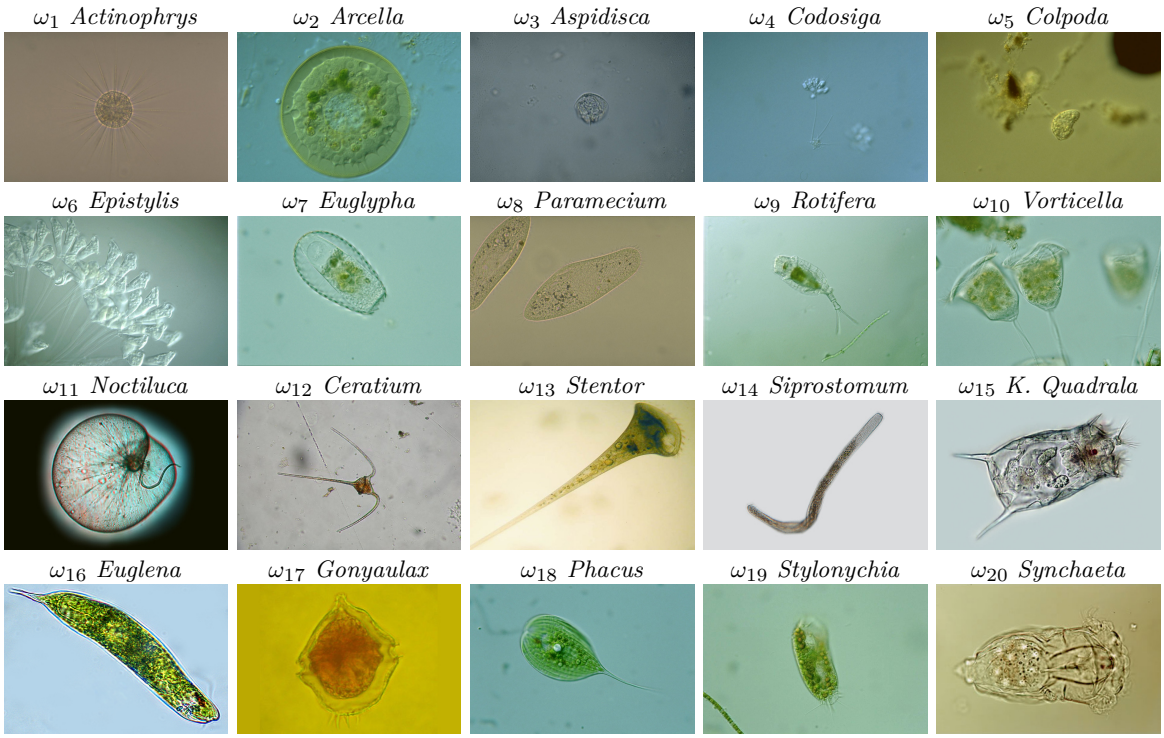


Fig. D.1: Examples of images in EMDS.

D.3 Features

From input data, we derive the site-wise feature vectors $\mathbf{f}_i(\mathbf{y})$, each consisting of m features. Particular site-wise feature vectors $\mathbf{f}_i(\mathbf{y})$, which represent the data in the association potentials and inter-layer interaction potentials, depend on the dataset. For numerical reasons, all features are scaled linearly into the range $[0;\ 255]$ and then quantized by 8 bit.

For the EMDS dataset we make use of pixel-level features extracted from DeepLab-VGG-16. For short, we call these 'DCNN' features and compare them to the following set of features: the intensity, calculated as the average of the red, blue and green channels; the saturation

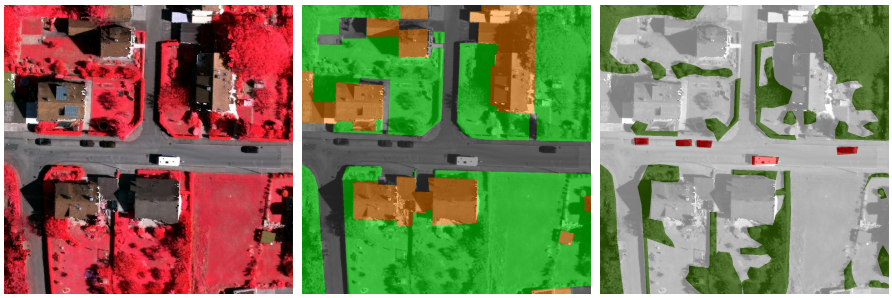


Fig. D.2: Example of a scene in Vaihingen dataset. **Left**: original color-infra-red image; **Center**: groundtruth for base layer; **Right**: groundtruth for occlusion layer.

	Category	Vaihingen	StreetScenes
base layer	*asphalt*	34,86 %	37,98 %
	building	17,51 %	25,75 %
	grass	38,76 %	–
	agriculture	8,87 %	–
	tree	–	11,32 %
	sky	–	4,57 %
	unknown	–	20,37 %
occl. layer	*tree*	18,07 %	–
	car	0,86 %	8,36 %
	bicycle	–	0,13 %
	pedestrian	–	0,51 %
	void	81,07 %	91,00 %
# of samples:		127 000 000	4 354 867 200

Tab. D.2: Categories overview. The percentage of samples belonging to different categories as well as the overall number of samples are given.

component in HSL color space. These two features are derived at 3 different scales: for the individual pixels and as the average in a local neighborhood of 15×15 and 25×25 pixels. Next we determine the variances of intensity, saturation and the gradient determined in local neighborhoods of 7×7, 15×15 and 25×25 pixels of each site. The last feature is the spacial coordinate feature, which describes the normalized distance of every pixel from the image center. This results in 16 features.

The Vaihingen data, based on aerial views, are available in a reference frame aligned with the North direction, which is not helpful to structure the scene because roads and buildings (the dominant objects in these data) are not necessarily aligned in North-South or East-West directions. As the original images were taken at the same flying height, all objects appear at a similar scale. On the other hand, for the StreetScenes data, the vertical (y coordinate axis) provides a physically defined reference direction that is clearly related to the scene structure. Furthermore, the distances at which objects are observed vary considerably, so that the scale of objects varies both within and between different scenes.

The features used for both Vaihingen and StreetScenes datasets comprise the image *intensity* (*int*), calculated as the average of non-infrared channels, the *saturation* (*sat*) component after transforming the image to the *Hue-Saturation-Lightness* (HSL) color space and the *gradient* (*grad*), achieved by applying to the image Sobel operator. We also make use of the *variance of intensity* (var_{int}), the *variance of saturation* (var_{sat}) and the *variance of gradient* (var_{grad}) determined from a local neighborhood of each site i (7×7 pixels for var_{int}, 13×13 pixels for var_{sat} and var_{grad}, in both cased evaluated at the original resolution).

For the Vaihingen dataset, we make use of the *Normalized Difference Vegetation Index* (NDVI), derived from the near infrared and the red band of the CIR orthophoto [Myn+95]. We also determine a *Digital Terrain Model* (DTM) by applying a morphological opening filter to the DSM with a structural element size corresponding to the size of the largest off-terrain structure in the scene, followed by a median filter with the same kernel size. The DTM is used to derive a *normalized* DSM (nDSM) [WF95], *i.e.* a model of the height differences between the DSM and DTM. The nDSM describes the relative elevation of objects above ground and its value at image site i is directly used as a feature. Finally, the feature *dist* models the fact that road pixels are usually found in a certain distance either from road edges or road markings. We generate an edge image by thresholding the intensity gradient of the input image. The *dist* feature is the distance of an image site to its nearest edge pixel. The last two features used for the Vaihingen data are the gradient strength of the DSM ($||\nabla DSM||$) and the car confidence feature (car).

The StreetScenes dataset has no infra-red channel and no height map. Nevertheless the classes in images of this dataset have a strong dependency on the image y-coordinate (*ordinate*) that reflects the vertical structure of the scenes. For instance, the sky is usually above road and buildings have vertical structure [YF11]. Consequently, we use the ordinate of a node as a feature. We make use of *Histogram of Oriented Gradients* (HOG) features [DT05] for the StreetScenes dataset. We calculate the HOG descriptors for cells consisting of 7×7 pixels, using blocks of 2×2 cells for normalization. Each histogram consists of 9 orientation bins (20° per bin). The gradient directions are determined relative to the vertical image axis. We extract nine features from the HOG descriptor, namely the value corresponding to each direction bin ($HOG_0, HOG_1, \ldots, HOG_9$).

For both datasets we make use of multiscale features. That is, the features described above are derived at three different scales. The first scale corresponds to the individual sites, the second and the third are calculated as the average in a local neighborhoods. For *int*, *sat*, *NDVI* and *nDSM*, these neighborhoods were chosen to be 21×21 and 49×49 pixels for the second and the third scales, respectively. For var_{int}, var_{sat}, var_{grad}, *dist*, $||\nabla DSM||$ and the HOG features the neighborhoods were chosen to be 10×10 and 100×100 pixels for scales two and three, respectively. In order to designate the scale at which a feature was extracted we use subscripts after the feature name with the scale (*e.g.* $NDVI_{21}, nDSM_{49}$, *etc.*).

Feature	EMDS	Vaihingen	StreetScenes
intensity	✓	✓	✓
saturation	✓	✓	✓
gradient	–	✓	✓
variance of intensity	✓	✓	✓
variance of saturation	✓	✓	✓
variance of gradient	✓	✓	✓
NDVI	–	✓	–
nDSM	–	✓	–
dist	–	✓	–
$\|\|\nabla DSM\|\|$	–	✓	–
car confidence	–	✓	–
coordinate	✓	–	✓
HOG	–	–	✓
DCNN	✓	–	–

Tab. D.3: Features overview.

Bibliography

[AE14] Selen Ayas and Murat Ekinci. "Random Forest-based Tuberculosis Bacteria Classification in Images of ZN-stained Sputum Smear Samples." In: *Signal Image Video Processing* 8.1 (2014), pp. 49–61 (Cited on page 19).

[Alt92] N. S. Altman. "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression." In: *The American Statistician* 46.3 (1992), pp. 175–185 (Cited on pages 15, 24, 55).

[And+03] Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. "An Introduction to MCMC for Machine Learning." In: *Machine Learning* 50.1 (Jan. 2003), pp. 5–43 (Cited on page 63).

[ASP11] Stavros Alchatzidis, Aristeidis Sotiras, and Nikos Paragios. "Efficient parallel message computation for MAP inference." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE Computer Society, 2011, pp. 1379–1386 (Cited on page 132).

[Atk89] K. Atkinson. *An Introduction to Numerical Analysis*. Wiley, 1989 (Cited on page 67).

[AW66] Mordecai Avriel and Douglass J. Wilde. "Optimality proof for the symmetric Fibonacci search technique." In: *The Fibonacci Quarterly*. 4 (1966), pp. 265–269 (Cited on page 67).

[AY06] Javed A. Aslam and Emine Yilmaz. "Inferring Document Relevance via Average Precision." In: *Proceedings of SIGIR*. ACM, 2006, pp. 601–602 (Cited on page 69).

[Bau+13] Stefan Bauer, Huanxiang Lu, Christian May, Lutz-Peter Nolte, Philippe Büchler, and Mauricio Reyes. "Integrated segmentation of brain tumor images for radiotherapy and neurosurgery." In: *International Journal of Imaging Systems and Technology* 23.1 (2013), pp. 59–63 (Cited on page 2).

[BCV13] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. "Representation Learning: A Review and New Perspectives." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 35.8 (2013), pp. 1798–1828 (Cited on page 16).

[Bil06] Stanley Michael Bileschi. "StreetScenes: Towards scene understanding in still images." PhD thesis. Massachusetts Institute of Technology, 2006 (Cited on page 145).

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. 1^{st}. Springer, 2006 (Cited on page 34).

[Bis95] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., 1995 (Cited on page 11).

[BJ01] Yuri Boykov and Marie-Pierre Jolly. "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. Vol. I. 2001, pp. 105–112 (Cited on pages 19, 20, 60).

[BK08] Gary R. Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with The OpenCV Library*. O'Reilly, 2008, pp. I–XVII, 1–555 (Cited on pages 24, 77, 79).

[BKC15] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. *SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation*. http://arxiv.org/abs/1511.00561. arXiv:1511.00561. 2015 (Cited on page 17).

[Bla+04] Andrew Blake, Carsten Rother, Matthew Brown, Patrick Perez, and Philip Torr. "Interactive Image Segmentation Using an Adaptive GMMRF Model." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2004, pp. 428–441 (Cited on page 19).

[BM12] Iain Brown and Christophe Mues. "An experimental comparison of classification algorithms for imbalanced credit scoring data sets." In: *Expert Systems with Applications* 39.3 (2012), pp. 3446–3453 (Cited on page 3).

[Bre01] Leo Breiman. "Random Forests." In: *Machine Learning* 45 (1 2001), pp. 5–32 (Cited on pages 24, 58).

[Bre96] Leo Breiman. *Bias, variance, and arcing classifiers*. Tech. rep. 460. Statistics Department, University of California at Berkeley, 1996 (Cited on page 131).

[BSC08] Dhruv Batra, Rahul Sukthankar, and Tsuhan Chen. "Learning class-specific affinities for image labelling." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008 (Cited on pages 19, 20).

[CG01] Giuseppe Calafiore and Laurent El Ghaoui. "Robust maximum likelihood estimation in the linear model." In: *Automatica* 37.4 (2001), pp. 573–580 (Cited on page 10).

[CGH97] Enrique Castillo, Jose M. Gutiérrez, and Ali S. Hadi. *Expert Systems and Probabilistic Network Models*. Monographs in Computer Science Series. Springer Verlag, 1997 (Cited on page 31).

[Cha+14] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. *Return of the Devil in the Details: Delving Deep into Convolutional Nets*. http://arxiv.org/abs/1405.3531. arXiv:1405.3531. 2014 (Cited on page 13).

[Cha+94] Pierre Charbonnier, Laure Blanc-Feraud, Gilles Aubert, and Michel Barlaud. "Two Deterministic Half-Quadratic Regularization Algorithms for Computed Imaging." In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. 1994, pp. 168–172 (Cited on page 61).

[Che+09] Minmin Chen, Yixin Chen, Michael R. Brent, and Aaron E. Tenney. "Constrained Optimization for Validation-guided Conditional Random Field Learning." In: *Proceedings of the 15^{th} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, 2009, pp. 189–198 (Cited on page 3).

[Che+16] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2016), pp. 1–1 (Cited on pages 13, 17, 39, 40, 82, 84, 88).

[CLY15] Yi-Ting Chen, Xiaokai Liu, and Ming-Hsuan Yang. "Multi-instance object segmentation with occlusion handling." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3470–3478 (Cited on page 22).

[Cor+01] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. 2nd. McGraw-Hill Higher Education, 2001 (Cited on page 64).

[Cow+07] Robert G. Cowell, A. Philip Dawid, Steffen L. Lauritzen, and David J. Spiegelhalter. *Probabilistic Networks and Expert Systems: Exact Computational Methods for Bayesian Networks*. 1^{st}. Springer, 2007 (Cited on pages 11, 31).

[Cra10] Michael Cramer. "The DGPF test on digital aerial camera evaluation - overview and test design." In: *Photogrammetrie-Fernerkundung-Geoinformation* 2.2010 (2010), pp. 73–82 (Cited on page 145).

[CS99] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines And Other Kernel-based Learning Methods*. Cambridge University Press, 1999 (Cited on pages 11, 56).

[CSK13] Neill D. F. Campbell, Kartic Subr, and Jan Kautz. "Fully-Connected CRFs with Non-Parametric Pairwise Potential." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2013, pp. 1658–1665 (Cited on page 19).

[Daw80] A. Philip Dawid. "Conditional Independence for Statistical Operations." In: *The Annals of Statistics* 8.3 (May 1980), pp. 598–617 (Cited on page 139).

[DeC+07] David DeCaprio, Jade P. Vinson, Matthew D. Pearson, Philip Montgomery, Matthew Doherty, and James E. Galagan. "Conrad: Gene prediction using conditional random fields." In: *Genome Research* 17.9 (2007), pp. 1389–1396 (Cited on page 2).

[Den+09] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "ImageNet: A large-scale hierarchical image database." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 248–255 (Cited on pages 25, 40).

[DH72] Richard O. Duda and Peter E. Hart. "Use of the Hough Transformation to Detect Lines and Curves in Pictures." In: *Commun. ACM* 15.1 (Jan. 1972), pp. 11–15 (Cited on pages 38, 106).

[DK02] Guilherme N. DeSouza and Avinash C. Kak. "Vision for Mobile Robot Navigation: A Survey." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 24.2 (2002), pp. 237–267 (Cited on page 3).

[DLR77] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm." In: *Journal of the Royal Statistical Society: Series B* 39.1 (1977), pp. 1–38 (Cited on pages 15, 38, 49).

[DP97] Pedro Domingos and Michael J. Pazzani. "On the Optimality of the Simple Bayesian Classifier under Zero-One Loss." In: *Machine Learning* 29.2-3 (1997), pp. 103–130 (Cited on pages 21, 43).

[DT05] Navneet Dalal and Bill Triggs. "Histograms of Oriented Gradients for Human Detection." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. IEEE, June 2005, pp. 886–893 (Cited on page 149).

[Eve+15] Mark Everingham, S. M. Eslami, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. "The Pascal Visual Object Classes Challenge: A Retrospective." In: *International Journal of Computer Vision (IJCV)* 111.1 (2015), pp. 98–136 (Cited on page 41).

[Faw06] Tom Fawcett. "An introduction to ROC analysis." In: *Pattern Recognition Letters* 27.8 (2006), pp. 861–874 (Cited on page 11).

[Fea04] Paul Fearnhead. "Particle filters for mixture models with an unknown number of components." In: *Statistics and Computing* 14.1 (2004), pp. 11–21 (Cited on page 18).

[FGM05] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling." In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 2005, pp. 363–370 (Cited on page 15).

[FH04] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. "Efficient Graph-Based Image Segmentation." In: *International Journal of Computer Vision (IJCV)* 59.2 (Sept. 2004), pp. 167–181 (Cited on pages 19, 20).

[Fil+15] Cicero F. F. C. Filho, Pamela C. Levy, Clahildek D. M. Xavier, Luciana B. M. Fujimoto, and Marly G. F. Costa. "Automatic identification of tuberculosis mycobacterium." In: *Research on Biomedical Engineering* 31 (Mar. 2015), pp. 33–43 (Cited on page 19).

[Fle87] Roger Fletcher. *Practical Methods of Optimization*. 2nd. Wiley, 1987 (Cited on page 18).

[FM98] Brendan J Frey and David J. C. MacKay. "A Revolution: Belief Propagation in Graphs with Cycles." In: *Advances in Neural Information Processing Systems 10*. Ed. by M. I. Jordan, M. J. Kearns, and S. A. Solla. MIT Press, 1998, pp. 479–485 (Cited on page 64).

[FM99] Yoav Freund and Llew Mason. "The Alternating Decision Tree Learning Algorithm." In: *Proceedings of the International Conference on Machine Learning (ICML)*. Morgan Kaufmann Publishers, 1999, pp. 124–133 (Cited on pages 11, 18, 19).

[Fou+11] James Foulds, Nicholas Navaroli, Padhraic Smyth, and Alexander Ihler. "Revisiting MAP Estimation, Message Passing and Perfect Graphs." In: *International Conference on AI and Statistics*. JMLR: W&CP 15, Apr. 2011, pp. 278–286 (Cited on page 66).

[Fry90] Morten Frydenberg. "The chain graph Markov property." In: *Scandinavian Journal of Statistics* 17 (1990), pp. 333–353 (Cited on page 23).

[GH12] Ruiqi Guo and Derek Hoiem. "Beyond the Line of Sight: Labeling the Underlying Surfaces." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Vol. 7576. Lecture Notes in Computer Science. Springer, 2012, pp. 761–774 (Cited on pages 23, 119).

[Gir+14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 580–587 (Cited on page 13).

[GNK10] Kapil Kumar Gupta, Baikunth Nath, and Ramamohanarao Kotagiri. "Layered Approach Using Conditional Random Fields for Intrusion Detection." In: *IEEE Transactions on Dependable and Secure Computing* 7.1 (Jan. 2010), pp. 35–49 (Cited on page 3).

[Gra+08] Helmut Grabner, Thuy Thi Nguyen, Barbara Gruber, and Horst Bischof. "Online boosting-based car detection from aerial images." In: *ISPRS Journal Photogrammetry and Remote Sensing* 63.3 (2008), pp. 382–396 (Cited on page 16).

[Gu+17] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, and G. Wang. "Recent Advances in Convolutional Neural Networks." In: *Pattern Recognition* (2017), Available online (Cited on page 12).

[Gut+14] E. Gutzeit, C. Scheel, T. Dolereit, and M. Rust. "Contour Based Split and Merge Segmentation and Pre-classification of Zooplankton in Very Large Images." In: *Proceedings of VISAPP*. 2014, pp. 417–424 (Cited on page 19).

[GZT11] Steve Gu, Ying Zheng, and Carlo Tomasi. "Extended pairwise potentials." In: *In CVPR Workshop on Inference in Graphical Models with Structured Potentials*. 2011 (Cited on pages 19, 20).

[HB03] Stefan Hinz and Albert Baumgartner. "Automatic extraction of urban road networks from multi-view aerial imagery." In: *ISPRS Journal Photogrammetry and Remote Sensing* 58 (1-2 2003), pp. 83–98 (Cited on page 12).

[HBS06] Stefan Hinz, Richard Bamler, and Uwe Stilla. “Theme issue: ”Airborne and spaceborne traffic monitoring”.” In: *ISPRS Journal Photogrammetry and Remote Sensing* 61.3-4 (2006), pp. 135–136 (Cited on page 17).

[He+16] K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778 (Cited on page 41).

[HG09] Haibo He and Edwardo A. Garcia. “Learning from Imbalanced Data.” In: *IEEE Transactions on Knowledge and Data Engineering* 21.9 (Sept. 2009), pp. 1263–1284 (Cited on page 70).

[HG10] Geremy Heitz and Chechik Gal. “Object separation in X-Ray image sets.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010, pp. 2093–2100 (Cited on page 23).

[HL02] Chih-Wei Hsu and Chih-Jen Lin. “A comparison of methods for multiclass support vector machines.” In: *IEEE Transactions on Neural Networks* 13.2 (2002), pp. 415–425 (Cited on page 57).

[HZC04] Xuming He, Richard S. Zemel, and Miguel Á. Carreira-Perpiñán. “Multiscale Conditional Random Fields for Image Labeling.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2004, pp. 695–703 (Cited on pages 2, 11, 22).

[Jen96] Finn V. Jensen. *Introduction to Bayesian Networks*. 1st. Springer, 1996 (Cited on pages 11, 31).

[Jia+14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. “Caffe: Convolutional Architecture for Fast Feature Embedding.” In: *Proceedings of ACM MM*. 2014, pp. 675–678 (Cited on page 88).

[Joa05] Thorsten Joachims. “A support vector method for multivariate performance measures.” In: *Proceedings of the International Conference on Machine Learning (ICML)*. ACM Press, 2005, pp. 377–384 (Cited on page 21).

[Jor+99] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. “An Introduction to Variational Methods for Graphical Models.” In: *Machine Learning* 37.2 (Nov. 1999), pp. 183–233 (Cited on page 63).

[Jor99] Michael Irwin Jordan. *Learning in Graphical Models*. Cambridge, MA, USA: MIT Press, 1999 (Cited on page 31).

[KFL06] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. "Factor Graphs and the Sum-product Algorithm." In: *IEEE Transactions on Information Theory* 47.2 (Sept. 2006), pp. 498–519 (Cited on page 35).

[KH03] Sanjiv Kumar and Martial Hebert. "Discriminative Fields for Modeling Spatial Dependencies in Natural Images." In: *Proceedings of the Neural Information Processing Systems Conference (NIPS)*. MIT Press, 2003, pp. 1531–1538 (Cited on page 11).

[KH05] Sanjiv Kumar and Martial Hebert. "A Hierarchical Field Framework for Unified Context-Based Classification." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. Vol. 2. IEEE Computer Society, 2005, pp. 1284–1291 (Cited on page 22).

[KH06] Sanjiv Kumar and Martial Hebert. "Discriminative Random Fields." In: *International Journal of Computer Vision (IJCV)* 68.2 (2006), pp. 179–201 (Cited on pages 8, 42, 59, 63).

[KHD11] Aniruddha Kembhavi, David Harwood, and Larry S. Davis. "Vehicle Detection Using Partial Least Squares." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 33.6 (2011), pp. 1250–1265 (Cited on page 16).

[KK11] Philipp Krähenbühl and Vladlen Koltun. "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials." In: *Proceedings of the Neural Information Processing Systems Conference (NIPS)*. 2011, pp. 109–117 (Cited on pages 20, 82, 89).

[KKS13] Byung-Soo Kim, Pushmeet Kohli, and Silvio Savarese. "3D Scene Understanding by Voxel-CRF." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2013, pp. 1425–1432 (Cited on page 23).

[KLT08] Pushmeet Kohli, Lubor Ladicky, and Philip H. S. Torr. "Robust higher order potentials for enforcing label consistency." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2008, pp. 1–8 (Cited on pages 19, 20).

[KM01] Svetlana Kiritchenko and Stan Matwin. "Email Classification with Co–training." In: *Proceedings of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research*. IBM Press, 2001, pp. 192–201 (Cited on page 44).

[Kol06] Vladimir Kolmogorov. "Convergent Tree-Reweighted Message Passing for Energy Minimization." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 28.10 (Oct. 2006), pp. 1568–1583 (Cited on page 66).

[Kra10] Oliver Kramer. “Iterated local search with Powell’s method: a memetic algorithm for continuous global optimization.” In: *Memetic Computing* 2.1 (2010), pp. 69–83 (Cited on page 66).

[Kru+16] Michal Kruk, Ryszard Kozera, Stanislaw Osowski, Trzcinski Pawel, Lidia Sas-Paszt, Beata Sumorok, and Boleslaw Borkowski. “Computerized Classification System for the Identification of Soil Microorganisms.” In: *Applied Mathematics & Information Sciences* 10.1 (2016), pp. 21–31 (Cited on pages 16, 19).

[KS08] Saurabh Kumar and Gauri S. Mittal. “Geometric and Optical Characteristics of Five Microorganisms for Rapid Detection Using Image Processing.” In: *Biosystems Engineering* 99.1 (2008), pp. 1–8 (Cited on page 16).

[KS09] Saurabh Kumar and Gauri S. Mittal. “Textural Characteristics of Five Microorganisms for Rapid Detection Using Image Processing.” In: *Journal of Food Process Engineering* 32.1 (2009), pp. 126–143 (Cited on page 16).

[KS10] Saurabh Kumar and Gauri S. Mittal. “Rapid Detection of Microorganisms Using Image Processing Parameters and Neural Network.” In: *Food and Bioprocess Technology* 3.5 (2010), pp. 741–751 (Cited on pages 16, 19).

[KSG18a] Lukas Köping, Kimiaki Shirahama, and Marcin Grzegorzek. “A general framework for sensor-based human activity recognition.” In: *Computers in Biology and Medicine* (2018) (Cited on page 3).

[KZ01] Vladimir Kolmogorov and Ramin Zabih. “Computing visual correspondence with occlusions using graph cuts.” In: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2001, pp. 508–515 (Cited on page 131).

[Lau96] Steffen L. Lauritzen. *Graphical Models*. Oxford University Press, 1996 (Cited on page 31).

[LD05] Daniel Lowd and Pedro Domingos. “Naive Bayes models for probability estimation.” In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 119. ACM Press, 2005, pp. 529–536 (Cited on pages 10, 18, 24).

[Lew90] David D. Lewis. “Representation Quality in Text Classification: An Introduction and Experiment.” In: *Proceedings of the Workshop on Speech and Natural Language*. Morgan Kaufmann Publishers, 1990, pp. 288–295 (Cited on page 119).

[Li+13] Chen Li, Kimiaki Shirahama, Marcin Grzegorzek, F. Ma, and B. Zhou. “Classification of Environmental Microorganisms in Microscopic Images Using Shape Features and Support Vector Machines.” In: *Proceedings of the IEEE Interna-*

tional Conference on Image Processing (ICIP). 2013, pp. 2435–2439 (Cited on page 16).

[Li09] Stan Z. Li. *Markov Random Field Modeling in Image Analysis*. 3rd. Springer, 2009 (Cited on pages 7, 11).

[Lis+05] Dimitri A. Lisin, Marwan A. Mattar, Matthew B. Blaschko, Mark C. Benfield, and Erik G. Learned-Miller. "Combining local and global image features for object class recognition." In: *Proceedings of CVPR Workshop*. 2005, pp. 47–55 (Cited on page 21).

[LKF16] Chen Liu, Pushmeet Kohli, and Yasutaka Furukawa. "Layered Scene Decomposition via the Occlusion-CRF." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 165–173 (Cited on page 23).

[Llo82] Stuart P. Lloyd. "Least Squares Quantization in PCM." In: *IEEE Transactions on Information Theory* 28.2 (Sept. 1982), pp. 129–137 (Cited on page 49).

[LLS08] Bastian Leibe, Aleš Leonardis, and Bernt Schiele. "Robust Object Detection with Interleaved Categorization and Segmentation." In: *International Journal of Computer Vision (IJCV)* 77 (1-3 2008), pp. 259–289 (Cited on page 23).

[LLW07] Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C. Weng. "A note on Platt's probabilistic outputs for support vector machines." In: *Machine Learning* 68.3 (2007), pp. 267–276 (Cited on pages 18, 19, 57).

[LMB02] Gildas Lefaix, Éric Marchand, and Patrick Bouthemy. "Motion-Based Obstacle Detection and Tracking for Car Driving Assistance." In: *Proceedings of the International Conference on Pattern Recognition (ICPR)*. Vol. 4. 2002, pp. 74–77 (Cited on page 3).

[LMP01] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data." In: *Proceedings of the International Conference on Machine Learning (ICML)*. Morgan Kaufmann Publishers, 2001, pp. 282–289 (Cited on pages 3, 5–7, 11, 15, 34).

[LOL09] Wei Lwun Lu, Kenji Okuma, and James J. Little. "A Hybrid Conditional Random Field for Estimating the Underlying Ground Surface from Airborne LiDAR Data." In: *IEEE Transactions on Geoscience and Remote Sensing* 47.8/2 (2009), pp. 2913–2922 (Cited on pages 2, 22).

[Low99] David G. Lowe. “Object Recognition from Local Scale-Invariant Features.” In: *Proceedings of the International Conference on Computer Vision (ICCV)*. 1999, pp. 1150–1158 (Cited on page 70).

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully Convolutional Networks for Semantic Segmentation.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3431–3440 (Cited on pages 17, 40, 82, 89).

[LSG15a] Chen Li, Kimiaki Shirahama, and Marcin Grzegorzek. “Environmental Microbiology Aided by Content-based Image Analysis.” In: *Pattern Analysis and Applications* 19.2 (2015), pp. 531–547 (Cited on page 16).

[LSG15b] Chen Li, Kimiaki Shirahama, and Marcin Grzegorzek. “Environmental Microorganism Classification Using Sparse Coding and Weakly Supervised Learning.” In: *Proceedings of EMR Workshop in ICMR*. ACM, 2015, pp. 9–14 (Cited on pages 86, 87).

[LW89] Steffen L. Lauritzen and Nanny Wermuth. “Graphical Models for Associations between Variables, some of which are Qualitative and some Quantitative.” In: *The Annals of Statistics* 17.1 (Mar. 1989), pp. 31–57 (Cited on page 23).

[LYT11] Ce Liu, Jenny Yuen, and Antonio Torralba. “Nonparametric Scene Parsing via Label Transfer.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 33.12 (2011), pp. 2368–2382 (Cited on page 23).

[Mac06] D. J.C. MacKay. “Good Error-correcting Codes Based on Very Sparse Matrices.” In: *IEEE Transactions on Information Theory* 45.2 (Sept. 2006), pp. 399–431 (Cited on page 64).

[Mat+16] Yasuyuki Matsumoto, Takashi Shinozaki, Kimiaki Shirahama, Marcin Grzegorzek, and Kuniaki Uehara. “Kobe University, NICT and University of Siegen on the TRECVID 2016 AVS Task.” In: *Proceedings of TRECVID*. 2016, pp. 1–8 (Cited on page 40).

[Mat17] John H Mathews. *Module for Powell Search Method for a Minimum*. http://mathfaculty.fullerton.edu/mathews/n2003/PowellMethodMod.html. 2017 (Cited on page 68).

[May+06] Helmut Mayer, Stefan Hinz, Uwe Bacher, and Emmanuel Baltsavias. “A test of automatic road extraction approaches.” In: *Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. 2006, pp. 209–214 (Cited on pages 14, 119).

[MB88] Geoffrey J. McLachlan and Kaye E. Basford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, 1988 (Cited on pages 15, 48).

[MK08] Geoffrey J. McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*. 2nd. Wiley series in probability and statistics. Wiley, 2008 (Cited on pages 15, 18, 49, 50).

[MN89] Peter McCullagh and John A. Nelder. *Generalized Linear Models*. Chapman & Hall, 1989 (Cited on page 10).

[MN99] Jan R. Magnus and Heinz Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. Wiley Series in Probability and Statistics: Texts and References Section. Wiley, 1999 (Cited on page 47).

[MP00] Geoffrey J. McLachlan and David Peel. *Finite mixture models*. Wiley Series in Probability and Statistics. Wiley, 2000 (Cited on pages 10, 15, 48).

[Myn+95] Ranga B. Myneni, Forrest G. Hall, Piers J. Sellers, and Alexander L. Marshak. "The interpretation of spectral vegetation indexes." In: *IEEE Transactions on Geoscience and Remote Sensing* 33 (2 1995), pp. 481–486 (Cited on page 149).

[NHB03] W. Nah, S. Hong, and J. Baek. "Feature Extraction for Classification of *Caenorhabditis Elegans* Behavioural Phenotypes." In: *Developments in Applied Artificial Intelligence*. Germany: Springer, 2003, pp. 287–295 (Cited on page 16).

[NSJ15] D. Nie, E. A. Shank, and V. Jojic. "A Deep Framework for Bacterial Image Segmentation and Classification." In: *Proceedings of ACM-BCB*. 2015, pp. 306–314 (Cited on pages 16, 17, 19).

[Nur+16] Jorge René Nuricumbo, Haider Ali, Zoltán-Csaba Márton, and Marcin Grzegorzek. "Improving object classification robustness in RGB-D using adaptive SVMs." In: *Multimedia Tools and Applications* 75.12 (June 2016), pp. 6829–6847 (Cited on page 23).

[NW06] Jorge Nocedal and Steve J. Wright. *Numerical optimization*. 2nd. Springer Series in Operations Research and Financial Engineering. Springer, 2006 (Cited on page 18).

[OM99] David Optiz and Richard Maclin. "Popular Ensemble Methods: An Empirical Study." In: *Journal of Artificial Intelligence Research* 11 (1999), pp. 169–198 (Cited on page 131).

[Ope14] OpenCV. *Machine Learning*. http://docs.opencv.org/modules/ml/doc/ml.html. Apr. 2014 (Cited on pages 89, 129).

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers, 1988 (Cited on pages 63, 98).

[PGG15] Ian L. Pepper, Charles P. Gerba, and Terry J. Gentry, eds. *Environmental Microbiology*. 3-rd. Academic Press, 2015 (Cited on page 3).

[PK12] Patrick Pletscher and Pushmeet Kohli. "Learning Low-order Models for Enforcing High-order Statistics." In: *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS)*. JMLR: W&CP 22, 2012, pp. 886–894 (Cited on pages 15, 21).

[Pla99] John C. Platt. "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods." In: *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74 (Cited on pages 18, 19, 57).

[PM90] Pietro Perona and Jitendra Malik. "Scale-space and edge detection using anisotropic diffusion." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 12 (June 1990), pp. 629–639 (Cited on page 61).

[Por05] Fatih Porikli. "Integral histogram: A fast way to extract histograms in cartesian spaces." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. IEEE. 2005, pp. 829–836 (Cited on page 42).

[Pow64] M. J. D. Powell. "An efficient method for finding the minimum of a function of several variables without calculating derivatives." In: *The Computer Journal* 7.2 (1964), pp. 155–162 (Cited on pages 66, 67).

[Pra+06] Mukta Prasad, Andrew Zisserman, Andrew Fitzgibbon, M. Pawan Kumar, and P. H. S. Torr. "Learning Class-specific Edges for Object Detection and Segmentation." In: *Proceedings of the 5th Indian Conference on Computer Vision, Graphics and Image Processing*. Springer, 2006, pp. 94–105 (Cited on pages 19, 20, 62).

[PWS14] Sansoen Promdaen, Pakaket Wattuya, and Nuttha Sanevas. "Automated Microalgae Image Classification." In: *Procedia Computer Science* 29.Supplement C (2014). 2014 International Conference on Computational Science, pp. 1981–1992 (Cited on pages 16, 19).

[QCD04] Ariadna Quattoni, Michael Collins, and Trevor Darrell. "Conditional Random Fields for Object Recognition." In: *Advances in Neural Information Processing Systems (NIPS)*. 2004 (Cited on pages 2, 16, 17).

[Qui86] J. R. Quinlan. "Induction of decision trees." In: *Machine Learning* 1.1 (Mar. 1986), pp. 81–106 (Cited on page 58).

[Rab89] Lawrence R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition." In: *Proceedings of the IEEE*. Vol. 77. IEEE, 1989, pp. 257–286 (Cited on page 5).

[RB93] Martin Riedmiller and Heinrich Braun. "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm." In: *IEEE International Conference on Neural Networks*. 1993, pp. 586–591 (Cited on page 58).

[Rey09] Douglas A. Reynolds. "Gaussian Mixture Models." In: *Encyclopedia of Biometrics*. Springer, 2009, pp. 659–663 (Cited on pages 15, 24, 49).

[RFF06] Binyamin Rosenfeld, Ronen Feldman, and Moshe Fresko. "A Systematic Cross-Comparison of Sequence Classifiers." In: *Proceedings of the Siam Conference on Data Mining (SDM)*. SIAM, 2006, pp. 563–567 (Cited on page 11).

[RS02] Thomas Richardson and Peter Spirtes. "Ancestral graph Markov models." In: *The Annals of Statistics* 30.4 (Aug. 2002), pp. 962–1030 (Cited on pages 23, 98, 101).

[Sch+09] Paul Schnitzspan, Mario Fritz, Stefan Roth, and Bernt Schiele. "Discriminative structure learning of hierarchical representations for object detection." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 2238–2245 (Cited on pages 16, 22).

[Sch12] Konrad Schindler. "An Overview and Comparison of Smooth Labeling Methods for Land-Cover Classification." In: *IEEE Transactions on Geoscience and Remote Sensing* 50 (11 2012), pp. 4534–4545 (Cited on pages 11, 12).

[SG16] Kimiaki Shirahama and Marcin Grzegorzek. "Towards Large-Scale Multimedia Retrieval Enriched by Knowledge about Human Interpretation." In: *Multimedia Tools and Applications* 75.1 (2016), pp. 297–331 (Cited on page 16).

[SGU14] Kimiaki Shirahama, Marcin Grzegorzek, and Kuniaki Uehara. "Multimedia Event Detection Using Hidden Conditional Random Fields." In: *International Conference on Multimedia Retrieval*. Apr. 2014, pp. 9–16 (Cited on page 13).

[SGU15] Kimiaki Shirahama, Marcin Grzegorzek, and Kuniaki Uehara. "Weakly Supervised Detection of Video Events Using Hidden Conditional Random Fields." In: *International Journal on Multimedia Information Retrieval* 4.1 (2015), pp. 17–32 (Cited on page 15).

[Sho+09] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. "TextonBoost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context." In: *International Journal of Computer Vision (IJCV)* 81 (Jan. 2009), pp. 2–23 (Cited on page 66).

[SHR10] Hongyu Su, Markus Heinonen, and Juho Rousu. "Multilabel Classification of Drug-like Molecules via Max-Margin Conditional Random Fields." In: *Proceedings of the 5^{th} European Workshop on Probabilistic Graphical Models*. HIIT, 2010, pp. 265–272 (Cited on page 2).

[Sil+14] Nathan Silberman, Lior Shapira, Ran Gal, and Pushmeet Kohli. "A Contour Completion Model for Augmenting Surface Reconstructions." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2014, pp. 488–503 (Cited on page 106).

[SKH08] Martin Szummer, Pushmeet Kohli, and Derek Hoiem. "Learning CRFs Using Graph Cuts." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2008, pp. 582–595 (Cited on page 21).

[SM00] Jianbo Shi and Jitendra Malik. "Normalized Cuts and Image Segmentation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 22.8 (Aug. 2000), pp. 888–905 (Cited on pages 19, 20, 61).

[SM04] Charles Sutton and Andrew McCallum. "Collective Segmentation and Labeling of Distant Entities in Information Extraction." In: *Proceedings of the ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*. 2004 (Cited on page 15).

[SM07] Charles Sutton and Andrew McCallum. "An Introduction to Conditional Random Fields for Relational Learning." In: *Introduction to Statistical Relational Learning*. MIT Press, 2007 (Cited on page 15).

[SM12] Charles Sutton and Andrew McCallum. "An Introduction to Conditional Random Fields." In: *Foundations and Trends in Machine Learning* 4.4 (2012), pp. 267–373 (Cited on page 31).

[Soi03] Pierre Soille. *Morphological Image Analysis: Principles and Applications*. 2nd ed. Springer-Verlag New York, 2003 (Cited on page 38).

[Son+15] D. Song, W. Liu, T. Zhou, D. Tao, and D. A. Meyer. "Efficient Robust Conditional Random Fields." In: *IEEE Transctions on Image Processing* 24.10 (2015), pp. 3124–3136 (Cited on page 21).

[SP03] Fei Sha and Fernando Pereira. "Shallow Parsing with Conditional Random Fields." In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL)*. Vol. 1. Association for Computational Linguistics, 2003, pp. 134–141 (Cited on pages 3, 11).

[SP07] Daniel Scharstein and Chris Pal. "Learning Conditional Random Fields for Stereo." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2007 (Cited on page 131).

[ST10] Dongjin Song and Dacheng Tao. "Biologically Inspired Feature Manifold for Scene Classification." In: *IEEE Transctions on Image Processing* 19.1 (2010), pp. 174–184 (Cited on page 17).

[Ste97] Stephen V. Stehman. "Selecting and Interpreting Measures of Thematic Classification Accuracy." In: *Remote Sensing of Environment* 62.1 (1997), pp. 77–89 (Cited on page 66).

[SZ14] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. http://arxiv.org/abs/1409.1556. arXiv:1409.1556. 2014 (Cited on pages 25, 39, 40).

[TB10] Zhuowen Tu and Xiang Bai. "Auto-Context and Its Application to High-Level Vision Tasks and 3D Brain Image Segmentation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 32.10 (Oct. 2010), pp. 1744–1757 (Cited on page 23).

[TIS06] R. Tautenhahn, A. Ihlow, and U. Seiffert. "Adaptive Feature Selection for Classification of Microscope Images." In: *Fuzzy Logic and Applications*. Germany: Springer, 2006, pp. 215–222 (Cited on pages 16, 19).

[TNL14] Joseph Tighe, Marc Niethammer, and Svetlana Lazebnik. "Scene Parsing with Object Instances and Occlusion Ordering." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 3748–3755 (Cited on page 22).

[TWH07] Katrin Tomanek, Joachim Wermter, and Udo Hahn. "A Reappraisal of Sentence and Token Splitting for Life Sciences Documents." In: *Proceedings of the 12th World Congress on Medical Informatics*. Vol. 129. Studies in Health Technology and Informatics. IOS Press, 2007, pp. 524–528 (Cited on page 3).

[Ver+15] Antanas Verikas, Gelzinis. Adas, Marija Bacauskiene, Irina Olenina, and Evaldas Vaiciukynas. "An Integrated Approach to Analysis of Phytoplankton Images." In: *IEEE Journal of Oceanic Engineering* 40.2 (2015), pp. 315–326 (Cited on pages 16, 19).

[Vis+06] S. V. N. Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. "Accelerated training of conditional random fields with stochastic gradient methods." In: *Proceedings of the International Conference on Machine Learning (ICML)*. ACM Press, 2006, pp. 969–976 (Cited on page 63).

[VJ04] Paul Viola and Michael J. Jones. "Robust Real-Time Face Detection." In: *International Journal of Computer Vision (IJCV)* 57.2 (May 2004), pp. 137–154 (Cited on page 37).

[VP91] Thomas Verma and Judea Pearl. "Equivalence and Synthesis of Causal Models." In: *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*. UAI '90. Elsevier Science Publishers, 1991, pp. 255–270 (Cited on page 101).

[Wal69] A. M. Walker. "On the Asymptotic Behaviour of Posterior Distributions." In: *Journal of the Royal Statistical Society. Series B (Methodological)* 31.1 (1969) (Cited on page 46).

[Wer+11] Manuel Werlberger, Markus Unger, Thomas Pock, and Horst Bischof. "Efficient Minimization of the Non-Local Potts Model." In: *International Conference on Scale Space and Variational Methods in Computer Vision*. 2011 (Cited on pages 19, 60).

[WF95] Uwe Weidner and Wolfgang Förstner. "Towards Automatic Building Reconstruction from High Resolution Digital Elevation Models." In: *ISPRS Journal Photogrammetry and Remote Sensing* (1995), pp. 38–49 (Cited on page 149).

[Whi09] Joe Whittaker. *Graphical Models in Applied Multivariate Statistics*. Wiley Publishing, 2009 (Cited on page 31).

[WJW05] Martin J. Wainwright, Tommi Jaakkola, and Alan S. Willsky. "MAP estimation via agreement on trees: message-passing and linear programming." In: *IEEE Transactions on Information Theory* 51.11 (2005), pp. 3697–3717 (Cited on page 66).

[WS06] John Winn and Jamie Shotton. "The Layout Consistent Random Field for Recognizing and Segmenting Partially Occluded Objects." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2006, pp. 37–44 (Cited on page 22).

[Yan+10] Yi Yang, Sam Hallman, Deva Ramanan, and Charless Fowlkes. "Layered Object Detection for Multi-Class Segmentation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2010) (Cited on page 16).

[Yan+12] Yi Yang, Sam Hallman, Deva Ramanan, and Charless Fowlkes. "Layered Object Models for Image Segmentation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 34 (2012), pp. 1731–1743 (Cited on page 22).

[YC07] Zhaozheng Yin and Robert T. Collins. "Belief Propagation in a 3D Spatio-temporal MRF for Moving Object Detection." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2007 (Cited on page 22).

[YF11] Michael Ying Yang and Wolfgang Förstner. "Regionwise classification of building facade images." In: *Photogrammetric Image Analysis*. Vol. 6952. Lecture Notes in Computer Science. Springer, 2011, pp. 209–220 (Cited on page 149).

[YFW03] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. "Understanding Belief Propagation and Its Generalizations." In: *Exploring Artificial Intelligence in the New Millennium*. Ed. by Gerhard Lakemeyer and Bernhard Nebel. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, pp. 239–269 (Cited on page 64).

[YS08] Inbal Yahav and Galit Shmueli. *An Elegant Method for Generating Multivariate Poisson Data*. http://arxiv.org/abs/0710.5670v2. arXiv:0710.5670v2 [stat.CO]. 2008 (Cited on page 46).

[Yu+11] Bo Yang Yu, Caglar Elbuken, Carolyn L. Ren, and Jan Paul Huissoon. "Image processing and classification algorithm for yeast cell morphology in a microfluidic chip." In: *Journal of Biomedical Optics* 16.6 (2011), pp. 1–9 (Cited on page 19).

[ZAV07] Xinhua Zhang, Douglas Aberdeen, and S. V. N. Vishwanathan. "Conditional Random Fields for Multi-agent Reinforcement Learning." In: *Proceedings of the International Conference on Machine Learning (ICML)*. ACM Press, 2007, pp. 1143–1150 (Cited on page 3).

[Zhe+15] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. "Conditional Random Fields As Recurrent Neural Networks." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. Washington, DC, USA: IEEE Computer Society, 2015, pp. 1529–1537 (Cited on page 131).

[Zou+16] Yanling Zou, Chen Li, Kimiaki Shirahama, Tao Jiang, and Marcin Grzegorzek. "Environmental Microorganism Image Retrieval Using Multiple Colour Channels Fusion and Particle Swarm Optimisation." In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. 2016, pp. 2475–2479 (Cited on pages 19, 145).

Own Publications

[KSG18b] Sergey Kosov, Kimiaki Shirahama, and Marcin Grzegorzek. “Labeling of partially occluded regions via the multi-layer CRF.” In: *Multimedia Tools and Applications* (July 2018), pp. 1–19 (Cited on pages 24, 94, 104, 105, 113).

[Kos+18] Sergey Kosov, Kimiaki Shirahama, Chen Li, and Marcin Grzegorzek. “Environmental microorganism classification using conditional random fields and deep convolutional neural networks.” In: *Pattern Recognition* 77 (2018), pp. 248–261 (Cited on pages 13, 24, 25, 36, 38, 39, 70, 71, 86).

[Kos15] Sergey Kosov. *Direct Graphical Models C++ library.* http://research.project-10.de/dgm/. 2015 (Cited on pages 24, 67, 68, 71, 78, 88, 118, 131).

[Bra+14] Andreas Christian Braun, Carolina Rojas, Cristian Echeverri, Franz Rottensteiner, Hans-Peter Bähr, Joachim Niemeyer, Mauricio Aguayo Arias, Sergey Kosov, Stefan Hinz, and Uwe Weidner. “Design of a Spectral - Spatial Pattern Recognition Framework for Risk Assessments Using Landsat Data - A Case Study in Chile.” In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (JSTARS)* 7.1 (Mar. 2014), pp. 917–928 (Cited on page 24).

[Kos+13a] Sergey Kosov, Pushmeet Kohli, Franz Rottensteiner, and Christian Heipke. *A Two-Layer Conditional Random Field for The Classification of Partially Occluded Objects.* http://arxiv.org/abs/1307.3043. arXiv:1307.3043 [cs.CV]. 2013 (Cited on pages 24, 116, 118).

[KRH13] Sergey Kosov, Franz Rottensteiner, and Christian Heipke. “Sequential Gaussian Mixture Models for Two-Level Conditional Random Fields.” In: *Proceedings of the 35^{th} German Conference on Pattern Recognition (GCPR).* Vol. 8142. Lecture Notes in Computer Science. Springer, 2013, pp. 153–163 (Cited on pages 15, 24, 45, 50, 77).

[Kos+13b] Sergey Kosov, Franz Rottensteiner, Christian Heipke, Jens Leitloff, and Stefan Hinz. “The Application of a Car Confidence Feature for the Classification of Cross-Roads Using Conditional Random Fields.” In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* II-3/W3 (2013), pp. 43–48 (Cited on pages 14, 16, 24, 42, 71, 74).

[Kos+12] Sergey Kosov, Franz Rottensteiner, Chrisitan Heipke, Jens Leitloff, and Stefan Hinz. "3D Classification Of Crossroads From Multiple Aerial Images Using Markov Random Fields." In: *Proceedings of the 22nd ISPRS Congress*. 2012, pp. 479–484 (Cited on page 24).

[KRH12] Sergey Kosov, Franz Rottensteiner, and Christian Heipke. "3D Classification Of Crossroads From Multiple Aerial Images Using Conditional Random Fields." In: *IAPR Workshop on Pattern Recognition in Remote Sensing (PRRS)*. IEEE. 2012, pp. 1–4 (Cited on pages 19, 20, 24, 79, 82).

[KTS11] Sergey Kosov, Thorsten Thormählen, and Hans-Peter Seidel. "Using Active Illumination for Accurate Variational Space-Time Stereo." In: *Proceedings of the 17th Scandinavian Conference on Image Analysis (SCIA)*. Vol. 6688. Lecture Notes in Computer Science. Springer, 2011, pp. 752–763 (Cited on page 24).

[KTS10] Sergey Kosov, Thorsten Thormählen, and Hans-Peter Seidel. "Rapid Stereo-Vision Enhanced Face Recognition." In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. IEEE, Sept. 2010, pp. 2437–2440 (Cited on page 24).

[Kos+09] Sergey Kosov, Kristina Scherbaum, Kamil Faber, Thorsten Thormählen, and Hans-Peter Seidel. "Rapid Stereo-Vision Enhanced Face Detection." In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. IEEE, Nov. 2009, pp. 1221–1224 (Cited on pages 24, 37).

[KTS09] Sergey Kosov, Thorsten Thormählen, and Hans-Peter Seidel. "Accurate Real-Time Disparity Estimation with Variational Methods." In: *International Symposium on Visual Computing (ISVC)*. Vol. 5875. Lecture Notes in Computer Science. Springer, 2009, pp. 796–807 (Cited on pages 14, 24, 96, 145).

[Kos08] Sergey Kosov. "3D Map Reconstruction With Variational Methods." Master Thesis. Saarland University, Jan. 2008 (Cited on pages 14, 24, 96).

Index