Variational Methods for Human Modeling

THÈSE N^{O} 8680 (2018)

PRÉSENTÉE LE 17 JUILLET 2018

À LA FACULTÉ INFORMATIQUE ET COMMUNICATIONS

LABORATOIRE DE VISION PAR ORDINATEUR

PROGRAMME DOCTORAL EN INFORMATIQUE ET COMMUNICATIONS

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Timur BAGAUTDINOV

acceptée sur proposition du jury:

Prof. A. Wegmann, président du jury
Prof. P. Fua, Dr F. Fleuret, directeurs de thèse
Prof. B. Leibe, rapporteur
Prof. Y. Sheikh, rapporteur
Prof. A. Alahi, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2018

Product of knowledge and optimism is a constant.
— Lev Landau

Acknowledgements

First of all, I would like to thank my professor and advisor Pascal Fua, for letting me spend these years in such a stimulating environment as CVLab, and for giving me so much support and freedom in my research.
I am also very grateful to my co-advisor, François Fleuret, I really learned a lot from him, and enjoyed so many fun discussions about simulation theory and the future of AI.
I thank all the members of my thesis committee: professors Bastian Leibe, Yaser Sheikh, Alexandre Alahi, and the thesis president professor Alain Wegmann. It is a great honour to be able to share the results of my scientific endeavours with you.
I also want to thank all the members of the CVLab. In particular, I want to thank my friend and colleague Bugra for all the brainstimulating conversations over uncountable *coffee* breaks. Thanks Pierre, a colleague that I enjoyed working with and learned a lot from. Thanks Artem, for being such a great officemate over all these years. And all the others, thanks for being such exceptional people to work and hang out with.
I would also like to thank my family, my parents Irina and Marat, as well as my dear brother Rustam, for always being there for me.
And, of course, I want to thank my dear friend and fiancée Joëlle, without whom my PhD adventures would not be nearly as bright.

Lausanne, April 2018 Timur Bagautdinov

Abstract

A large part of computer vision research is devoted to building models and algorithms aimed at understanding human appearance and behaviour from images and videos. Ultimately, we want to build automated systems that are *at least* as capable as people when it comes to interpreting humans. Most of the tasks that we want these systems to solve can be posed as a problem of inference in probabilistic models. Although probabilistic inference in general is a very hard problem of its own, there exists a very powerful class of inference algorithms, variational inference, which allows us to build efficient solutions for a wide range of problems.

In this thesis, we consider a variety of computer vision problems targeted at modeling human appearance and behaviour, including detection, activity recognition, semantic segmentation and facial geometry modeling. For each of those problems, we develop novel methods that use variational inference to improve the capabilities of the existing systems.

First, we introduce a novel method for detecting multiple potentially occluded people in depth images, which we call DPOM. Unlike many other approaches, our method does probabilistic reasoning *jointly*, and thus allows to propagate knowledge about one part of the image evidence to reason about the rest. This is particularly important in crowded scenes involving many people, since it helps to handle ambiguous situations resulting from severe occlusions. We demonstrate that our approach outperforms existing methods on multiple datasets.

Second, we develop a new algorithm for variational inference that works for a large class of probabilistic models, which includes, among others, DPOM and some of the state-of-the-art models for semantic segmentation. We provide a formal proof that our method converges, and demonstrate experimentally that it brings better performance than the state-of-the-art on several real-world tasks, which include semantic segmentation and people detection. Importantly, we show that parallel variational inference in discrete random fields can be seen as a special case of proximal gradient descent, which allows us to benefit from many of the advances in gradient-based optimization.

Third, we propose a unified framework for multi-human scene understanding which simultaneously solves three tasks: multi-person detection, individual action recognition and collective activity recognition. Within our framework, we introduce a novel multi-person detection scheme, which relies on variational inference and *jointly* refines detection hypotheses instead of relying on suboptimal post-processing. Ultimately, our model takes as an inputs a frame sequence and produces a comprehensive description of the scene. Finally, we experimentally demonstrate that our method brings better performance than the state-of-the-art.

Fourth, we propose a new approach for learning facial geometry with deep probabilistic models and variational methods. Our model is based on a variational autoencoder with multiple sets of hidden variables, which are capturing various levels of deformations, ranging from global to local, high-frequency ones. We experimentally demonstrate the power of the model on a variety of fitting tasks. Our model is completely data-driven and can be learned from a relatively small number of individuals.

Keywords: human modeling, variational inference, depth-based human detection, conditional random fields, activity recognition, facial modeling, deep probabilistic models.

Résumé

La majorité de la recherche en vision par ordinateur est dévouée à la construction de modèles et algorithmes ayant pour but de comprendre l'apparence et le comportement humain, sur la base d'images et de vidéos. In fine, l'objectif est de construire des systèmes automatisés au moins aussi performants que les humains. La plupart des tâches que ces systèmes doivent résoudre peuvent être formulées comme problème d'inférence dans le contexte de modèles probabilistes. Bien que l'inférence statistique soit un problème difficile, il existe une catégorie très puissante d'algorithmes, à savoir l'inférence variationnelle, qui permet de créer des solutions pratiques et efficaces pour une large variété de problèmes.

Dans cette thèse, nous considérons plusieurs problèmes en vision par ordinateur visant à modéliser l'apparence et le comportement humain, incluant la détection, la reconnaissance d'activité, la segmentation sémantique ainsi que la modélisation de la géométrie faciale. Pour chacun de ces problèmes, nous développons de nouveaux modèles et méthodes utilisant des approches variationnelles, afin d'améliorer les capacités de systèmes existants.

Dans un premier temps, nous introduisons une nouvelle méthode pour la détection de plusieurs personnes en utilisant des images de profondeur, DPOM. À la différence d'autres approches, notre méthode effectue un raisonnement probabiliste conjoint, en propageant la connaissance d'une part de l'image sur l'autre afin de guider le raisonnement à faire. Ceci est particulièrement important dans le contexte de scènes clauses comprenant un grand nombre d'individus, comme cette méthode aide à gérer les situations ambigües résultant d'occlusions importantes. Nous démontrons sur de multiples ensembles de données que notre approche surpasse les méthodes actuelles.

Dans un deuxième temps, nous développons un nouvel algorithme d'inférence variationnelle qui fonctionne pour une large classe de modèles probabilistes qui incluent, entre autres, DPOM et certains modèles contemporains de segmentation sémantique. Nous fournissons une preuve formelle de convergence, et démontrons expérimentalement que la performance de notre algorithme est supérieure à celle des méthodes contemporaines sur plusieurs tâches de vie réelle. In particulier, nous montrons que l'inférence variationnelle parallèle dans le contexte de champs aléatoires discrets peut être considérée comme un cas spécial

de descente du gradient proximal, ce qui nous permet de bénéficier d'un grand nombre de progrès réalisés en optimisation basée sur le gradient.

Dans un troisième temps, nous proposons un cadre unifié pour la compréhension de scènes comprenant plusieurs individus, qui résout trois tâches simultanément : la détection d'individus multiples, la reconnaissance d'actions individuelles, ainsi que la reconnaissance d'activités collectives. Nous introduisons un nouvel algorithme de détection d'individus multiples, qui repose sur l'inférence variationnelle et qui raffine les hypothèses de détection de manière conjointe, plutôt que de se baser sur un post-traitement sous-optimal. Notre modèle prend pour entrée une séquence d'images et produit une description complète de la scène. Finalement, nous montrons expérimentalement que notre méthode présente une meilleure performance que les approches contemporaines.

Dans un quatrième et dernier temps, nous proposons une nouvelle approche pour l'apprentissage de la géométrie faciale, utilisant des modèles probabilistes profonds ainsi que des méthodes variationnelles. Notre modèle est basé sur un auto-encodeur variationnel possédant plusieurs ensembles de variables cachées qui capturent différents niveaux de déformations, allant du global au local. Nous démontrons expérimentalement la puissance du modèle sur plusieurs tâches d'inférence. Notre modèle est entièrement fondé sur les données et peut être appris à partir d'un nombre d'individus relativement restreint.

Mots clés : modélisation humaine, inférence variationnelle, détection humaine basée sur la profondeur, champs aléatoires conditionnels, reconnaissance d'activité, modélisation faciale, modèles probabilistes profonds.

Contents

List of Figures

List of Tables

1 Introduction

Understanding scenes that involve human beings is a long-standing problem in computer vision. This should not come as a surprise, because the amount of applications of automated systems which can interpret scenes and make predictions based on the images of people is countless: ranging from automotive industry to social science.

When one wants to create an automated system that is capable of interpreting and making predictions about certain phenomena, such as human appearance or behaviour, it seems natural to require this system to have an internal abstract representation of the phenomena, or, in other words, to have a *model*. In our own subjective human experiences, we sometimes experience the presence of such models, e.g. when spotting a familiar figure when staring at the clouds in the sky or looking at kids drawings of human bodies. In fact, evidence from neuroscience [32, 118, 131] suggests that a primate brain, a system that has inspired a significant amount of computer vision research, possibly contains structured models of faces and body parts.

For the purpose of this thesis, we will think of a model as a formal representation of *dependencies* between *variables*. Variables can represent some quantities of interest that we either observe, e.g. pixels of images of people, or are ultimately interested in estimating, such as the locations of humans in the images or the geometrical structure of their bodies. More formally, these dependencies between variables can be represented by a multivariate function, which assigns a single scalar value to any combination of the variables. Intuitively, the values of such a function indicate how *compatible* are its inputs: for example, how *likely* it is that certain pixels of the image contain a human.

A natural choice for such a function is a probability density function (probability mass function if the variables are discrete). Using probabilistic approach bears a lot of benefits, ultimately making models much more interpretable and comparable between each other, in particular by providing built-in mechanisms for measuring *uncertainty*. This principled

way of designing models is called *probabilistic modeling*.

In this context, a model is represented as a probability distribution over two categories of variables: observed, which often correspond to the inputs of the system, and hidden (latent) variables, which are usually the unknowns. Ultimately we want to use these models to answer some specific questions about the phenomena of interest, in other words, to do *inference*. For example, given a model that represents interdependencies between the *image* of a human face and its *geometry*, we might want to infer the latter from the former. More generally, the goal of inference is to estimate the states of the hidden variables, given the observations, either by finding a specific assignment of those variables, or by estimating distributions over those variables or their subsets. As it turns out, for a lot of non-trivial models the *exact* inference is computationally intractable, and thus various approximations have been developed. Among those approximate methods one of the most prominent ones is *variational inference*.

The core idea behind variational inference is to convert the problem of inference into the problem of *optimization*. Namely, instead of looking for the true distribution of interest, we introduce a parametric approximation, and then minimize the discrepancy between the two. Interestingly, it can be shown that this minimization is equivalent to minimizing the quantity known as (variational) free energy, a concept that is extensively used in many fields outside of computer vision, including statistical physics [163], biology and neuroscience [53, 140].

In this work we demonstrate that a lot of core machine perception problems, in particular those focused on understanding humans, can be successfully approached by designing probabilistic models and developing corresponding variational inference techniques. An important message of this work is that these methods not only provide a solid theoretical framework, but are also of great practical importance for vision-based human modeling.

1.1 Applications

The ultimate goal of probabilistic modeling and inference is to solve real-world problems. In what follows, we informally introduce the main application areas and their core challenges, which we tackle with the methods developed in this thesis.

Human Detection

Localizing objects in images is a well-studied problem in computer vision. Traditionally, the goal of object detection is to produce a bounding box for each object of interest in a given image (Figure 1.1). Detecting *humans* is a well-established field on its own, both

because of the impact it has in various industries, and because the problem is particularly challenging, due to high variations in appearance and frequent occlusions.

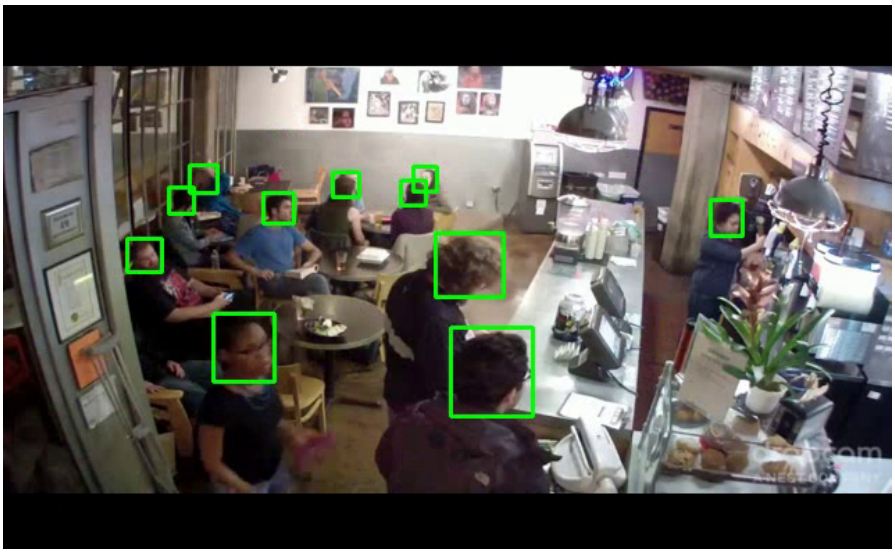


Figure 1.1: Human Detection Example. The goal of a human detection algorithm is to produce a bounding box (green squares) for each person (or person's head in this case) present in the given image.

This problem is particularly interesting from the perspective of the kind of methods discussed in this thesis, because it is naturally suitable to probabilistic models and variational inference. One of the reasons for this is the fact that the problem is inherently multi-modal, that is, there can be multiple bounding boxes which are all valid answers to the question of where an individual is. Moreover, in many realistic scenarios the scenes are very crowded, and there are a lot of occlusions, which makes the problem significantly harder and more ambiguous.

However, the vast majority of existing techniques for human detection (with few exceptions [9, 52]) ignore the aforementioned issues, and rely on sub-optimal post-processing. To help filling this gap, we introduce two alternative approaches that rely on variational inference, both in the context of depth-based human detection in Chapter 3, and in the context of RGB-based human detection and action recognition in Chapter 5.

Activity Recognition

Human detection is an important vision problem, yet it is only the first step to understanding *what* is actually happening in the scene, a problem usually referred to as *activity* or *action recognition*. Typically, the task of activity recognition is formulated as a classification problem, that is, we want to assign labels to the image or its parts,

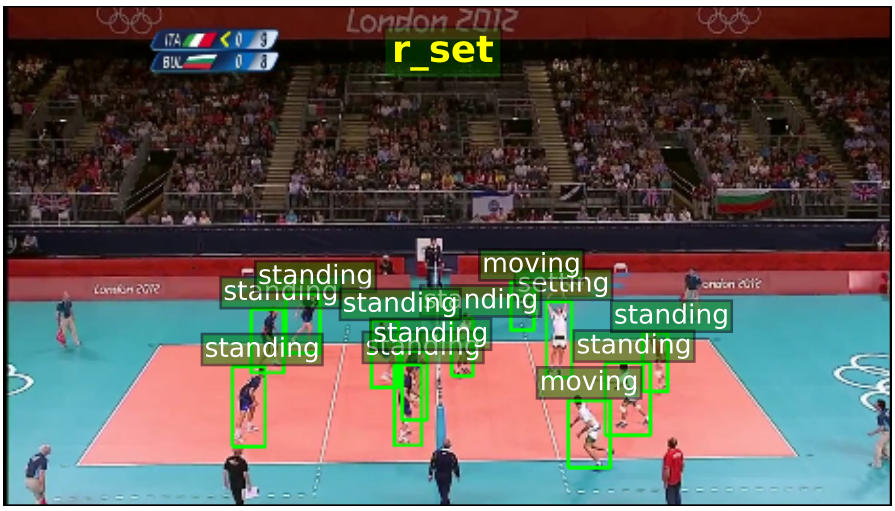


Figure 1.2: Activity Recognition Example. In this example, our goal is to produce a label for each volleyball player in the scene (e.g. "moving" or "standing"), as well as a label describing the overall activity that is happening on the court ("right set" or "left set" depending on which team is attacking).

in order to describe what is happening in the entire image or regions corresponding to individual humans (Figure 1.2). Most of the existing approaches for multi-person scene understanding use fragmented pipelines [31, 69, 123], with completely independent detection, tracking and activity recognition steps. This ultimately leads to sub-optimal decisions at each of those stages.

In Chapter 5, we propose a method that aims at solving those issues. Namely, we build on top of a variational human detection algorithm and develop an end-to-end system for multi-person scene understanding, which takes as input a sequence of images and produces a comprehensive interpretation of the scene: locations of the individuals, their actions and the group activity that they are performing.

Semantic Segmentation

Semantic image segmentation is a task of per-pixel image labelling, that is, we want to assign each pixel in the image to a specific entity class (Figure 1.3). This problem is quite challenging, one of the reasons being that it ultimately requires joint reasoning on the local and global evidence.

Conditional random fields (CRFs), a specific class of probabilistic models, have been successfully applied to this problem in a multitude of contexts [94, 113, 168]. CRFs are very generic and offer a lot flexibility in how to model various dependencies between

Figure 1.3: Semantic Segmentation Example. In this example we want to assign a "person" label (indicated by pink color) for each pixel of the image containing a person, and "background" label (indicated by black color) for all the others.

variables, thus allowing to jointly take into account image evidence on multiple levels. Unfortunately, this flexibility comes at a price, and for many CRF-based models the traditional inference algorithms either do not work or are too inefficient. In Chapter 4 we introduce an efficient and principled algorithm to do inference in these models, and demonstrate that it can be successfully applied to the problem of semantic segmentation, among others.

Modeling Faces

Figure 1.4: Modeling Facial Geometry Example. The goal of a facial reconstruction algorithm is to estimate the facial geometry (picture on the right), typically represented a collection of polygons (meshes), from a given image (picture on the left).

The problem of human geometry reconstruction, and, in particular, facial geometry

reconstruction (Figure 1.4) has traditionally relied on parametric models [16, 33, 75]. The reason for this is that the space of rough head shape geometries is rather limited. However, it is still a huge challenge to build a model that is useful for *high-quality* reconstructions, because they need to capture not only the overall head shape, but various levels of detail, ultimately including small, high-frequency deformations. In Chapter 6 we introduce a method to model *multi-scale* facial geometry which is based on deep probabilistic models, variational autoencoders.

1.2 Contributions

In this section, we formally describe the main contributions of this thesis.

Variational Human Detection in Depth Images

We propose a novel approach to computing the probabilities of presence of multiple and potentially occluding humans in a scene from a single depth map. To this end, we use a generative approach that explicitly models the distribution of depth images that would be produced if the probabilities of presence (occupancy maps) were known. We then optimize these occupancy maps with a variational scheme such that the model explains observed evidence as closely as possible. This allows us to exploit very effectively the available evidence and outperform state-of-the-art methods without requiring large amounts of data, or without using the RGB signal that modern RGB-D sensors also provide.

Efficient Variational Inference

Mean-field variational inference is a specific instance of variational inference which makes certain factorization assumptions on the variational posterior distribution. It is one of the most popular approaches to inference in conditional random fields, which are frequently used for tasks related to human detection and semantic segmentation. Standard mean-field optimization is based on coordinate descent and in many situations can be impractical. Thus, in practice, various parallel techniques are used, which either rely on *ad hoc* smoothing with heuristically set parameters, or put strong constraints on the type of models.

In this thesis, we propose a novel proximal gradient-based approach to optimizing the variational objective. It is naturally parallelizable and easy to implement. We prove its convergence, and demonstrate that, in practice, it yields faster convergence and often finds better optima than more traditional mean-field optimization techniques. Moreover, we show experimentally that our method is less sensitive to the choice of hyper-parameters.

Multi-Human Scene Understanding

We present a unified framework for understanding human social behaviors in raw image sequences. Our model jointly detects multiple individuals, infers their social actions, and estimates their collective actions with a single feed-forward pass through a neural network. We propose a single architecture that does not rely on external detection algorithms but rather is trained end-to-end to generate dense proposal maps that are refined via a novel variational inference scheme. The temporal consistency is handled via a person-level matching Recurrent Neural Network. The complete model takes as input a sequence of frames and outputs detections along with the estimates of individual actions and collective activities. We demonstrate state-of-the-art performance of our algorithms on multiple benchmarks.

Variational Human Face Modeling

We propose a method for learning non-linear face geometry representations using deep generative models. Our model is a variational autoencoder with multiple levels of hidden variables where lower layers capture global geometry and higher ones encode more local deformations. Based on that, we propose a new parameterization of facial geometry that naturally decomposes the structure of the human face into a set of semantically meaningful levels of detail. This parameterization enables us to do model fitting while capturing varying level of detail under different types of geometrical constraints.

1.3 Outline

We start this thesis with a general introduction into probabilistic models and inference methods in Chapter 2, with the main focus on variational methods. In Chapter 3, we introduce DPOM, an approach for depth-based multi-human detection based on probabilistic generative models. In Chapter 4 we propose a principled and efficient way to do variational inference in a large family of discrete probabilistic models which include, among others, the aforementioned DPOM and CRFs. In Chapter 5 we introduce a generic framework for multi-human scene understanding which relies on variational inference for joint multi-human detection. In Chapter 6 we discuss Compositional Variational Autoencoders, a novel deep probabilistic model for learning facial geometry of humans. Finally, in Chapter 7 we conclude this work with a general discussion and propose possible future research directions.

2 Background

In this thesis we consider a range of computer vision problems focusing on capturing human appearance and behaviour. One of the primary tools that we use to tackle those problems are probabilistic models and algorithms for inference. This chapter thus is meant to give a formal introduction to modern probabilistic modeling with a focus on variational inference-based methods.

2.1 Probabilistic Modeling and Inference

Computer vision is largely about building models and algorithms for reasoning about what is happening in the real world based on the image evidence. Probabilistic modeling provides a unified framework for reasoning and learning under uncertainty. Thus ultimately, many computer vision problems, ranging from low-level perception to high-level scene understanding, can be formulated as a problem of inference in probabilistic models.

More formally, let $\mathbf{I}$ be the set of *observed* variables, and $\mathbf{X}$ be a set of *latent* variables. In the context of computer vision, $\mathbf{I}$ represents the image evidence, such as image pixels or features, and $\mathbf{X}$ represents the unknowns we want to infer: depending on the application, they could be the locations of the individuals or the image labelling. The relationship between the observations and latent variables is captured by the joint density:

$$p(\mathbf{X}, \mathbf{I}) = p(\mathbf{I}|\mathbf{X})p(\mathbf{X}) \ , \tag{2.1}$$

where $p(\mathbf{I}|\mathbf{X})$ is the *likelihood* that encodes how likely it is to observe $\mathbf{I}$ given the hidden state $\mathbf{X}$, and $p(\mathbf{X})$ encodes our *prior* beliefs about $\mathbf{X}$. Note that, Eq. 2.1 actually defines the *generative process*: we can generate hidden variables by sampling from the prior distribution, and then substitute them into the likelihood to obtain samples of the observations, thus Eq. 2.1 is often called a *generative model*. This notion is clearly quite

useful if we are interested in actually generating novel images, but it also demonstrated a lot of potential in applications related to 3D reconstruction [7] and image completion [116].

Inference under the model of Eq. 2.1 amounts to computing the posterior distribution:

$$p(\mathbf{X}|\mathbf{I}) = \frac{p(\mathbf{X}, \mathbf{I})}{\int p(\mathbf{X}, \mathbf{I}) d\mathbf{X}} \ . \tag{2.2}$$

For non-trivial models, computing $p(\mathbf{X}|\mathbf{I})$ exactly is often very expensive. The main reason for this is the integral in the denominator of Eq. 2.2: for many non-trivial models it is impossible to compute it in closed-form, and depending on the space of the hidden variables, directly marginalizing out $\mathbf{X}$ can be prohibitively expensive if we do not put any simplifying assumptions on $p(\mathbf{X}, \mathbf{I})$. For example, if we consider a task of binary segmentation for images of relatively modest size 32×32, the number of possible combinations of latent variables $\mathbf{X}$ would be 2^{1024}, which is much more than the current estimate of the number of atoms in the observable universe ($\approx 2^{265}$). Thus, in most of the realistic scenarios we have to resort to approximations. We discuss some of the most prominent approximate inference methods in Section 2.2.

Note that, the formulation of Eq. 2.1 does not put any specific constraints on the functional form of the distribution. In what follows, we discuss several important model families which are used in computer vision applications.

2.1.1 Markov Random Fields

An important class of models that is especially popular in computer vision is known as Markov Random Fields (MRF). MRFs are defined as a following Gibbs family of distributions:

$$p(\mathbf{X}, \mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\{-\sum_{c \subset \mathcal{C}} \phi_c(\mathbf{X}_c, \mathbf{I}_c)\} \ , \tag{2.3}$$

where ϕ_c are a so-called *potentials*, functions defined on subsets of variables indexed by $c \in \mathcal{C}$, and $Z(\mathbf{I})$ is the normalizer also known as *partition function*. Every potential ϕ_c defines how variables in a specific subset interact between each other. If we consider a task of image segmentation, a potential can describe our prior knowledge about the distribution of the labels, for example that nearby pixels are more probable to correspond to the same object class. *Markov* in the model name comes from the markovian property that all the variables in the model should satisfy: every variable $\mathbf{X}_i$ should be independent of all the other variables given all its neighbours (variables that appear in the same potentials with it).

2.1.2 Conditional Random Fields

A *conditional* version of MRFs are known as Conditional Random Fields (CRF). CRFs have ultimately the same factorized form as MRFs, but instead of modeling the joint distribution, directly model the posterior:

$$p(\mathbf{X}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\{\sum_c -\phi_c(\mathbf{X}_c|\mathbf{I})\} , \tag{2.4}$$

where every potential function is conditioned on the global evidence $\mathbf{I}$.

A specific version of this model, *pairwise* CRFs, have been widely used for semantic segmentation problems:

$$p(\mathbf{X}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\{-\sum_i \phi_i^u(X_i|\mathbf{I}) - \sum_{i<j} \phi_{ij}^p(X_i, X_j|\mathbf{I})\} , \tag{2.5}$$

where the output $\mathbf{X}$ represents the labelling of the image, with individual X_i representing the label of an individual pixel, and $\mathbf{I}$ is the image evidence. Functions ϕ_i^u and ϕ_{ij}^p are called respectively *unary* and *pairwise* potentials. Unary potentials can be obtained from a classifier which produces a distribution of the labels for individual pixels from the image evidence, and pairwise potentials encode interactions between those individual pixels. Pairwise CRFs have been particularly successful because there are very efficient inference algorithms designed for this class of models [88, 89].

Pairwise CRFs are useful when we want to model interactions between individual variables, but often it can be beneficial to model higher-order statistics, for example, label consistency in certain image regions or cooccurrence of object classes in entire images. In these cases, it is often useful to introduce *higher-order potentials*:

$$p(\mathbf{X}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\{-\sum_i \phi_i(X_i|\mathbf{I}) - \sum_{i<j} \phi_{ij}(X_i, X_j|\mathbf{I}) - \sum_{c\in\mathcal{C}} \phi_c(\mathbf{X}_c|\mathbf{I})\} , \tag{2.6}$$

where potentials $\phi_c(\mathbf{X}_c|\mathbf{I})$ are functions of multiple (more than two) hidden variables. In the case of semantic segmentation, people have explored using potentials defined on entire images [84, 85], superpixels [66, 67] and regions corresponding to potential object candidates [93].

In general, CRFs allow a lot of flexibility in the way we define the exact functional form of the potential functions. However, depending on the particular choice of the potentials, designing an efficient inference algorithm can be tricky. In Chapter 4 we discuss some of these issues for discrete CRFs, and propose an efficient algorithm based on proximal gradient descent, which works for a wide range of potential functions.

2.1.3 Deep Probabilistic Models

Deep Neural Networks, and in particular Convolutional Neural Networks are a very powerful tool that achieved astonishing results for a lot of computer vision tasks, such as image classification [65, 92], detection [101, 125], action recognition [5, 72] and semantic segmentation [22, 117, 168]. It is thus a natural question to ask if we can use the strengths of these methods to build more powerful probabilistic models.

In fact, it is relatively straightforward to construct a probabilistic model such that it is parameterized with a neural network. One example would be to construct a CRF in such a way so that the parameters of its potential functions are produced by a neural network, something that has been extensively done for semantic segmentation [168] and people detection [9].

More generally, given a prior distribution $p(\mathbf{X})$, we can take the likelihood distribution from a parametric family, and use a neural network to map the hidden variables $\mathbf{X}$ to the *parameters* of this distribution:

$$p(\mathbf{X}, \mathbf{I}) = p(\mathbf{I}|\mathbf{X}; \boldsymbol{\theta})p(\mathbf{X}) = p(\mathbf{I}; f(\mathbf{X}; \boldsymbol{\theta}))p(\mathbf{X}) \ , \tag{2.7}$$

where $f(\mathbf{X}; \boldsymbol{\theta})$ is an arbitrary function with its own set of parameters $\boldsymbol{\theta}$, which can be e.g. a deep neural network.

The main issue with this kind of models, when we ultimately do not make any simplifying assumptions on the shape of the likelihood, is that inference becomes very hard. That is, not only the exact inference is intractable, but also many of the existing methods for approximate inference, such as mean-field (Section 2.3) are also intractable. Nevertheless, some recent advances [83, 128] demonstrated that one can still do inference in these models by combining variational approximations with stochastic optimization. We will discuss those methods in more detail in Section 2.4.

2.2 Approximate Inference

Once the model have been specified, the goal of probabilistic inference is to compute the posterior density $p(\mathbf{X}|\mathbf{I})$ distribution of the latent variables $\mathbf{X}$ given the observed variables $\mathbf{I}$. Since directly computing this density is often not possible, we need to resort to approximations. In this section, we discuss two popular approaches that have been extensively used in the field: those based on Monte Carlo sampling, and those based on variational approximations.

2.2.1 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) is a class of methods which have been widely used for approximate inference. The general idea of MCMC is to approximate the density of interest, e.g. $p(\mathbf{X}|\mathbf{I})$, by a set of samples. Since directly sampling from the target distribution is often not feasible, MCMC algorithms use approximate samples from a *Markov chain*, constructed in such a way so that the stationary distribution of this chain is the actual target distribution, $p(\mathbf{X}|\mathbf{I})$.

One of the classical MCMC algorithms is Metropolis-Hastings [64, 104], which works under the assumption that we have access to the *unnormalized density* $\tilde{p}(\mathbf{X}|\mathbf{I})$, which is proportional to the true posterior $p(\mathbf{X}|\mathbf{I})$. Given $\tilde{p}$ and an additional *proposal distribution* q, the Markov chain is built by iteratively sampling candidates from the proposal distribution and stochastically accepting or rejecting those based on the acceptance ratio: a quantity which indicates how probable is the new sample candidate with respect to the previously accepted sample. This acceptance ratio is computed only using the *ratios* of probabilities, thus having access to unnormalized $\tilde{p}$ is sufficient. For a complete introduction into MCMC for probabilistic inference and machine learning see e.g. [4, 111].

The main benefit of MCMC-based algorithms is that *asymptotically* they converge to the true posterior. However, this comes at a price: the main drawback of MCMC is its efficiency, since obtaining correct samples requires the Markov chain to converge, which can be very slow, thus making MCMC impractical for many real-world applications.

2.2.2 Variational Inference

Variational inference is a very popular alternative strategy to MCMC for computing approximate densities. Generally, variational methods are much more efficient than sampling-based methods, making them more scalable and applicable to larger datasets, and thus in most cases these methods are more suitable for computer vision applications.

The main idea of variational inference is to pose the problem of density estimation as *optimization*. In order to do so, we introduce an auxiliary density, a *variational distribution* $q(\mathbf{X};\boldsymbol{\lambda})$, from a family of distributions parameterized by free parameters $\boldsymbol{\lambda}$. We then proceed by minimizing the discrepancy between this parametric density and the true density:

$$\hat{\boldsymbol{\lambda}} = \arg\min_{\boldsymbol{\lambda}} \mathrm{KL}(q(\mathbf{X};\boldsymbol{\lambda})||p(\mathbf{X}|\mathbf{I})) \ , \tag{2.8}$$

where KL is the Kullback-Leibler (KL) divergence, a classical way to measure similarity

between two distributions, defined as:

$$\mathrm{KL}\left[q(\mathbf{X})||p(\mathbf{X})\right] = \int q(\mathbf{X}) \log \frac{q(\mathbf{X})}{p(\mathbf{X})} = \mathbb{E}_{q(\mathbf{X})}\left[\log q(\mathbf{X}) - \log p(\mathbf{X})\right] \ . \tag{2.9}$$

If the family $q(\mathbf{X};\boldsymbol{\lambda})$ has enough flexibility, then minimizing the KL divergence should yield a distribution $q(\mathbf{X};\hat{\boldsymbol{\lambda}})$ which is a good approximation to the true posterior.

Note that, the optimization problem of Eq. 2.8 is not fully specified: for each problem we need to specify the true posterior and the variational family. Both of these choices are important when developing efficient inference algorithms: the more complex the model distribution and the approximating family are, the harder the optimization problem is. Some popular choices for the "true" model are discussed in Section 2.1, and a very popular choice for the variational family is presented in Section 2.3.

Assuming that we specified the model and the approximating family, if we examine the KL divergence:

$$\begin{aligned}\mathbb{E}_q\left[\log q(\mathbf{X};\boldsymbol{\lambda}) - \log p(\mathbf{X}|\mathbf{I})\right] &= \mathbb{E}_q\left[\log q(\mathbf{X};\boldsymbol{\lambda})\right] - \mathbb{E}_q\left[\log \frac{p(\mathbf{X},\mathbf{I})}{p(\mathbf{I})}\right] \\ &= \mathbb{E}_q\left[\log q(\mathbf{X};\boldsymbol{\lambda})\right] - \mathbb{E}_q\left[\log p(\mathbf{X},\mathbf{I})\right] + \log p(\mathbf{I}) \ ,\end{aligned} \tag{2.10}$$

we will see that there is a term $\log p(\mathbf{I})$, log-evidence, which is the main reason we resorted to approximations in the first place. However, since this term now does not depend on q, we can equivalently maximize the following quantity:

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathbb{E}_{q(\mathbf{X};\boldsymbol{\lambda})}\left[\log p(\mathbf{X},\mathbf{I})\right] - \mathbb{E}_{q(\mathbf{X};\boldsymbol{\lambda})}\left[q(\mathbf{X};\boldsymbol{\lambda})\right] \ , \tag{2.11}$$

which is often referred to as variational (evidence) lower bound, or (negative) variational free energy. During optimization, the first term in Eq. 2.11 puts more weight on such values of the parameters that explain the observations better, whereas the second term puts weight on configurations which have higher entropy, and effectively acts as a regularizer. It is not hard to show from Eq. 2.10-2.11 that $\mathcal{L}(\boldsymbol{\lambda})$ indeed bounds the log-evidence:

$$\log p(\mathbf{I}) \geq \log p(\mathbf{I}) - \mathrm{KL}\left[q(\mathbf{X};\boldsymbol{\lambda})||p(\mathbf{I}|\mathbf{X})\right] = \mathcal{L}(\boldsymbol{\lambda}) \ , \tag{2.12}$$

which holds for arbitrary choice of $q(\mathbf{X};\boldsymbol{\lambda})$, since KL-divergence by definition is non-negative.

To summarize, the goal of variational inference is to approximate the true posterior $p(\mathbf{X}|\mathbf{I})$ with a variational distribution $q(\mathbf{X};\boldsymbol{\lambda})$, by optimizing $\mathcal{L}(\boldsymbol{\lambda})$ from Eq. 2.11 to find an optimal set of parameters $\hat{\boldsymbol{\lambda}}$. Depending on the settings of the problem, different optimization algorithms can be used: the traditional approach which works for certain

models and variational families is based on coordinate ascent, as discussed in Section 2.3. In Chapter 4, we present an optimization algorithm developed for discrete CRFs which is based on proximal gradient descent. More general variational approaches that work for deep probabilistic models are discussed in Section 2.4.

2.2.3 Discussion

Although MCMC-based methods and variational inference are ultimately two different approaches to solve exactly the same problem, that is not to say that one of them is necessarily better than the other. In fact, there has been an extensive research effort to understand the statistical properties of both of these approaches, and potentially building hybrid approaches that combine the strengths of both [83, 128, 137]. In the remainder of this chapter, we are focusing primarily on the variational methods, as they are generally more suitable for large-scale computer vision problems.

2.3 Mean-Field Variational Inference

One of the simplest choices for the variational family is to use a fully-factorized approximation, which assumes that the latent variables are independent given variational parameters:

$$q(\mathbf{X}; \boldsymbol{\lambda}) = \prod_k q_i(X_i; \boldsymbol{\lambda}_i) \ , \tag{2.13}$$

where each variable X_i has a separate density $q_i(X_i; \boldsymbol{\lambda}_i)$ with its own parameters $\boldsymbol{\lambda}_i$. This approach is known as *mean-field* variational inference, and originates from similar methods in statistical physics. The specific functional form of q_i depends a lot on the application in hand and the general structure of the problem. For example, in Chapter 3 we are representing location occupancy maps with binary variables, and Bernoulli distribution is a natural choice, whereas in Chapter 5 we are modeling human locations as continuous vectors, and thus set q_i-s to be multivariate Gaussians.

Recall from Section 2.2.2, that we want to find a set of parameters $\hat{\boldsymbol{\lambda}}$ that are a maxima of the variational lower bound $\mathcal{L}(\boldsymbol{\lambda})$. Note that, generally $\mathcal{L}(\boldsymbol{\lambda})$ might be non-convex, thus we are looking for an algorithm to find a *local* optima. For the mean-field variational family, we can derive a *coordinate ascent* algorithm that proceeds by iteratively optimizing the parameters of each individual factor $q_i(X_i; \boldsymbol{\lambda}_i)$, while keeping all the others fixed.

Namely, if we substitute the factorized form of $q(\mathbf{X}; \boldsymbol{\lambda})$ from Eq. 2.13 into Eq. 2.11, we

will get the following expression for the lower bound $\mathcal{L}(\boldsymbol{\lambda})$:

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathbb{E}_q[\log p(\mathbf{X}, \mathbf{I})] - \sum_i \mathbb{E}_{q_i}[\log q_i] \ , \tag{2.14}$$

where we omit the parameters of q-s for clarity. Further, we can write the lower bound $\mathcal{L}$ function of a single factor q_j, by separating the corresponding terms and collapsing the remaining terms that do not depend on X_j into the constant:

$$\mathcal{L}(q_j) = \mathbb{E}_{q_j}[\mathbb{E}_{\neg q_j}[\log p(\mathbf{X}, \mathbf{I})]] - \mathbb{E}_{q_j}[\log q_j] + \text{const} \tag{2.15}$$

where $\mathbb{E}_{\neg q_j}$ is the expectation with respect to all the factors *except* $q_j(X_j; \boldsymbol{\lambda}_j)$, i.e. $\prod_{i \neq j} q(X_i; \boldsymbol{\lambda}_i)$. If we now introduce the following auxiliary distribution:

$$q_j^*(X_j; \boldsymbol{\lambda}_j^*) \propto \exp\left\{\mathbb{E}_{\neg q_j}[\log p(\mathbf{X}, \mathbf{I})]\right\} \ , \tag{2.16}$$

we can rewrite the lower bound $\mathcal{L}(\boldsymbol{\lambda})$ as follows:

$$\mathcal{L}(q_j) = -\text{KL}[q_j || q_j^*] + \text{const} \tag{2.17}$$

Only the first term in Eq. 2.17 depends on q_j, thus *maximization* of $\mathcal{L}$ with respect to a single factor q_j is equivalent to the *minimization* of $\text{KL}[q_j || q_j^*]$. KL-divergence is minimized when its two argument distributions are identical, hence in order to maximize $\mathcal{L}$ with respect to q_j, we should set $q_j = q_j^*$. Consequently, by sequentially updating each factor with the update of Eq. 2.16, we gradually maximize the lower bound $\mathcal{L}(\boldsymbol{\lambda})$, eventually converging to a local maxima.

Importantly, convergence of the coordinate ascent algorithm is guaranteed only if we update each factor of the approximate posterior *sequentially*. In practice, as the number of factors becomes larger, the algorithm quickly becomes highly inefficient. In some specific cases, it can be shown [89] that updating all the factors *in parallel* does not break the convergence guarantees. However, this result does not hold for a lot of useful models [6, 52]. We address this issue in a more general setting in Chapter 4, and propose an efficient and provably convergent algorithm that works for a large class of discrete probabilistic models.

2.4 Stochastic Optimization for Variational Inference

The closed-form coordinate ascent algorithms exist only for specific models and variational families [59]. If we do not want to put any restrictions on the kind models that we use, and in particular parameterize our models with deep neural networks, those algorithms are no longer applicable.

An alternative class of algorithms that have been developed to work in a more general context is largely relying on stochastic optimization [83, 124]. Recall that our goal is to optimize the lower bound $\mathcal{L}(\boldsymbol{\lambda})$:

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathbb{E}_{q(\mathbf{X};\boldsymbol{\lambda})} \left[\log p(\mathbf{X}, \mathbf{I}) - \log q(\mathbf{X}; \boldsymbol{\lambda})\right] \ , \tag{2.18}$$

which can be intractable for the generic shape of $p(\mathbf{X}, \mathbf{I})$. One way to overcome this issue is to use Monte Carlo methods, and approximate the quantity with the *samples* from the approximation $q(\mathbf{X}; \boldsymbol{\lambda})$:

$$\mathcal{L}(\boldsymbol{\lambda}) \approx \sum_{s=1}^{S} \log(\mathbf{X}^{(s)}, \mathbf{I}) - \log q(\mathbf{X}^{(s)}; \boldsymbol{\lambda}) \ , \tag{2.19}$$

where S is the number of samples, and $\mathbf{X}^{(s)} \sim q(\mathbf{X}; \boldsymbol{\lambda})$. For optimization purposes, we are actually interested in the *gradient* of the variational objective $\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda})$:

$$\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda}) = \mathbb{E}_q \left[\nabla_{\boldsymbol{\lambda}} \log q(\mathbf{X}; \boldsymbol{\lambda})(\log p(\mathbf{X}, \mathbf{I}) - \log q(\mathbf{X}; \boldsymbol{\lambda}))\right] \ , \tag{2.20}$$

which, similarly to $\mathcal{L}$, is also intractable in the general case. However, as was demonstrated in a classical work of [132], as far as we can compute unbiased estimates of the gradient, we can still successfully use gradient-based optimization, subject to certain conditions on the step size.

In fact, there are multiple ways to compute an unbiased estimate of the gradient $\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda})$. For example, one could use the standard Monte Carlo estimate, sometimes called *score function* estimator [55]:

$$\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda}) \approx \sum_{s=1}^{S} \nabla_{\boldsymbol{\lambda}} \log q(\mathbf{X}^{(s)}; \boldsymbol{\lambda}) \left(\log p(\mathbf{X}, \mathbf{I}) - \log q(\mathbf{X}; \boldsymbol{\lambda})\right) \ , \tag{2.21}$$

where $\mathbf{X}^{(s)} \sim q(\mathbf{X}; \boldsymbol{\lambda})$. Unfortunately, although this estimate is relatively cheap to compute, it is known to suffer from large variance [83], and typically requires carefully designed variance reduction techniques [107, 124].

An alternative gradient estimate, known as *reparameterization estimate*, works in the case of continuous latent variables, and usually has significantly lower variance [83, 105]. The main idea, the *reparameterization trick*, is relatively straightforward: if $\mathbf{X}$ is a continuous variable, we can represent it as a deterministic differentiable function of the variational parameters $\boldsymbol{\lambda}$ and an auxiliary random variable $\boldsymbol{\epsilon}$ sampled from a known distribution $p(\boldsymbol{\epsilon})$:

$$\mathbf{X} = f(\boldsymbol{\epsilon}; \boldsymbol{\lambda}), \text{ where } \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}) \ , \tag{2.22}$$

where $p(\boldsymbol{\epsilon})$ and $f(\boldsymbol{\epsilon}, \boldsymbol{\lambda})$ are chosen such that $\mathbf{X} \sim q(\mathbf{X}; \boldsymbol{\lambda})$. For example, if we only have a single scalar hidden variable, X, and we choose the variational distribution q to be a univariate Gaussian, such that $X \sim \mathcal{N}(\mu, \sigma^2)$, it is not hard to see that we can express the sampling procedure as $X = \mu + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(\epsilon; 0, 1)$.

Now, we can use the parameterization of Eq. 2.22 to get the stochastic estimator of the variational lower bound of Eq. 2.19:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\lambda}) \approx \frac{1}{S} \sum_{s=1}^{S} \Big(\log p(\mathbf{X}^{(s)}, \mathbf{I}) - \log q(\mathbf{X}^{(s)}; \boldsymbol{\lambda}) \Big) \ , \\ \text{where } \mathbf{X}^{(s)} = f(\boldsymbol{\epsilon}, \boldsymbol{\lambda}) \ , \ \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}) \ . \end{aligned} \tag{2.23}$$

If $p(\mathbf{X}, \mathbf{I})$ and $q(\mathbf{X}; \boldsymbol{\lambda})$ are both differentiable functions of $\mathbf{X}$, we can directly take the gradients of the estimate in Eq. 2.23, and use those with one of the standard stochastic gradient descent updates. Ultimately, these techniques allow us to use complicated likelihood models, and have led to a lot of progress in creating powerful generative models in various domains [7, 83, 120, 127].

2.5 Related Paradigms

Probabilistic approach is not the only paradigm for building useful models for complex data. In this section, we discuss related modeling paradigms that received a lot of attention, especially in computer vision.

2.5.1 Structured Prediction

Structured prediction [114] is a generic term for prediction models on data where outputs are not limited to *single* discrete or scalar values, but rather consist of multiple, possibly high-dimensional and highly inter-dependent output variables. The key characteristic of structured models is that they take into account not only how the outputs depend on the inputs, but also model the dependencies *between the output variables* themselves. In many cases, these dependencies are represented explicitly, through a specific loss or density function. Ultimately, most of the probabilistic models that are discussed in this thesis, and in particular CRFs, can be seen as specific instances of structured models.

2.5.2 Energy-Based Models

Energy-Based models, EBMs [97], is a large class of models which can be seen as a generalization of probabilistic models. The key difference is that EBMs lift the requirement of the model density to be a valid probability distribution, thus potentially avoiding

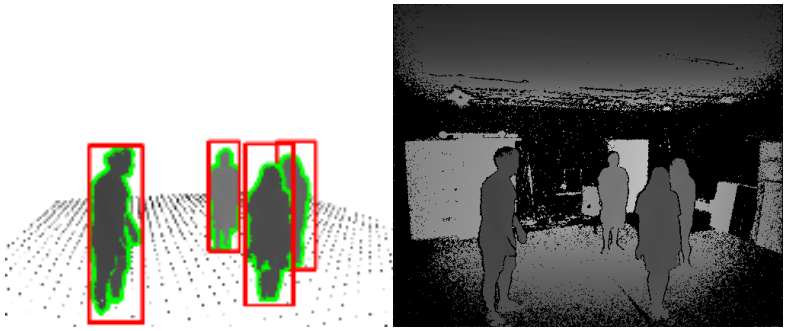
problems arising from our inability to efficiently compute partition function. However, when the application requires us to estimate uncertainty by producing a density model, EBMs still have to resort to the probabilistic interpretation. Nevertheless, in certain scenarios we might be interested in obtaining a single correct answer, and in this case it seems reasonable to trade-in probabilistic interpretation and uncertainty estimates for efficiency.

2.6 Discussion

In this section we formally introduced the general framework of probabilistic modeling and variational inference. We gave several examples of probabilistic models, mainly those which are widely used in vision applications. We introduced the standard factorized variational family, the *mean-field* approximation, and used it to describe the traditional coordinate ascent inference algorithm. We identified the main issues of that algorithm, some of which we address in Chapter 4. We also explored the use of stochastic optimization for variational inference in models with complicated likelihoods, which we use in Chapter 6 to develop a deep probabilistic model of facial geometries. In the remaining chapters, we demonstrate the utility of the models and inference techniques described above on a variety of human-centered computer vision problems.

3 Variational Human Detection in Depth Images

The advent of the original Kinect camera [80] and its sucessors has sparked a tremendous regain of interest for RGB-D imagers, which were formerly perceived as being either cumbersone or expensive. They have been used with great success for motion capture [141, 142] and are becoming increasingly popular for people detection in robotics applications [71, 103, 108, 147]. However, the former requires the algorithms to be trained on very large training databases, which may not always be easy to create, to achieve the desired level of performance while the latter usually do not make provisions for the fact that people may occlude each other. This results in failures such as those depicted by Figure 3.2.



(a) (b)

Figure 3.1: DPOM: generative model for depth maps. (a) Objects can be thought of as boxes, and images of objects are outlines within their rectangular projections. (b) Background is modeled explicitly: for each pixel there is a probability distribution, whose parameters are estimated from a set of background images.

In this chapter, we propose an approach that relies on a generative model to evaluate the probability of target objects being present in the scene while explicitly accounting for occlusions, which prevents such failures. It is inspired by an earlier approach to estimating these probabilities from background subtraction results from multiple cameras with

overlapping fields of view [52]. Here, we use instead a single depth-map and approximate probabilities of occupancy at separate locations by choosing these probabilities so that the lower bound on the evidence is maximized. In contrast to many other approaches, ours does statistical reasoning jointly, i.e. knowledge about one piece of image evidence helps us to reason about the rest. This allows in particular to properly infer the presence of a severely occluded target from the presence of a small fragment. The generative process is illustrated by Figure 3.1.

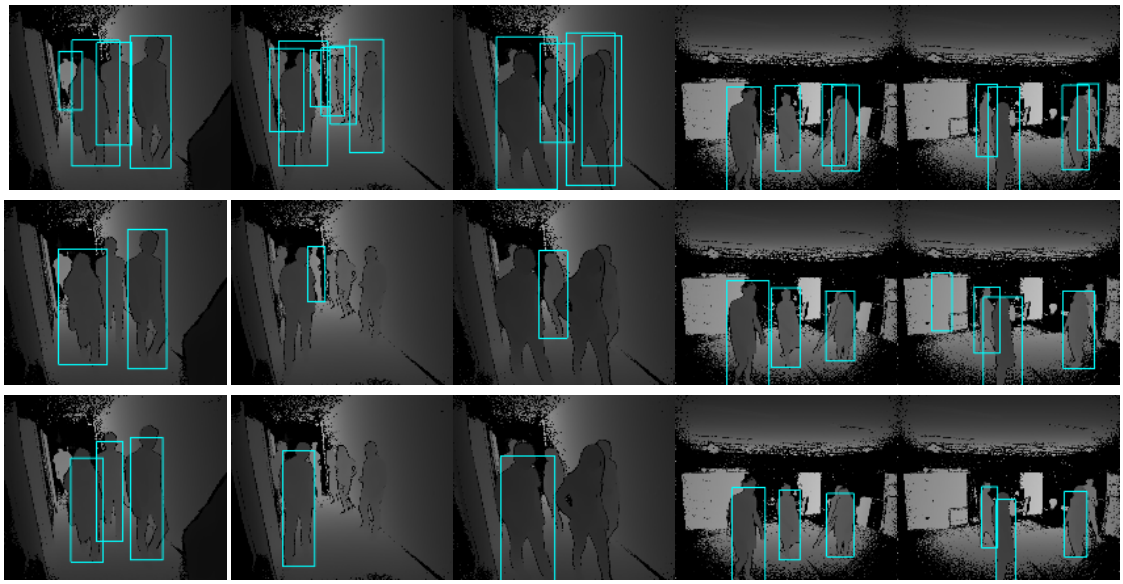


Figure 3.2: Situations in which we outperform state-of-the-art methods. Our approach (top) correctly detects most of the people including those that are severely occluded, whereas [108] (middle) and [80] (bottom) fail to do so.

We demonstrate that our approach outperforms [108,142,147] for people detection purposes while using only the depth image, whereas these other approaches also require either the use of the RGB image and an additional classifier or extensive training. Furthermore, because we do not require training, it took us very little additional effort to also detect a completely different kind of objects, that is, flying drones potentially occluding each other.

3.1 Related Work

There is an immense body of literature on people detection from regular images such as [29, 35, 51] to name but a few. However, most algorithms rely on features that are not necessarily present in depth images and are only rarely designed to be robust to occlusions.

As far as using depth images is concerned, an impressive success was the original Kinect algorithm [141, 142] that could not only detect people but also estimate their 3D poses. It has since been improved and is included in the latest Kinect for Windows SDK [80]. This constitutes an extremely strong baseline against which we will compare our algorithm to show that our approach to occlusion handling does boost performance when people hide each other.

One of the reasons why the Kinect algorithm [141, 142] works so well is that it relies on decision forests that have been trained on huge datasets of synthetically generated data, which would make it nontrivial to extend it to other categories of objects as we do for drones in this chapter.

Among the recent approaches that do not require such extensive training databases are those proposed in [71, 108, 147], which we briefly describe below and will also use as baselines in our result section.

In [147], the authors introduce a descriptor called the Histogram of Oriented Depths (HOD) that extends the HOG descriptor [29]. They train two separate SVMs, one for HOG features for RGB data and the other on HOD features for depth images, and combine their scores.

In [108], a complete framework for tracking people in RGB-D data is described. Detection comprises two steps: hierarchical-clustering of the depth maps and HOG-based RGB detection. The clustering step involves finding top-level clusters in a depth image and then applying heuristics to detect people's heads to produce more fine-grained sub-clusters. The RGB detector, which is based on an improved version of the HOG features [36] and trained on the INRIA Person dataset [29], is then applied to the corresponding parts of the RGB image. The code is available in the Point Cloud Library [134] and we used it in our experiments.

In [71], two detectors are also used, a depth-based one for people at close range and a color-based one for those further afield. The depth detector relies on template-matching followed by non-maxima suppression applied to regions of interests which are extracted using 3D point cloud clustering. Specifically, a 2D histogram is built by projecting all 3D points that could belong to objects on the ground plane and then finding clusters in that histogram. The RGB detector is a HOG-based detector with additional geometric constraints to reduce the search-space. This approach is very similar to that of [108], the main differences being the way RGB data is handled. Since this is not the main focus of our work and since the code that implements [108] runs on standard hardware, whereas that of [71] requires a modern GPU to use the complete RGB-D signal, we used the former as a representative of this class of techniques in our experimental evaluation.

To summarize: approaches discussed above typically do not perform occlusion reasoning explicitly, and mostly rely on heuristics when handling depth signals, which in many cases can provide reasonable results, but sometimes can lead to failures that are hard to interpret and predict, such as ones depicted in Figure 3.2.

There is also a number of approaches related to ours [49, 74] in that they also apply generative modeling and variational inference to vision problems. However, to the best of our knowledge, they focus on very different problems, such as learning medium-level representations of images [74] or learning natural scene categories [49], whereas our goal is to estimate location of multiple occluding objects in the environment.

3.2 Method

As discussed in the previous section, given a depth map of a scene featuring several people who are not occluding each other, state-of-the-art methods [80, 108, 147] do a good job of detecting them. However, these techniques do not perform the detection of all the targets jointly. Consequently, they can not re-assess properly the presence of a certain target, given evidence *and* the presence of occluding targets. More simply: a fragment of a target T in an empty room is a poor evidence of the presence of T. However, the presence of the same fragment when it is known that another target T' is present and hides the rest of T is a good evidence of the presence of T. Moreover, some of these methods [71, 108] rely on heuristics that sometimes result in failures even in simple cases. Figure 3.2 depicts both situations.

A similar problem arises when attempting to detect people on the basis of background subtraction results from multiple cameras with overlapping fields of view. It was addressed in [52] by using a generative model for background subtraction images. Namely, people were represented as boxes that project to rectangles in individual views. The algorithm was then estimating people locations on the discretized ground plane, such that the image synthesized according to the generative model matched the background subtraction results as well as possible in all the views. We will refer to this approach as POM. The strength of POM is that occlusions are naturally handled by the fact that rectangles corresponding to people further away from the camera are hidden by those corresponding to people that are closer.

Here, we also advocate the use of such a generative model to handle occlusions, but one designed to synthesize depth maps instead of binary images, as illustrated by Figure 3.1. We will refer to this approach as DPOM.

In our model, we consider a finite number of locations on the ground. An object of interest, located at one of these, is represented by a flat free-shape inside a rectangular bounding

box, as demonstrated in the Figure 3.1(a). In practice, with each location k we therefore associate two random variables. The first is a Boolean X_k that denotes the presence or absence of the object at location k. The second, a Boolean mask $\mathbf{M}_k$, represents the 2D contour of that object and is intended to improve the fit of the generative model to the data. We model the measured depths at each pixel in the image as conditionally independent given these variables, and distributed around the depth of the closest object, or according to the background distribution if no object is present.

Given this model, we estimate the $\mathbf{M}_k$ through a segmentation procedure, and turn the estimation of the probabilities of presence $P(X_k|\mathbf{D}, \mathbf{M})$ into a Bayesian inference problem as described formally in Section 3.3. Intuitively, what it allows us to do is predict the distribution of depth images that would be produced if the probabilities of presence were known and then to optimize them so that this distribution is centered around the observed one.

The introduction of the shape latent variables $\mathbf{M}_k$ leads to a better fit between the observed signal and the model, which is critical given that we exploit a single camera view. The standard POM algorithm achieves target localization through triangulation, using two or more cameras: even if the correct location of a target does not correspond to the best match in a individual view – in particular along the axis toward the camera – it is enforced through consistency in the other views which have non-parallel camera axis. In DPOM, since we use a single view signal, the accuracy along the axis of view is entirely due to the precision of the model.

3.3 Formulation

In this section, we formally describe our generative model, explain how we do inference on it, and then describe some implementation aspects.

3.3.1 Generative Model

We introduce first some notations, which are summarized with those of other sections in Table 3.1.

Let $\mathbf{D} \in \mathcal{D}^{|\mathcal{L}|}$ denote the depth map, with $\mathcal{L} = \{1, \ldots, N\}$ being the set of all pixels, and $\mathcal{D}$ being a set of all possible depth values. Let $d^\infty \in \mathcal{D}$ be a special value of depth encoding situation when no depth is observed (depicted in black in Figure 3.1(b)).

Let us assume we have discretized the ground plane into possible object locations $\mathcal{K} = \{1, \ldots, K\}$, as depicted in Figure 3.1(a). We introduce hidden binary variables $\mathbf{X} = \{X_k, \forall k \in \mathcal{K}\}$ with $X_k = 1$ if location k is occupied, 0 otherwise. Furthermore, for each

$\mathcal{K}$	set of all locations
$\mathcal{L}$	set of all pixels
$\mathcal{D}$	set of all possible depth values
X_k	binary occupancy variable for a location k
D_i	observed depth value variable at pixel i
d^∞	special value for when no depth is observed
M_{ki}	segmentation mask at pixel i for location k
$\mathcal{S}_k$	object silhouette for k-th location
$\lvert\mathcal{S}_k\rvert$	number of pixels in the k-th silhouette
$\mathbb{E}_p \cdot$	expectation w.r.t. a distribution p
$\sigma(x)$	sigmoid function $(1 + e^{-x})^{-1}$
ρ_k	approximate posterior $Q(X_k = 1)$
π^∞	probability of observing d^∞
π°	probability of observing an outlier
$\Delta_{l,i}$	$\log \theta_{l,i}(d)$, $l \in \mathcal{K} \cup \{\text{bg}\}$

Table 3.1: Notations used in this chapter.

location k, we have a corresponding crude rectangular representation of an object, which we call *silhouette* $\mathcal{S}_k \subset \mathcal{L}$. For each pixel of a silhouette we specify a corresponding depth distribution over $d \in \mathcal{D}$:

$$\theta_{ki}(d), \forall i \in \mathcal{S}_k \ , \tag{3.1}$$

where the specific shape and parameterization of the distribution θ_{ki} depends very much on the sensor used for depth acquisition. We will introduce a more detailed model in Section 3.3.3.

The silhouette specifies a very simplistic rectangular shape, whereas most of the objects have much more complex outlines. To encounter for that, we introduce *segmentation masks* $\mathbf{M}_k \subset \mathcal{L}, \forall k \in \mathcal{K}$. If $i \in \mathbf{M}_k$, it means that the pixel $i \in \mathcal{S}_k$ actually belongs to the object outline at k-th location (see Figure 3.1(a)).

Finally, some pixels belong to the background, rather than objects. In particular, when there are no objects in the scene, we observe only background. Thus, for each pixel of the depth map we have a corresponding background distribution over $d \in \mathcal{D}$:

$$\theta_{\text{bg},i}(d), \forall i \in \mathcal{L} \ . \tag{3.2}$$

Our ultimate goal is to estimate the posterior distribution $P(\mathbf{X}|\mathbf{D}, \mathbf{M})$ given the depth image $\mathbf{D}$ and segmentation masks $\mathbf{M}$. To do that, we introduce a generative model $P(\mathbf{D}|\mathbf{X}, \mathbf{M})$ and then apply Bayes' rule.

First, we assume that the prior occupancies $\mathbf{X}$ are independent from each other, *i.e.*:

$$P(\mathbf{X}) = \prod_{k \in \mathcal{K}} P(X_k) \ , \tag{3.3}$$

which intuitively means that objects occupy locations regardless of the presence of other objects.

Second, we assume that the observations for individual pixels $D_i, \forall i \in \mathcal{L}$ are *conditionally* independent, *i.e.* given $\mathbf{X}$ and $\mathbf{M}$.

We can now synthesize depth D_i for each pixel $i \in \mathcal{L}$ of the depth image. We select the model corresponding to a silhouette which is present ($X_k = 1$), contains the pixel ($i \in \mathcal{S}_k$), belongs to the object segmentation mask ($i \in \mathbf{M}_k$), and is the closest to the camera. It is, of course, also possible that we observe background at a specific pixel. This happens either if no silhouettes are present that has a model for this pixel ($i \notin \mathbf{M}_k, \forall k \in \mathcal{K}$), or if all of the silhouettes are further away from the camera (object is occluded by a part of the background). Note, that we assume that all the depth distributions $\theta_{l,i}(d), \forall l \in \mathcal{K} \cup \{\text{bg}\}$ are ordered w.r.t. the distance to the camera. In practice, we order them by their mean value $\mathbb{E}[\theta_{l,i}(d|d \neq d^{\infty})]$. More formally, $\forall i \in \mathcal{L}$:

$$l^* = \underset{l:\{X_l=1, i \in \mathbf{M}_l, l \in \mathcal{K}\} \cup \{\text{bg}\}}{\arg\min} \mathbb{E}[\theta_{l,i}(d|d \neq d^{\infty})], \tag{3.4}$$

$$D_i \sim \theta_{l^*,i}(d). \tag{3.5}$$

3.3.2 Inference

Even under the assumptions of our generative model, computing $P(\mathbf{X}|\mathbf{D}, \mathbf{M})$ directly is still computationally untractable, due to the dimensionality of $\mathbf{X}, \mathbf{M}$ and $\mathbf{D}$. To solve this, we first assume that $\mathbf{M}$ is given, by computing it as described in Section 3.3.4, and then derive a variational approximation for $P(\mathbf{X}|\mathbf{D}, \mathbf{M})$. Let us introduce the following mean-field variational posterior over hidden variables $\mathbf{X}$:

$$Q(\mathbf{X}) = \prod_{k \in \mathcal{K}} Q(X_k) \ , \tag{3.6}$$

where each $Q(X_k)$ is a Bernoulli distribution.

We then minimize the KL-divergence between $Q(\mathbf{X})$ and $P(\mathbf{X}|\mathbf{D}, \mathbf{M})$, which, as we saw in Chapter 2, is equivalent to getting an updated approximate posterior $Q^*(X_k)$ for each

X_k:

$$Q^*(X_k) \propto \exp\left(\mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D},\mathbf{M},\mathbf{X})]\right) \ , \tag{3.7}$$

where $\mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\cdot]$ denotes an expectation w.r.t. all the variables *except* for X_k.

Knowing that X_k is a Bernoulli variable, we can get the following update rule for $\rho_k = Q(X_k = 1)$:

$$\begin{aligned} \rho_k = \sigma(\quad & \mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D},\mathbf{X},\mathbf{M})|X_k = 1] - \\ & \mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D},\mathbf{X},\mathbf{M})|X_k = 0]) \ , \end{aligned} \tag{3.8}$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is a sigmoid function.

We want to substitute our generative model $P(\mathbf{D}|\mathbf{X},\mathbf{M})$ into Eq. 3.8. Let's first introduce some notations. Let $\Delta_{k,i} = \log \theta_{ki}(d)$, and $\Delta_{\mathrm{bg},i} = \log \theta_{\mathrm{bg},i}(d)$. Let us denote the prior of occupancy $\epsilon = P(X_k = 1)$, assuming it is identical $\forall k \in \mathcal{K}$. If we assume, without loss of generality, that silhouettes are sorted w.r.t. the distance to the sensor, then the probability of all silhouettes $\mathcal{S}_l : l < k$ being absent at a pixel $i \in \mathcal{L}$ will be:

$$\tau_{ki} = \prod_{l<k, i\in\mathbf{M}_l} (1 - \rho_l) \ , \tag{3.9}$$

which can be considered as a transparency at a certain pixel.

If we now substitute our model into Eq. 3.7, and evaluate expectation w.r.t. the current estimate of $Q(\mathbf{X})$ we will get the following update for $\rho_k, \forall k \in \mathcal{K}$:

$$\begin{aligned} &\rho_k = \sigma(\log \tfrac{\epsilon}{1-\epsilon} + \\ &\textstyle\sum_{i\in\mathbf{M}_k} \tau_{k,i}\Delta_{k,i} - \\ &\textstyle\sum_{i\in\mathbf{M}_k} \frac{1}{1-\rho_k}(\sum_{l>k, i\in\mathbf{M}_l} \tau_{l,i}\rho_l\Delta_{l,i} + \tau_{|\mathcal{K}|,i}\Delta_{\mathrm{bg},i})) \ . \end{aligned} \tag{3.10}$$

A complete derivation of Eq. 3.10 is provided in Appendix A.

3.3.3 Numerical Model

Up until now, we have not specified exactly our pixel depth distributions $\theta_{ki}(d)$ and $\theta_{\mathrm{bg},i}(d)$. The shape of those distributions can vary depending on the type of the sensor, but here we describe one that worked well for both versions of Kinect.

We assume that the distribution has two components: a Dirac and a robust Gaussian. The former is necessary, since in some cases sensor is incapable of producing a reasonable estimate of the depth, and reports a special value, d^∞. The robust Gaussian component

is simply a mixture of a Gaussian ($\mathcal{N}$) and a uniform distribution ($\mathcal{U}$), which takes care of possible outliers. Thus, each $\theta_{l,i}$ has four parameters: π^{∞}_{li}, a probability of observing d^{∞}, π°_{li}, a probability of observing an outlier, and μ_{li}, σ^2_{li}, which are the mean and the variance of the Gaussian component.

Finally, $\forall l \in \mathcal{K} \cup \{\text{bg}\}$:

$$\theta_{li}(d|d \neq d^{\infty}) = \pi^{\circ}_{li}\mathcal{U}(d) + (1 - \pi^{\circ}_{li})\mathcal{N}(d|\mu_{li}, \sigma^2_{li}) \ . \tag{3.11}$$

Particularly, the mean for object pixel distributions $\theta_{ki}(d), \forall k \in \mathcal{K}$ is a value one would observe if object was a flat surface facing the camera. The variance is fixed for an object type, e.g. for people we use a fixed value $\sigma^2_{ki} = 100, \forall k \in \mathcal{K}$. For background pixel distributions, we estimate all the parameters from a set of background frames.

3.3.4 Computing Segmentation Masks

As already mentioned, in theory we also could have obtained an approximation to posterior $P(\mathbf{M}|\mathbf{D})$, but it would be rather expensive computationally. Thus, given the observed depth map $\boldsymbol{d} \in \mathcal{D}^{|\mathcal{L}|}$, we apply the following simple procedure to obtain a point estimate for $\mathbf{M}_k, \forall k \in \mathcal{K}$:

$$\mathbf{M}_k = \{i : \pi^{\circ}\mathcal{U}(d_i) > (1 - \pi^{\circ})\mathcal{N}(d_i|\mu_{ki}, \sigma^2), d_i \neq d^{\infty}\} \ , \tag{3.12}$$

which in words means that we consider a pixel i to be a part of the object if depth value is observed and not considered an outlier under the model Eq. 3.11.

Note that, with the advent of fully-convolutional deep learning architectures [22, 102], the quality of semantic segmentation has dramatically improved. Thus ultimately, one can instead use a neural network trained on a large dataset of RGB images to estimate $\mathbf{M}$. This means, however, that the method would additionally require RGB signals as inputs, as well as the calibration between depth and RGB sensors.

3.3.5 Implementation Details

In reality, we have noticed that the update of Eq. 3.10 is not very robust. Namely, the predicted ρ_k are very peaky, and sometimes for a relatively small amount of evidence, the confidence of occupancy is very high. Since the depth maps are relatively noisy, it can lead to a large number of false positives. To avoid that, we use soft thresholding based

on the amount of pixel evidence:

$$\rho_k^* = \sigma\left(\alpha \frac{\sum_{i \in \mathbf{M}_k} \tau_{ki}}{|\mathcal{S}_k|} + \beta\right) \cdot \rho_k \ , \tag{3.13}$$

where, α and β are sigmoid parameters. They were set to disable those estimates that have very little evidence w.r.t. the size of our crude rectangular silhouette (we are using $\alpha = -100$, $\beta = 8$). In addition to that, we use a damping parameter to be able to make updates for each k in parallel. A more principled method based on gradient descent is proposed in Chapter 4.

3.4 Evaluation

In this section, we first report our people detection results and compare them against those of the baseline methods introduced in Section 3.1. We then show that our approach can be easily adapted to a very different detection problem, namely detecting flying drones that may occlude each other.

3.4.1 Datasets

There are many well-known and publicly available RGB datasets for testing pedestrian detection algorithms, such as those of [29, 37]. For RGB-D data, there are far fewer.

The Kinect Tracking Precision dataset (`KTP`) presented in [108] contains several sequences of at most 5 people walking in a small lab environment. They were recorded by a depth camera mounted on a robot platform and we use here the only one that was filmed while the camera was static. Authors provide ground truth locations of the individuals both on the image plane, and on the ground plane. Unfortunately, the quality of the ground truth for the ground plane is limited, due to the poor quality registration of the depth sensor location to the environment. In order to fix this, we made an effort and manually specified points corresponding to individuals on the depth maps, then projected them on the ground plane, and took an average to get a single point representing person location. This introduces a small bias as we only observe the outer surface of the person but any motion capture system would have similar issues.

In [103, 147], the authors report their results in a dataset containing about 4500 RGB-D images recorded in a university hall from a three statically mounted Kinects (`UNIHALL`). Unfortunately, there is no ground plane ground truth available, thus we only report results for image plane. To compare to their results, we follow evaluation procedure described in [147], that is, without penalizing approaches for not detecting occluded or hidden people. We also report our performance for the full dataset separately.

There are no publicly available datasets for multiple people tracking using the latest Kinect SDK [80] and we therefore created two of them ourselves. The first one (`EPFL-LAB`) contains around 1000 RGB-D frames with around 3000 annotated people instances. There are at most 4 people who are mostly facing the camera, presumably the scenario for which the Kinect software was fine-tuned. The second one (`EPFL-CORRIDOR`) was recorded in a more realistic environment, a corridor in a university building. It contains over 3000 frames with up to 8 individuals. Sample frames together with our detection results are shown in Figure 3.6.

The ground truth for the ground plane locations was obtained similarly to what has been described for `KTP` dataset, that is, for each person instance, we specified the points on the depth maps, projected them on the ground plane, and computed the average to get a location. In order to get individuals' bounding boxes in the image plane, for every target we compute the average of the projections of the marked pixels onto the image plane, and add a bounding box centered on it and sized according to their average depth.

Some approaches, including ours, require knowing both extrinsic and intrinsic camera parameters. The intrinsics were fixed for the specific Kinect we used. To compute the extrinsics, we manually specified the region of the depth map corresponding to the ground plane and then estimated the transformation from camera space to that plane.

3.4.2 Baselines

We use the following baselines for comparison purposes:

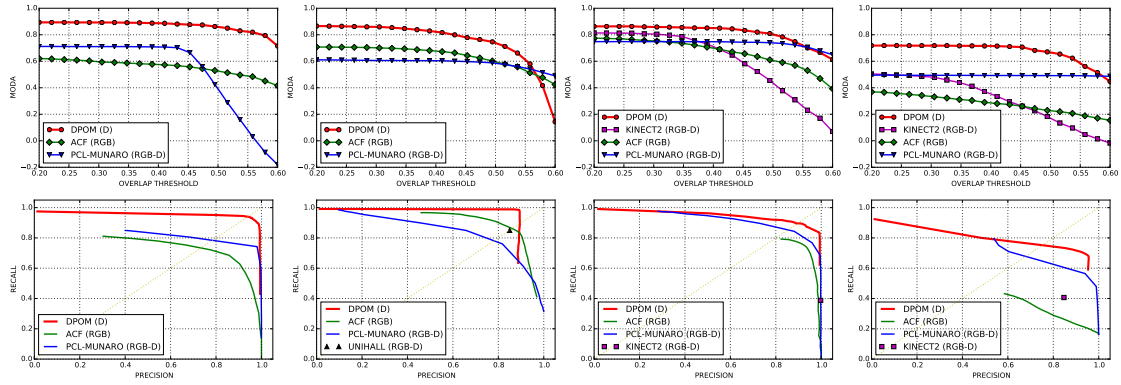
- `KINECT2` - the results obtained from the human pose estimation of the latest Kinect for Windows SDK [80]. It is not publicly known what specific algorithm is used. However in [141], the authors report that their algorithm is at the core of the human pose estimation for the older version of the Kinect software. For undisclosed reasons, the framework supports tracking up to 6 people, with the working depth range limited to 4.5 meters. To ensure fairness, we kept this restrictions in mind when using the `EPFL-LAB` and `EPFL-CORRIDOR` datasets. We do not penalize algorithms for not detecting more than 6 people or people who are further than 4.5 meters away.

- `UNIHALL` - RGB-D detector [147] based on HOG and HOD features. The code is not available and we therefore report only a single point on the precision-recall curve.

- `PCL-MUNARO` - RGB-D detector [108]. It uses modified HOG features [36] on regions extracted by depth segmentation. We used the implementation from the PCL library [134].

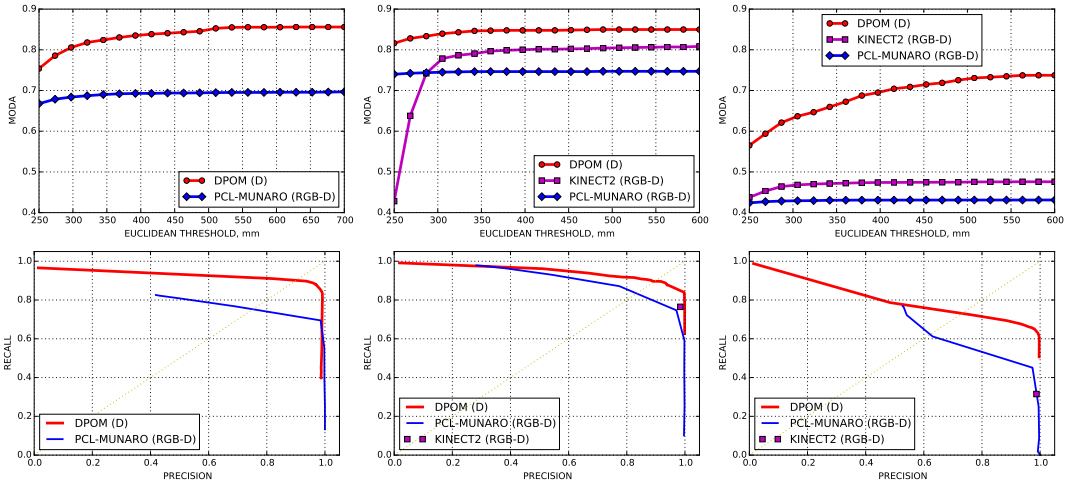
- `ACF` - RGB detector from [34], based on AdaBoost and aggregate channel features [35] to give a sense of what a state-of-the-art detector that does not use depth can do on these sequences.

3.4.3 Overall Performance



(a) `KTP` (b) `UNIHALL` (c) `EPFL-LAB` (d) `EPFL-CORRIDOR`

Figure 3.3: MODA (top) and precision-recall (bottom) for image plane ground truth. For each algorithm, the label indicates what type of information it uses: D - depth only, RGB - color only, RGB-D - both depth and color.



(a) `KTP` (b) `EPFL-LAB` (c) `EPFL-CORRIDOR`

Figure 3.4: MODA (top) and precision-recall (bottom) for ground plane ground truth. For each algorithm, the label indicates what type of information it uses: D - depth only, RGB - color only, RGB-D - both depth and color.

In Figure 3.3, we report overall performance comparisons on the four datasets introduced in Section 3.4.1. For each one, we report results in two different ways. We plot both

Multiple Objects Detection Accuracy (MODA) [13], as a function of bounding box overlap in the image plane, and also precision-recall curves. We made this choice to compare ourselves to authors who report image plane-only results. Here, precision-recall curves are shown for an overlap threshold of 0.4 as in [147]. The MODA curves are computed for fixed detector confidence threshold, the best-performing one for each approach, except in the case of `KINECT2` for which we have no way to set any threshold.

DPOM clearly outperforms all other approaches. This is true even though, as the `KINECT2`, we use *only* the depth information whereas the other algorithms also use the RGB information. Only for `UNIHALL` dataset, at overlap threshold above 0.55, does DPOM become slightly worse. This can be ascribed to the fact that we use a fixed-sized object model, and for this particular dataset and ground truth this size happens to be too small. This is not very crucial though, since at those values of overlap threshold, absolute performance of all the evaluated methods is rather low.

Note that `KINECT2` performs much worse on the `EPFL-CORRIDOR` sequence that in `EPFL-LAB` one. It is very hard to know why exactly, because the specific algorithm being used is a trade-secret. Our best guess is that in `EPFL-CORRIDOR`, the camera is slightly tilted and people do not appear to as being strictly vertical. If this interpretation were correct, it would illustrate the dangers of training an algorithm under specific assumptions that may prevent generalization. Another possible explanation is that some sequences start with people already present in the field of view, thus making it hard to use any kind of background subtraction. Whatever the case, the `EPFL-LAB` sequence presents neither of these difficulties and DPOM still performs better.

In Figure 3.4, we report MODA and precision-recall values computed in the ground plane instead of the image plane for the three methods for which it can be done. In ground-plane settings, we consider detection a match to the ground truth if it is within a certain Euclidean distance to it. The values of MODA are shown as a function of Euclidean thresholds, for a single best detection threshold for each algorithm. Precision-recall curves are plotted for a fixed Euclidean distance threshold of 500mm. The performance ordering stays essentially the same.

Recall that the evaluation procedure we have used so far is that of [147] in which not detecting occluded or hidden people is *not penalized*. To demonstrate that this choice does not have a major impact on our conclusions, we plot in Figure 3.5 equivalent precision-recall curves when they are penalized. As expected, the performance numbers are worse than those shown in Figure 3.3 for all methods. However, the ranking is preserved and the performance drop is smaller for DPOM. This highlights once more its ability to deal with occlusions.

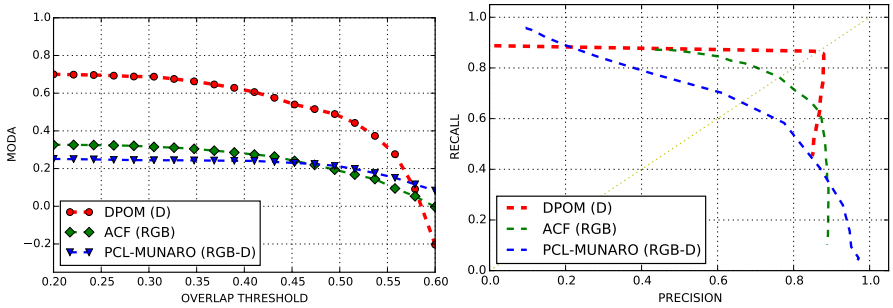


Figure 3.5: MODA (left) and precision-recall (right) for image plane ground truth for `UNIHALL` dataset. Approaches *were* penalized for not detecting occluded or hidden people.

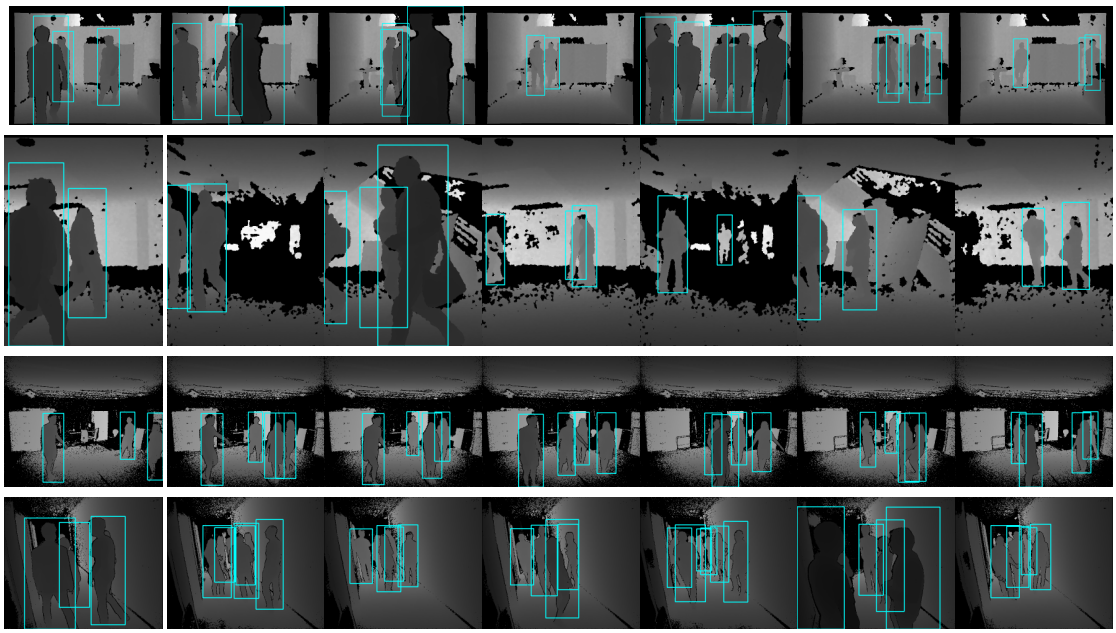


Figure 3.6: Sample detections for selected frames of our test datasets. From top to bottom, we show `KTP`, `UNIHALL`, `EPFL-LAB`, and `EPFL-CORRIDOR`.

3.4.4 Drones

To demonstrate the versatility of our approach, we have also applied it to a completely different type of objects, that is, drones. Note that, for people, we estimated their locations on the discretized ground plane. For drones, we instead use a discretized 3D space, and our algorithm thus estimates occupancy probabilities for each discrete 3D location in that space.

We filmed two drones flying in a room, sometimes occluding each other and sometimes being hidden by furniture. As in our people sequences, we obtained ground truth by

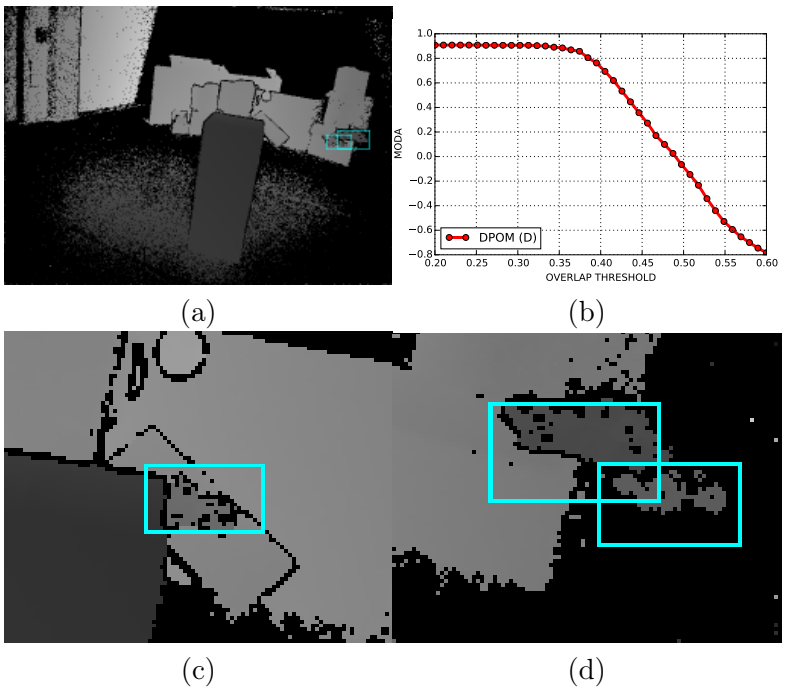


Figure 3.7: Detection results for drones. (a) Sample detections. (b) MODA score. (c) Detection with background occlusions. (d) Detection with occlusions.

manually specifying points on the drones, and then computing the bounding cube. To determine whether a detection is a match, we use overlap in bounding cubes.

Since there are no canonical baseline approaches, we only report our own MODA values in Figure 3.7. For overlap thresholds below 0.4, we obtain reasonable performance. For larger thresholds, the drop in performance is attributable to the fact that we discretize the 3D space, which means a relatively large localization error compared to the small size of the drones.

3.5 Discussion

We have introduced a probabilistic approach to estimating occupancy maps given depth images. We have shown that it outperforms state-of-the-art approaches both on publicly available datasets and our own challenging sequences. Moreover, the approach is generic enough to be easily adapted to a completely different object type, which we demonstrated by using it for detecting drones.

However, a weak point of our approach is speed: our current implementation is not real-time, and takes several seconds to process a single depth frame on a 2.3GHz Intel

CPU. This problem can be addressed using GPUs, since the bottleneck of our algorithm is iterating through the pixels. Another limitation, which is a consequence of using a rough generative model, is the lack of discriminative power. Our approach requires no training data but cannot distinguish between different object types as long as they fit our model well enough. Therefore, a possible future direction for extending this work would be to provide means for either combining our occupancy maps with the output of a discriminative classifier or making object models more sophisticated, possibly by learning them from the data. Furthermore, our approach relies on a static camera set-up, which limits the scope of potential applications. In practice, this issue can be solved by re-estimating the pose of the sensor with respect to the ground plane at every frame.

4 Efficient Variational Inference in Discrete Random Fields

Many Computer Vision problems, ranging from image segmentation to depth estimation from stereo, can be naturally formulated in terms of Conditional Random Fields (CRFs). Solving these problems then requires either estimating the most probable state of the CRF, or the marginal distributions over the unobserved variables. Since in general there can be many such variables, it is usually impossible to get an exact answer, and one must instead look for an approximation.

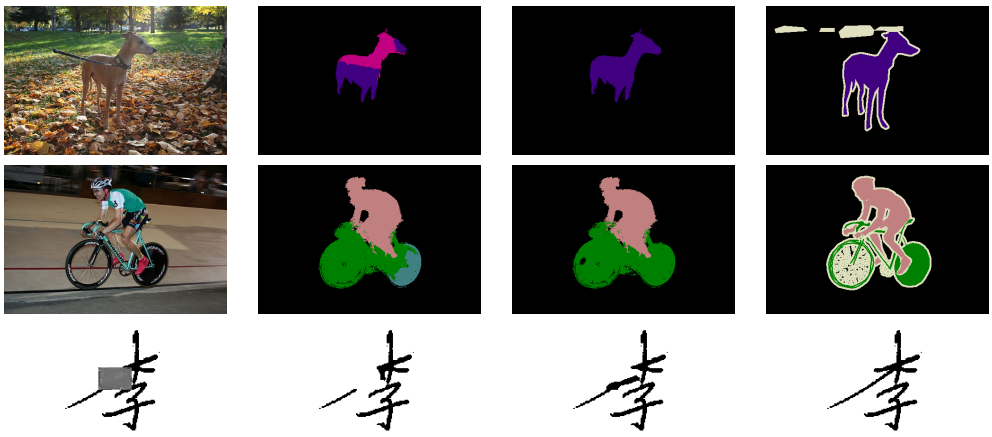


Figure 4.1: **First two rows:** VOC2012 images in which we outperform a baseline by adding simple co-ocurrence terms, which our optimization scheme, unlike earlier ones, can handle. **Bottom row:** Our scheme also allows us to improve upon a baseline for the purpose of recovering a character from its corrupted version.

Mean-field variational inference [158] is one of the most effective ways to do approximate inference and has become increasingly popular in our field [89, 136, 156]. It involves introducing a variational distribution that is a product of terms, typically one per hidden variable. These terms are then estimated by minimizing the Kullback-Leibler (KL)

divergence between the variational and the true posterior. The standard scheme is to iteratively update each factor of the distribution one-by-one. This is guaranteed to converge [14, 86], but is not very scalable, because all variables have to be updated sequentially. It becomes impractical for realistically-sized problems when there are substantial interactions between the variables. This can be remedied by replacing the sequential updates by parallel ones, often at the cost of failing to converge.

It has nonetheless recently been shown that parallel updates could be done in a provably convergent way for pairwise CRFs, provided that the potentials are concave [89]. When they are not, an *ad hoc* heuristic designed to achieve convergence, which essentially smooths steps by averaging between the next and current iterate, has been used over the years. This heuristic is mentioned explicitly in some works [20, 54, 149], or used implicitly in optimization schemes [6, 52, 156] by introducing an additional damping parameter.

However, a formal justification for such smoothing is never provided, which we do in this chapter. More specifically, we show that, by damping in the natural parameter space instead of the mean-parameter one, we can reformulate the optimization scheme as a specific form of proximal gradient descent. This yields a theoretically sound and practical way to chose the damping parameters, which guarantees convergence, no matter the shape of the potentials. When they are attractive, we show that our approach is equivalent to that of [89]. However, even when they are repulsive and can cause the earlier methods to oscillate without ever converging, our scheme still delivers convergence. For example, as shown in Figure 4.1, this allows us to add co-occurrence terms to the model used by a state-of-the-art semantic segmentation method [22] and improves its results. Furthermore, we retain the simplicity of the closed-form mean-field update rule, which is one of the key strengths of the mean-field approach.

In short, the contribution of this chapter is threefold:

- We introduce a principled, simple, and efficient approach to performing parallel inference in discrete random fields. We formally prove that it converges and demonstrate that it performs better than state-of-the-art inference methods on realistic Computer Vision tasks such as segmentation and people detection.
- We show that many of the earlier methods can be interpreted as variants of ours. However, we offer a principled way to set its metaparameters.
- We demonstrate how parallel mean-field inference in random fields relates to the gradient descent. This allows us to integrate advanced gradient descent techniques, such as momentum and ADAM [81], which makes mean-field inference even more powerful.

To validate our approach, we first evaluate its performance on a set of standardized

benchmarks, which include a range of inference problems and have recently been used to assess inference methods [54]. We then demonstrate that the performance improvements we observed carry over to three realistic computer vision problems, namely Characters Inpainting, People Detection and Semantic Segmentation. In each case, we show that modifying the optimization scheme while retaining the objective function of state-of-the-art models [22, 52, 115] yields improved performance and addresses the convergence issues that sometimes arise [156].

4.1 Related Work

In this section, we briefly revisit basic Conditional Random Field theory and the use of variational mean-field inference to solve the resulting optimization problems. We also give a short introduction into proximal gradient descent algorithms, on which our method is based. Note that, in this chapter we focus on models involving *discrete* random variables.

4.1.1 Conditional Random Fields

Let $\mathbf{X} = (X_1, \ldots, X_N)$ represent hidden variables and $\mathbf{I}$ represent observed variables. For example, for semantic segmentation, the X_is are taken to be variables representing semantic classes of N pixels, and $\mathbf{I}$ represents the observed image evidence.

Recall from Section 2.1.2, a Conditional Random Field (CRF) models the relationship between $\mathbf{X}$ and $\mathbf{I}$ in terms of the posterior distribution

$$P(\mathbf{X} \mid \mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp\left(\sum_{c \subset \{1,\ldots,N\}} \phi_c(\mathbf{X}_c \mid \mathbf{I})\right) , \tag{4.1}$$

where $\phi_c(.)$ are non-negative functions known as potentials and $Z(\mathbf{I})$ is the partition function. It is a constant that we will omit for simplicity since we are mostly concerned by estimating values of $\mathbf{X}$ that maximize $P(\mathbf{X} \mid \mathbf{I})$.

This model is often further simplified by only considering unary and pairwise terms:

$$P(\mathbf{X} \mid \mathbf{I}) \propto \exp\left(\sum_i \phi_i(X_i, I_i) + \sum_{(i,j)} \phi_{ij}(X_i, X_j)\right) . \tag{4.2}$$

4.1.2 Mean-Field Inference

Typically, one wants either to estimate the posterior $P(\mathbf{X}|\mathbf{I})$ or to find the vector $\hat{\mathbf{X}}$ that maximizes $P(\mathbf{X}|\mathbf{I})$, which is known as the MAP assignment. Unfortunately, even for the simplified formulation of Eq. 4.2, both are intractable for realistic sizes of $\mathbf{X}$. As a result, many approaches settle for approximate solutions. These include sampling methods, such as Gibbs sampling [58], and deterministic ones such as mean-field variational inference [164], belief propagation [87, 106, 109], and others [17, 62]. A comprehensive comparison of inference methods in discrete models is provided in [76].

Note that, mean-field methods have been shown to combine the advantages of good convergence guarantees [14], flexibility with respect to the potential functions that can be handled [136], and potential for parallelization [89]. As a result, they have become very popular in our field. Furthermore, they have recently been shown to yield state-of-the-art performance for several computer vision tasks [22, 136, 156, 168].

Mean-field involves introducing a distribution Q of the factorized form

$$Q(\mathbf{X} = (x_1, \ldots, x_N); \mathbf{q}) = \prod_{i=1}^{N} Q_i(X_i = x_i; \mathbf{q}_i) \ , \tag{4.3}$$

where $Q_i(\,.\,; \mathbf{q}_i)$ is a categorical distribution with mean parameters $\mathbf{q}_i$. That is,

$$\forall l,\ Q_i(X_i = l; \mathbf{q}_i) = q_{i,l}, \tag{4.4}$$

with $\mathbf{q}$ in the space $\mathcal{M}$ such that $\forall i \in \{1, \ldots, N\}, l \in \{1, \ldots, L\},\ 0 \leq q_{i,l} \leq 1$ and $\forall i, \sum_l q_{i,l} = 1$, where N is often the number of pixels, and L is the number of labels.

Q is then used to approximate $P(\mathbf{X} \mid \mathbf{I})$ by minimizing the KL-divergence:

$$\mathrm{KL}(Q||P) = \sum_{\boldsymbol{x}} Q(\mathbf{X} = \mathbf{x}; \mathbf{q}) \log \frac{Q(\mathbf{X} = \mathbf{x}; \mathbf{q})}{P(\mathbf{X} = \mathbf{x} \mid \mathbf{I})} \ . \tag{4.5}$$

In some cases, this approximation is the desired final result. In others, one seeks a MAP assignment. To this end, a standard method is to select the assignment that maximizes the *approximate* posterior $Q(\mathbf{X}; \mathbf{q})$, which is equivalent to rounding when the X_is are Bernoulli variables. An alternative approach is to draw samples from $Q(\mathbf{X}; \mathbf{q})$.

When minimizing the KL-divergence of Eq. 4.5, $Q(\mathbf{X}; \mathbf{q})$ can be reparameterized in terms of its *natural* parameters defined as follows. For each variable X_i and label l, we take the natural parameter $\theta_{i,l}$ to be such that

$$Q(X_i = l; \mathbf{q}_i) = q_{i,l} \propto \exp[-\theta_{i,l}]. \tag{4.6}$$

As we will see below, this parameterization often yields simpler notations and implementations.

Sweep Mean-Field Inference

Minimizing the expression of Eq. 4.5 is equivalent (see Section 2.2.2) to minimizing

$$\mathcal{F}(\mathbf{q}) = \underbrace{-\mathbb{E}_{Q(\mathbf{X};\mathbf{q})}[\log P(\mathbf{X} \mid \mathbf{I})]}_{\mathcal{E}(\mathbf{q})} + \underbrace{\mathbb{E}_{Q(\mathbf{X};\mathbf{q})}[\log Q(\mathbf{X};\mathbf{q})]}_{-\mathcal{H}(\mathbf{q})}, \tag{4.7}$$

with respect to $\mathbf{q} \in \mathcal{M}$. $\mathcal{F}(.)$ is sometimes called the variational free energy, or (negative) variational lower bound. Its first term is the expectation of the energy under $Q(\mathbf{X};\mathbf{q})$, and its second term is the negative entropy, which acts as a regularizer.

As discussed in Section 2.3, one can minimize $\mathcal{F}(\mathbf{q})$ by iteratively updating each $q_{i,l}$ in sequence while keeping the others fixed. Each update then involves setting $q_{i,l}$ to

$$q^{\star}_{i,l} \propto \exp\left[\mathbb{E}_{Q(\mathbf{X}/X_i;\mathbf{q})}\left[\log P(\mathbf{X} \mid \mathbf{I})\right]\right]. \tag{4.8}$$

In what follows, we refer to this coordinate descent procedure, which we will call `SWEEP`. As demonstrated in Section 2.3, it is guaranteed to converge to a local minimum of $\mathcal{F}$. However, it tends to be very slow for realistic image sizes and impractical for many computer vision problems [89, 156]. Namely, in the case of dense random fields, it involves re-computing a large number of expectations (one per factor adjacent to the variable) after each sequential update. Filter-based mean-field inference [88] attempts to reduce the complexity of these updates, but it effectively performs parallel updates, which we will describe below.

Parallel Mean-Field Inference

To obtain reasonable efficiency in practice, computer vision practitioners often perform the updates of Eq. 4.8 in parallel as opposed to sequentially. Not only does it avoid having to reevaluate a large number of factors after each update, it also allows the use of vectorized instructions and GPUs, both of which can have a dramatic impact on the computation speed.

Unfortunately, these parallel updates invalidate the convergence guarantees and in practice often lead to undesirable oscillations in the objective. Several approaches to remedying this problem have been proposed, which we review below.

Damping A natural way to improve convergence is to replace the updates of Eq. 4.8 by a damped version, expressed as

$$q_{i,l}^{t+1} = (1-\eta) \cdot q_{i,l}^{t} + \eta \cdot q_{i,l}^{\star} \ , \tag{4.9}$$

where t denotes the current iteration, $q_{i,l}^{\star}$ is the result of solving the optimization problem of Eq. 4.8, and η is a heuristically chosen damping parameter. This damping is explicitly mentioned in papers such as [20, 54, 149]. In [156], convergence issues are mentioned and a damping parameter is provided in the publicly available code. Similarly, multi-person detection methods such as POM [52] and DPOM, which we discussed in the previous chapter, rely on mean-field optimization with repulsive terms and use the same damped update of Eq. 4.9.

Damping delivers satisfactory results in many cases, but does not formally guarantee convergence. It may fail if the parameter η is not carefully chosen, and sometimes changed at different stages of the optimization. In all the approaches that we are aware of, this is done heuristically. We will refer to this type of methods as `ADHOC`.

Concave potentials A principled way to address the convergence issue for the pairwise random fields is offered in [89], and we refer to the corresponding algorithm as `FULL-PARALLEL`. However, authors restrict their potentials ϕ_{ij} of Eq. 4.2 to be concave, which in some cases is reasonable, but as we will show in Section 4.3, many Computer Vision models violate this requirement. By contrast, our approach is similarly principled but without additional constraints. In practice it works for higher-order, or, equivalently, non-pairwise potentials.

4.1.3 Proximal Gradient Descent

Let F be a generic objective function of the form $F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$, where g is a regularizer, and $\mathbf{x}_t$ is the value of the optimized variable at iteration t of a minimization procedure on a constraint set $\mathcal{X}$. Proximal gradient descent, also known as composite mirror-descent [43], is an iterative method that relies on the update rule

$$\mathbf{x}^{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \{ \langle \mathbf{x}, \nabla f(\mathbf{x}^t) \rangle + g(\mathbf{x}) + \lambda \Psi(\mathbf{x}, \mathbf{x}^t) \} \ , \tag{4.10}$$

where Ψ is a non-negative *proximal* function that satisfies $\Psi(\mathbf{x}, \mathbf{x}^t) = 0$ if and only if $\mathbf{x} = \mathbf{x}^t$, and $\lambda > 0$ is a scalar parameter. g contains the terms of the objective function that do not need to be approximated to the first order, while still allowing efficient computation of update of Eq. 4.10. Ψ can be understood as a distance function that accounts for the geometry of $\mathcal{X}$ [151] while also making it possible to compute the update of Eq. 4.10 efficiently. λ can then be thought of as the inverse of the step size. Intuitively,

solving the minimization problem of Eq. 4.10 amounts to minimizing the first-order approximation of $F(\mathbf{x})$, while taking into account the composite structure of the function and keeping the next estimate $\mathbf{x}^{t+1}$ close to $\mathbf{x}^t$.

As shown in Section 4.2.1, our algorithm is a version of proximal gradient descent in which Ψ is based on the KL-divergence and allows automated step-size adaptation as the optimization progresses. Recently, a variational approach that also relies on the KL-divergence as the proximal function has been proposed [79]. This work explores the connection between the KL-proximal method and the Stochastic Variational Inference [2, 68]. However, the method presented there is not directly applicable to discrete random fields, especially for the Vision problems we consider. Moreover, it does not allow for step size adaptation, which often yields better performance, as we demonstrate in our experiments.

4.2 Method

As discussed in the previous section, the goal of mean-field inference is to

$$\underset{\mathbf{q}\in\mathcal{M}}{\text{minimize}}\ \mathcal{F}(\mathbf{q}) \tag{4.11}$$

where $\mathcal{F}$ is the variational free energy of Eq. 4.7. Performing sequential updates of the $q_{i,l}$ is guaranteed to converge, but can be slow. Parallel updates are usually much faster, but the optimization procedure may fail to converge.

In this section, we introduce our approach to guaranteeing convergence whatever the shape of the pairwise potentials. To this end, we rely on proximal gradient descent as described in Section 4.2.1 and formulate the proximal function Ψ in terms of the KL-divergence. This is motivated by the fact that it is more adapted to measuring the distance between probability distributions than the usual L2 norm, while being independent of how the distribution is parameterized.

We will show that this both guarantees convergence and yields a principled way to obtain a closed form damped update equation equivalent to Eq. 4.9.

4.2.1 Proximal Gradient for Mean-Field Inference

In our approach to minimizing the variational free energy of Eq. 4.7, we treat $\mathcal{E}$ as the function f of Eq. 4.10 and the negative entropy $-\mathcal{H}$ as the regularizer g. This choice stems from the fact that $-\mathcal{H}$ is separable, and therefore, can be minimized in parallel in Eq. 4.10, without using a first order approximation. Also, $-\mathcal{H}$ being the regularizer g

means that we do not need to look at its derivatives with respect to the mean-parameters, which are not well behaved when they approach zero. We then define

$$\Psi^t(\mathbf{q}, \mathbf{q}^t) = \sum_i \sum_l d^t_{i,l} q_{i,l} \log \frac{q_{i,l}}{q^t_{i,l}} = \mathbf{D}^t \odot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t) \ , \tag{4.12}$$

where KL is the non-negative KL-divergence, which is a natural choice to measure discrepancy between distributions. $\mathbf{D}^t$ is a diagonal matrix with positive diagonal elements $d^t_{i,l}$s, which we introduce to allow for anisotropic scaling of the proximal KL-divergence term. As will be discussed below, different choices of the $d^t_{i,l}$s yield different variants of our algorithms. Note however that, Ψ^t is a valid proximal function.

The update of Eq. 4.10 then becomes

$$\mathbf{q}^{t+1} = \arg\min_{\mathbf{q}\in\mathcal{M}} \{\langle \mathbf{q}, \nabla\mathcal{E}(\mathbf{q}^t)\rangle - \mathcal{H}(\mathbf{q}) + \mathbf{D}^t \odot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t)\} \ . \tag{4.13}$$

This computation can be performed independently for each index $i \in \{1, \ldots, N\}$. Furthermore, we prove in the Appendix B that it can be done in closed form and can be written as

$$\begin{aligned} q^{t+1}_{i,l} \quad \propto \exp[\quad & \eta^t_{i,l} \cdot \mathbb{E}_{Q(\mathbf{X}/X_i=l;\mathbf{q})} \left[\log P(\mathbf{X}|\mathbf{I})\right] \\ & +(1-\eta^t_{i,l}) \cdot \log q^t_{i,l}] \ , \end{aligned} \tag{4.14}$$

where $\eta^t_{i,l} = \dfrac{1}{1+d^t_{i,l}}$. Eq 4.14 can be rewritten as

$$\theta^{t+1}_{i,l} = \eta^t_{i,l} \cdot \theta^\star_{i,l} + (1-\eta^t_{i,l}) \cdot \theta^t_{i,l} \ \ , \tag{4.15}$$

where $\theta^\star_{i,l} = -\,\mathbb{E}_{Q(\mathbf{X}/X_i;\mathbf{q})} \left[\log P(\mathbf{X}|\mathbf{I})\right]$ now is a natural parameter, like those of Eq. 4.6. In other words, we have replaced the heuristic update rule of Eq. 4.9 in the space of mean parameters by a principled one in the space of natural ones. As we will see, this yields performance and convergence improvements in most cases. As for the stopping criteria, one can define one based on the value of the objective, or, in practice, run inference for a fixed number of iterations.

4.2.2 Fixed Step Size

The simplest way to instantiate our algorithm is to fix all the $d_{i,l}^t$s of Eq. 4.12 to the same value d and to write

$$\forall t \ , \ \mathbf{D}^t = \mathbf{D} = d\mathbb{I} \ \Rightarrow \ \forall t, i, l, \eta_{i,l}^t = \frac{1}{1+d} \ , \tag{4.16}$$

where $\eta_{i,l}^t$ plays the same role as the damping factor of Eq. 4.9. We now show that this is guaranteed to converge when the proximal term is given enough weight.

In our mean-field settings, $\mathcal{E}(\mathbf{q})$ is a polynomial function of the mean-parameters vector $\mathbf{q}$. Therefore, one can always find some positive real number L such that the gradient of $\mathcal{E}$ is *L-Lipschitz continuous*.

In Appendix B, we prove that this property implies that our proximal gradient descent scheme is guaranteed to converge for any fixed matrix $D = d\mathbb{I}$ such that $d > L$.

Intuitively, when updating the value of $\mathbf{q}^t$ to $\mathbf{q}^{t+1}$, the magnitude of the gradient change controlled and thus the coordinate-wise optimum $\theta_{i,l}^\star = -\nabla\mathcal{E}(\mathbf{q}^t)_{i,l}$ will also be changing smoothly across iterations. As a result, L is the key value to understand oscillations. In practice, our goal is to find its smallest possible value to allow steps as large as possible while guaranteeing convergence.

In the pairwise case, the Hessian of the objective function is a constant matrix, which we call potential matrix. Therefore, the highest eigenvalue of the potential matrix is a valid Lipschitz constant and efficient methods allow to compute it for moderately sized problems.

In fact, the convergence result presented in [89] is strongly related to this. Namely, assuming that the potential matrix is negative semi-definite, is equivalent to assuming that $L < 0$ in our formulation. This directly corresponds to the concavity assumptions on the potentials in [89]. Therefore, under the assumptions of [89], our algorithm leads to $\eta = 1$, corresponding to the fully-parallel update procedure. In that sense, our procedure is a generalization of the one proposed by [89].

In the non-pairwise case, the Hessian is not constant, and the calculation of the Lipschitz constant is not trivial. For each specific problem, bounds should be derived using the particular shape of the CRF at hand.

4.2.3 Adaptive Step Size

Note that the Hessian of the KL-proximal term is diagonal with

$$\frac{\partial^2 \mathbf{D}^t \cdot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t)}{\partial q_{i,l}^2}|_{\mathbf{q}=\mathbf{q}^t} = \frac{d_{i,l}^t}{q_{i,l}^t} \ . \tag{4.17}$$

Therefore, when some of the $q_{i,l}$s get close to 0, the elements of the Hessian may become very large, especially when using a constant value for the $d_{i,l}^t$ as suggested above. When that happens, the local KL-approximation remains a valid upper bound of the objective function, but not a tight enough one, which results in step sizes that are too small for fast convergence.

This can be reduced by choosing a matrix $\mathbf{D}^t$ that compensates for this. A simple way to do this would be to scale the $d_{i,l}^t$ proportionally to $\max(q_{i,0}, \ldots, q_{i,L_i-1})$ to start compensating for diagonal terms. However, this method is still sub-optimal because it ignores the fact that all our variables lie inside the simplex $\mathcal{M}$. A better alternative is to bound from below the proximal term by a quadratic function, but on $\mathcal{M}$ rather than on $\mathbb{R}^n$.

Note that, we only apply this method to the binary case, for which we set

$$d_{i,0}^t = d_{i,1}^t = q_{i,0}^t q_{i,1}^t \cdot d \ , \tag{4.18}$$

were d is an additional parameter that should be set close to L. Extending this approach to the multi-label case will be a topic for future work. In Section 4.2.4, we provide a different alternative to performing adaptive anisotropic updates in all settings.

Intuitively, when the current parameters are close to the borders of the simplex, the mean parameters are less sensitive to natural parameters, which, therefore, need less damping. We demonstrate in our experiments that it provides a way to choose the step size without tuning.

4.2.4 Momentum

Our approach can easily be extended to incorporate techniques that are known to speed-up gradient descent and help to avoid local minima, such as the classic momentum method [119] or the more recent ADAM technique [81]. The momentum method involves averaging the gradients of the objective $f(\mathbf{x})$ over the iterations in a *momentum* vector $\mathbf{m}$ and use it as the direction for the update instead of simply following the current gradient. To integrate it into our framework, we replace the gradient $\nabla\mathcal{E}$ in Eq. 4.13 by its rolling

exponentially weighted average $\mathbf{m}$ computed as

$$\mathbf{m}^{t+1} = \gamma_1 \mathbf{m}^t + (1-\gamma_1)\nabla\mathcal{E}(\mathbf{q}^t) , \tag{4.19}$$

with the exponential decay parameter $\gamma_1 \in [0;1]$. This substitution brings the following update rule

$$\theta_{i,l}^{t+1} = \eta \cdot m_{i,l}^t + (1-\eta)\cdot\theta_{i,l}^t . \tag{4.20}$$

In what follows, we refer to this approach as `OURS-MOMENTUM`.

4.2.5 ADAM

The ADAM method [81] has become very popular in deep learning. Our framework makes it easy to use for mean-field inference as well by appropriately choosing the matrix $\mathbf{D}^t$ at each step and combining it with the momentum technique.

We define the averaged second moment vector $\mathbf{v}$ of the natural gradient as

$$v_{i,l}^{t+1} = \gamma_2[\theta_{i,l}^t + \nabla\mathcal{E}(\mathbf{q}^t)_{i,l}]^2 + (1-\gamma_2)v_{i,l}^t , \tag{4.21}$$

where $\mathbf{v}$ is initialized to a strictly positive value and $\gamma_2 \in [0;1]$ is an exponential memory parameter for $\mathbf{v}$.

Then, the $\mathbf{D}^t$ matrix is defined through each of its diagonal entries as

$$d_{i,l}^t = \sqrt{v_{i,l}^{t+1}}d + \epsilon - 1 , \tag{4.22}$$

where ϵ is a fixed parameters and d controls the damping. We will refer to this method as `OURS-ADAM`.

Intuitively it is good at exploring parameter space thanks to a form of auto-annealing of the gradient. The natural gradient $\boldsymbol{\theta}_t + \nabla\mathcal{E}(\mathbf{q}^t)$ is zero at a local minimum of the objective function [68]. Therefore, close to a minimum, the proximal term $\mathbf{D}^t$ becomes small, thus allowing more exploration of the space. On the other hand, after a long period of exploration with large natural gradients, more damping will tend to make the algorithm converge.

4.3 Evaluation

In this section, we evaluate our method on a variety of inference problems and demonstrate that in most cases it yields faster convergence and better minima.

4.3.1 Baselines and Variants

We compare several variants of our approach to some of the baselines we introduced in the related work section. The baselines we consider are as follows:

- `SWEEP`. As discussed in Section 4.1.2, it involves sequential coordinate descent [14] and is not always computationally tractable for large problems.
- `ADHOC`. As discussed in Section 4.1.2, it performs parallel updates with the *ad hoc* damping parameter η of Eq. 4.9 chosen manually.
- `FULL-PARALLEL`. As also discussed in Section 4.1.2, it relies on the inference described in [89]. For example, the popular `densecrf` framework [88] uses this approach.

We compare to these the following variants of our approach:

- `OURS-FIXED`. Damping occurs in the space of natural parameters instead of mean ones as described in Section 4.2.2.
- `OURS-ADAPTIVE`. Adaptive and anisotropic damping in the space of natural parameters as described in Section 4.2.3.
- `OURS-MOMENTUM`. Similar to `OURS-ADAPTIVE`, but using the momentum method instead of ordinary gradient descent, as described in Section 4.2.4. We use the same parameter value $\gamma_1 = 0.95$ for all datasets.
- `OURS-ADAM`. Similar to `OURS-ADAPTIVE` but using the ADAM method instead of ordinary gradient as described in Section 4.2.5. We use the same parameters as in the original publication [81], $\gamma_1 = 0.99$, $\gamma_2 = 0.999$ and ϵ =1E-8 for all datasets.

All four methods involve a parameter $\eta = \frac{1}{1+d}$, defined in Eq. 4.16 for `OURS-FIXED`, Eq. 4.18 for `OURS-ADAPTIVE`, Eq. 4.20 for `OURS-MOMENTUM` and Eq. 4.22 for `OURS-ADAM`. Additionally, in Section 4.3.3 and Figure 4.2 we demonstrate that our method is less sensitive to the choice of this parameter than its competitors.

4.3.2 Experimental Setup

We evaluated all the methods first on a set of standardized benchmarks [54]: `DBN`, containing 108 instances of deep belief networks (on average 920 variables), `GRID`, containing

21 instances of two-dimensional grids (1600 variables), and `SEG`, containing 100 instances of segmentation problems (230 variables), where each instance is represented as a binary pairwise random field.

We then consider three realistic Computer Vision tasks that all involve minimizing a functional of the form given in Eq. 4.7. We describe them below.

Characters Inpainting We consider character inpainting, formulated as a binary pairwise random field, Decision Tree Fields (DTF, [115]). The dataset contains 100 test instances of occluded characters, and the goal is to restore the occluded part, as shown in the last row of Figure 4.1. We use pre-computed potentials provided by the authors of [115]. Note, that this model consists of data-driven potentials, and includes both short and long-range interactions, which makes it particularly interesting from the optimization perspective.

People Detection We consider detecting upright people in a multi-camera settings, using the Probabilistic Occupancy Map approach (POM, [52]), that relies on a random field with high-order repulsive potentials, which models background subtraction signal given the presences of people in the environment. We evaluate it on the ISSIA [40] dataset, which contains 3000 frames of a football game, captured by 6 cameras located on two sides of the field. The original work [52] does not explicitly mention it, but the publicly available implementation uses the `ADHOC` damping method. We implement all our methods and remaining baselines directly in this code of [52].

Semantic Segmentation We consider semantic segmentation on PASCAL VOC 2012 dataset [48], which defines 20 object classes and 1 background class. We based our evaluation on DeepLab-CRF model [22], which is currently one of the best-performing methods. This model uses CNNs to obtain unary potentials, and then employs `densecrf` of [89] with dense pairwise potentials. However, this basic CRF model does not contain any strong repulsive terms, and thus we expect `densecrf`'s standard inference, `FULL-PARALLEL`, to work well. To improve performance, we additionally introduced co-occurrence potentials [156], which, as we will show, violate the conditions assumed in `densecrf`, but can still be successfully handled by our method. Intuitively, these co-occurrence terms put priors on the sets of classes that can appear together. We made minor modifications of `densecrf` to support both our inference and co-occurrence potentials.

We performed all the experiments on Intel(R) Xeon(R) CPU E5-2680 2.50GHz, and a GPU GeForce GTX TITAN X (12GB GRAM).

4.3.3 Comparative Results

In order to understand how the methods behave in practical settings, when the available computational time is limited, we evaluate all methods for several computational *budgets*. The shortest budget corresponds to the early-stopping scenario after few iterations, the longest one roughly models the time until convergence, and the middle one is around 20-30% of the longest.

method	0.05s	0.30s	1.00s
SWEEP	-112.94	-2088.07	-2138.13
FULL-PARALLEL	-1952.52	-1951.54	-1942.86
ADHOC	-2047.31	-2047.31	-2047.31
OURS-FIXED	-2081.91	-2081.91	-2081.91
OURS-ADAPTIVE	-2125.48	-2130.61	-2130.61
OURS-MOMENTUM	**-2260.98**	**-2362.14**	**-2374.51**
OURS-ADAM	-2107.98	-2107.93	-2107.93

Table 4.1: Results for KL minimization for the DBN (deep belief networks) benchmark dataset [54]. All the numbers are KL divergence (lower is better) averaged over the instances.

method	0.05s	0.30s	1.00s
SWEEP	-5540.59	-16675.55	-18592.26
FULL-PARALLEL	-2564.39	-2777.33	-2439.08
ADHOC	-18345.42	-18348.80	-18349.03
OURS-FIXED	-18213.81	-18219.42	-18219.45
OURS-ADAPTIVE	-18245.93	-18252.48	-18252.48
OURS-MOMENTUM	-18143.48	**-19074.45**	**-19184.37**
OURS-ADAM	**-18617.06**	-18732.59	-18740.36

Table 4.2: Results for KL minimization for GRID (two-dimensional grids) benchmark dataset [54]. All the numbers are KL divergence (lower is better) averaged over the instances.

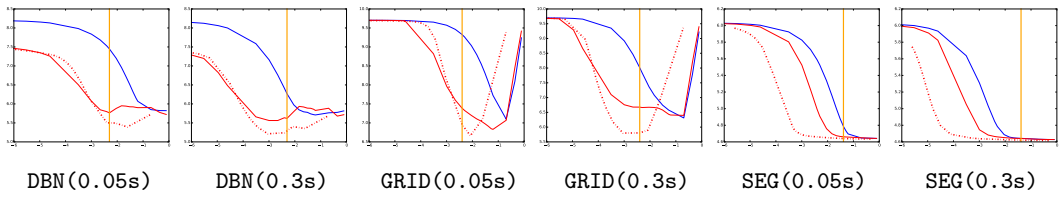
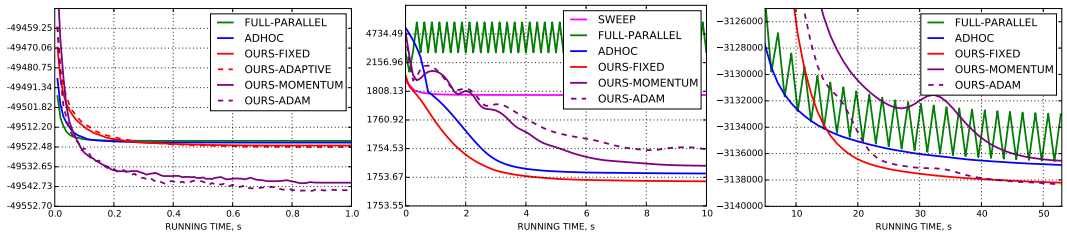


Figure 4.2: Sensitivity of OURS-FIXED (red) and OURS-ADAPTIVE (dashed red) vs ADHOC (blue) to the damping parameter $\eta = \frac{1}{1+d}$. We report KL-divergence (lower is better) vs the value of the parameter, both in log-space.

method	0.05s	0.30s	1.00s
SWEEP	78.81	75.50	75.50
FULL-PARALLEL	75.66	75.66	75.66
ADHOC	76.10	75.66	75.66
OURS-FIXED	77.17	75.61	75.61
OURS-ADAPTIVE	77.68	75.64	75.61
OURS-MOMENTUM	74.35	73.75	73.75
OURS-ADAM	**72.37**	**72.32**	**72.32**

Table 4.3: Results for KL minimization for the SEG (binary segmentation) benchmark datasets [54]. All the numbers are KL divergence (lower is better) averaged over the instances.



(a) Characters Inpainting (b) People Detection (c) Semantic Segmentation

Figure 4.3: Convergence results. (a) OURS-ADAM and OURS-MOMENTUM converge very fast to a much better minima. (b) OURS-FIXED outperforms ADHOC both in terms of speed of convergence and the value of the objective. (c) OURS-ADAM and OURS-FIXED show the best performance. The former converges a bit slower, but in the end provide slightly better minima. ADHOC for this dataset converges rather fast, but fails to find a better optima.

Benchmarks Quantitative results for the benchmarks are given in Tables 4.1-4.3. Our methods systematically outperform the ADHOC damping method. The SWEEP method usually provides good performance, but is generally slow due to its sequential nature.

Figure 4.2 shows that our methods are less sensitive to damping parameter changes than ADHOC. In Figure 4.2, the vertical orange lines corresponds to the choice of the damping parameter according to $d = L$, which can be computed directly by the power-method. Interstingly, for the GRID dataset, which includes strong repulsive potentials, algorithms do not produce reasonable results when no damping is applied. On the other hand, for the segmentation task, SEG, all the algorithms work well even without damping, in accordance with the results of [89] or Section 4.2.2.

Characters Inpainting Quantitative results in terms of average pixel accuracy and KL-divergence are given in Table 4.4 and Figure 4.3 (a). Our method, especially when used with more advanced gradient descent schemes, outperforms all the baselines. SWEEP

	0.05s		0.3s		3s	
method	KL	PA	KL	PA	KL	PA
SWEEP	-6342.56	54.57	-25233.54	58.38	-49519.33	62.50
FULL-PARALLEL	**-49516.98**	60.99	-49519.27	62.00	-49519.33	62.05
ADHOC	-49514.27	61.46	-49520.09	62.15	-49520.20	62.17
OURS-FIXED	-49505.59	60.99	-49520.33	62.26	-49521.71	62.35
OURS-ADAPTIVE	-49503.43	60.93	-49520.14	62.32	-49522.49	62.60
OURS-MOMENTUM	-49513.57	63.69	-49536.67	65.26	-49540.76	65.95
OURS-ADAM	-49516.02	**65.36**	**-49538.84**	**67.03**	**-49544.58**	**67.12**

Table 4.4: Results for characters inpainting problem [115] based on DTFs. PA is the pixel accuracy for the occluded region (bigger is better). Our methods outperform the baselines by a margin of 3-5%. Since FULL-PARALLEL is not damped, it gets to low KL-divergence value quickly, however the actual solution is significantly worse.

	0.5s		1.3s		5s	
method	KL	MODA	KL	MODA	KL	MODA
SWEEP	1865.43	**0.630**	1795.66	0.656	1795.60	0.656
FULL-PARALLEL	2573.79	0.000	2573.79	0.000	8500.90	0.030
ADHOC	2573.79	0.308	1760.02	0.781	1753.71	**0.829**
OURS-FIXED	**1783.63**	0.626	**1754.55**	**0.802**	**1753.63**	**0.829**
OURS-MOMENTUM	1931.36	0.040	1797.19	0.650	1753.83	0.826
OURS-ADAM	2008.52	0.021	1813.66	0.501	1754.52	0.824

Table 4.5: Results for people detection task [40] based on POM [52]. OURS-FIXED outperforms the baselines and adaptive methods. This means that this problem does not require more sophisticated parameter exploration techniques.

shows relatively good performance, but does not scale as well in terms of the running time. See the bottom row of Figure 4.1 for an example of a result.

People Detection Quantitative results, presented in Table 4.5 and Figure 4.3 (b), demonstrate that our method with a fixed step size, OURS-FIXED, brings both faster convergence and better performance. Thanks to our optimization scheme, the time required to get a Multiple Object Detection Accuracy (MODA, [13]) within 3% of the value at convergence is reduced by a factor of two. This can be of big practical importance for surveillance applications of the algorithm [6, 12], in which it is required to run in real-time. SWEEP exhibits much worse performance than our parallel method because of its greedy behavior.

	5s		15s		50s	
method	KL	I/U	KL	I/U	KL	I/U
FULL-PARALLEL [o]	–	67.18	–	67.70	–	68.00
OURS-ADAM [o]	–	66.45	–	67.50	–	68.07
FULL-PARALLEL	**-3129799**	67.21	-3134437	67.72	-3133010	68.01
ADHOC	-3129469	67.19	-3134557	67.73	-3136865	68.04
OURS-FIXED	-3100079	**67.76**	**-3135225**	**68.18**	-3138206	68.44
OURS-MOMENTUM	-3060405	66.20	-3128121	67.39	-3136543	68.18
OURS-ADAM	-3091787	67.08	-3131624	68.02	**-3138335**	**68.47**

Table 4.6: Results for semantic segmentation problem [48] based on DeepLab-CRF [22]. For all the budgets, our method obtains better segmentation accuracy. Again, FULL-PARALLEL obtains lower KL faster, with a price of reduced performance. On the top, we provide results for the original DeepLab-CRF model without co-occurrence potentials (denoted by [o]), for which the KL divergence has therefore a different meaning and is not shown.

Semantic Segmentation Quantitative results are presented in Table 4.6 and Figure 4.3 (c). We observe that a similar oscillation issue as noted by [156] starts happening when the FULL-PARALLEL method is used in conjunction with co-occurrence potentials, producing even worse results than without those. Using our convergent inference method fixes oscillations and provides an improvement of 0.5% in the average Intersection over Union measure (I/U) compared to the basic method without co-occurrence. This is a significant improvement that would be sufficient to increase the position of an algorithm by 2 or 3 places in the official ranking [48]. What it represents is a big improvement in performance, as the ones shown in Figure 4.1, for at least 30-40 images out of total 1449. Note also, that we obtain this improvement with minimal changes in the original code. By contrast, authors [22] get similar or smaller improvements by significantly augmenting the training set or by exploiting multi-scale features, which leads to additional computational burden.

4.4 Discussion

We have presented a principled and efficient way to do parallel mean-field inference in discrete random fields. We have demonstrated that proximal gradient descent is a powerful theoretical framework for mean-field inference, which unifies and sheds light on existing approaches. Moreover, it naturally allows to incorporate existing adaptive gradient descent techniques, such as ADAM, to mean-field methods. As shown in our experiments, it often brings dramatic improvements in performance. Additionally, we have demonstrated, that our approach is less sensitive to the choice of parameters.

Our method makes it possible to use variational mean-field inference with a wider range of potential functions, which was previously unachievable due to the lack of convergent optimization. Thus, there is a large amount of possible future applications of our approach, especially in the tasks where higher-order and repulsive potentials can be useful, not only in segmentation, but also in object localization.

5 Multi-Human Scene Understanding

Human social behavior can be characterized by "social actions" – an individual act which nevertheless takes into account the behaviour of other individuals – and "collective actions" taken together by a group of people with a common objective. For a machine to perceive both of these actions, it needs to develop a notion of collective intelligence, *i.e.*, reason jointly about the behaviour of multiple individuals. In this chapter, we propose a method to tackle such intelligence. Given a sequence of image frames, our method jointly locates and describes the social actions of each individual in a scene as well as the collective actions (see Figure 5.1). This perceived social scene representation can be used for sports analytics, understanding social behaviour, surveillance, and social robot navigation.

Recent methods for multi-person scene understanding take a sequential approach [31, 69, 123]: i) each person is detected in every given frame; ii) these detections are associated over time by a tracking algorithm; iii) a feature representation is extracted for each individual detection; and finally iv) these representations are joined via a structured model. Whereas the aforementioned pipeline seems reasonable, it has several important drawbacks. First of all, the vast majority of state-of-the-art detection methods do not use any kind of joint optimization to handle multiple objects, but rather rely on heuristic post-processing, and thus are susceptible to greedy non-optimal decisions. Second, extracting features individually for each object discards a large amount of context and interactions, which can be useful when reasoning about collective behaviours. This point is particularly important because the locations and actions of humans can be highly correlated. For instance, in team sports, the location and action of each player depend on the behaviour of other players as well as on the collective strategy. Third, having independent detection and tracking pipelines means that the representation used for localization is discarded, whereas re-using it would be more efficient. Finally, the sequential approach does not scale well with the number of people in the scene, since it requires multiple runs for a single image.

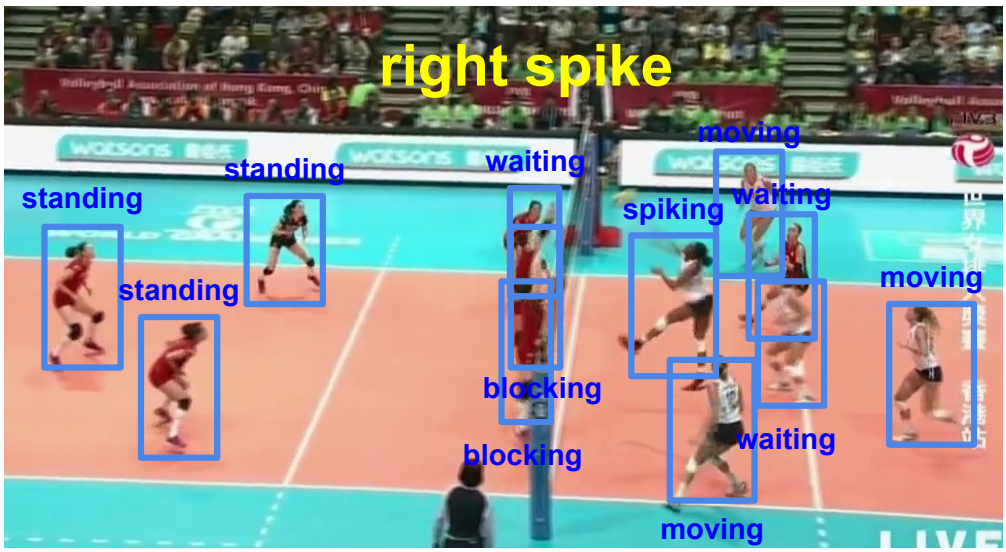


Figure 5.1: Jointly reasoning on social scenes. Our method takes as input raw image sequences and produces a comprehensive social scene interpretation: locations of individuals (as bounding boxes), their individual social actions (e.g., “blocking"), and the collective activity (“right spike" in the illustrated example).

Our method aims at tackling these issues. Inspired by recent work in multi-class object detection [125, 126] and image labelling [73], we propose a single architecture that jointly localizes multiple people, and classifies the actions of each individual as well as their collective activity. Our model produces all the estimates in a single forward pass and requires neither external region proposals nor pre-computed detections or tracking assignments.

Our contributions can be summarized as follows:

- We propose a unified framework for social scene understanding by simultaneously solving three tasks in a single feed forward pass through a Neural Network: multi-person detection, individual’s action recognition, and collective activity recognition. Our method operates on raw image sequences and relies on joint multi-scale features that are shared among all the tasks. It allows us to fine-tune the feature extraction layers early enough to enable the model to capture the context and interactions.
- We introduce a novel multi-object detection scheme, inspired by the classical work on Hough transforms. Our scheme relies on probabilistic inference that jointly refines the detection hypotheses rather than greedily discarding them, which makes our predictions more robust.
- We present a person-level matching Recurrent Neural Network (RNN) model to propagate information in the temporal domain, while not having access to the the

trajectories of individuals.

In Section 5.3, we show quantitatively that these components contribute to the better overall performance. Our model achieves state-of-the-art results on challenging multi-person sequences, and outperforms existing approaches that rely on the ground truth annotations at test time. We demonstrate that our novel detection scheme is on par with the state-of-the art methods on a large-scale dataset for localizing multiple individuals in crowded scenes. Our implementation will be made publicly available.

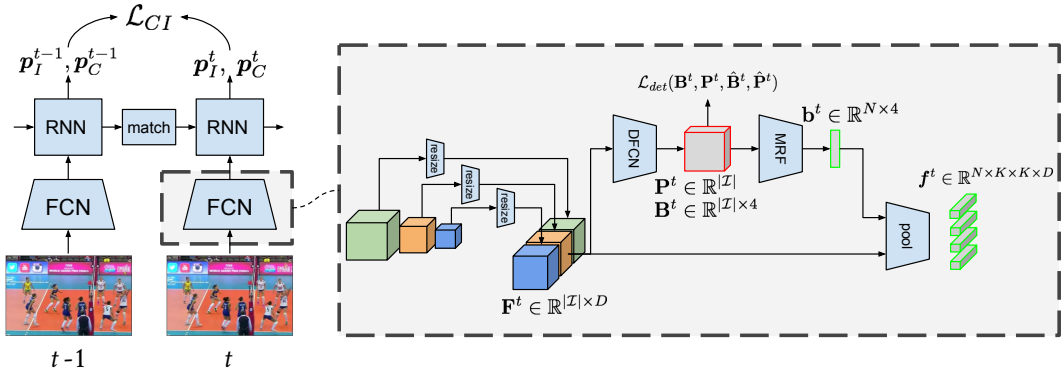


Figure 5.2: General overview of our architecture. Each frame of the given sequence is passed through a fully-convolutional network (FCN) to produce a multi-scale feature map $\mathbf{F}^t$, which is then shared between the detection and action recognition tasks. Our detection pipeline is another fully-convolutional network (DFCN) that produces a dense set of detections $\mathbf{B}^t$ along with the probabilities $\mathbf{P}^t$, followed by inference in a hybrid MRF. The output of the MRF are reliable detections $\mathbf{b}^t$ which are used to extract fixed-sized representations $\boldsymbol{f}^t$, which are then passed to a matching RNN that reasons in the temporal domain. The RNN outputs the probability of an individual's action, $\boldsymbol{p}_I$, and the collective activity, $\boldsymbol{p}_c$ across time. Note that $\mathcal{L}_{det}$ (Eq. 5.3) is the loss function for the detections, and $\mathcal{L}_{CI}$ (Eq. 5.14) is the loss function for the individual and collective actions.

5.1 Related Work

The main focus of this chapter is creating a unified model that can simultaneously detect multiple individuals and recognize their individual social actions and collective behaviour. In what follows, we give a short overview of the existing work on these tasks.

Multi-object detection - There already exists large body of research in the area of object detection. Most of the current methods either rely on a sliding window approach [139, 167], or on the object proposal mechanism [60, 126], followed by a CNN-based classifier. The vast majority of those state-of-the-art methods do not reason jointly on the presence of multiple objects, and rely on very heuristic post-processing steps to get the final

detections. A notable exception is the ReInspect [148] algorithm, which is specifically designed to handle multi-object scenarios by modeling detection process in a sequential manner, and employing a Hungarian loss to train the model end-to-end. We approach this problem in a very different way, by doing probabilistic inference on top of a dense set of detection hypotheses, while also demonstrating state-of-the-art results on challenging crowded scenes. Another line of work that specifically focuses on joint multi-person detection [6, 52, 166] uses generative models, however, those methods require multiple views or depth maps and are not applicable in monocular settings.

Action recognition - A large variety of methods for action recognition traditionally rely on handcrafted features, such as HOG [29, 162], HOF [95] and MBH [159]. More recently, data-driven approaches based on deep learning have started to emerge, including methods based on 3D CNNs [72] and multi-stream networks [50, 144]. Some methods [145, 160], exploit the strengths of both handcrafted features and deep-learned ones. Most of these methods rely in one way or another on temporal cues: either through having a separate temporal stream [50, 145], or directly encoding them into compact representations [95, 159, 159]. Yet another way to handle temporal information in a data-driven way is Recurrent Neural Networks (RNNs). Recently, it has received a lot of interest in the context of action recognition [38, 42, 144, 155]. All these methods, however, are focusing on recognizing actions for single individuals, and thus are not directly applicable in multi-person settings.

Collective activity recognition - Historically, a large amount of work on collective activity recognition relies on graphical models defined on handcrafted features [3, 23, 24, 25, 77, 78]. The important difference of this type of methods with the single-person action recognition approaches is that they explicitly enforce simultaneous reasoning on multiple people. The vast majority of the state-of-the-art methods for recognizing multi-person activities thus also rely on some kind of structured model, that allows sharing information between representations of individuals. However, unlike earlier handcrafted methods, the focus of the recent developments has shifted towards merging the discriminative power of neural networks with structured models. In [31], authors propose a way to refine individual estimates obtained from CNNs through inference: they define a trainable graphical model with nodes for all the people and the scene, and pass messages between them to get the final scene-level estimate. In [69], authors propose a hierarchical model that takes into account temporal information. The model consists of two LSTMs: the first operates on person-level representations, obtained from a CNN, which are then max pooled and passed as input to the second LSTM capturing scene-level representation. [123] explores a slightly different perspective: authors notice that in some settings, the activity is defined by the actions of a single individual and propose a soft attention mechanism to identify her. The complete model is very close to that of [69], except that the attention pooling is used instead of a max pool. All of those methods are effective, however, they start joint reasoning in late inference stages, thus possibly discarding useful context information.

Moreover, they all rely on ground truth detections and/or tracks, and thus do not really solve the problem end-to-end.

Our model builds upon the existing work in that it also relies on the discriminative power of deep learning, and employs a version of person-level temporal model. It is also able to implicitly capture the context and perform social scene understanding, which includes reliable localization and action recognition, all in a single end-to-end framework.

5.2 Method

Our main goal is to construct comprehensive interpretations of social scenes from raw image sequences. To this end, we propose a unified way to jointly detect multiple interacting individuals and recognize their collective and individual actions.

5.2.1 Overview

The general overview of our model is given in Figure 5.2. For every frame $\mathbf{I}^t \in \mathbb{R}^{H_0 \times W_0 \times 3}$ in a given sequence, we first obtain a dense feature representation $\mathbf{F}^t \in \mathbb{R}^{|\mathcal{I}| \times D}$, where $\mathcal{I} = \{1, \ldots, H \times W\}$ denotes the set of all pixel locations in the feature map, $|\mathcal{I}| = H \times W$ is the number of pixels in that map, and D is the number of features. The feature map $\mathbf{F}^t$ is then shared between the detection and action recognition tasks. To detect, we first obtain a preliminary set of detection hypotheses, encoded as two dense maps $\mathbf{B}^t \in \mathbb{R}^{|\mathcal{I}| \times 4}$ and $\mathbf{P}^t \in \mathbb{R}^{|\mathcal{I}|}$, where at each location $i \in \mathcal{I}$, $\mathbf{B}_i^t$ encodes the coordinates of the bounding box, and $\mathbf{P}_i^t$ is the probability that this bounding box represents a person. Those detections are refined jointly by inference in a hybrid Markov Random Field (MRF). The result of the inference is a smaller set of N reliable detections, encoded as bounding boxes $\mathbf{b}^t \in \mathbb{R}^{N \times 4}$. These bounding boxes are then used to smoothly extract fixed-size representations $\boldsymbol{f}_n^t \in \mathbb{R}^{K \times K \times D}$ from the feature map $\mathbf{F}^t$, where K is the size of the fixed representation in pixels. Representations $\boldsymbol{f}_n^t$ are then used as inputs to the matching RNN, which merges the information in the temporal domain. At each time step t, RNN produces probabilities $\boldsymbol{p}_{I,k}^t \in \mathbb{R}^{N_I}$ of individual actions for each detection $\mathbf{b}_n^t$, along with the probabilities of collective activity $\boldsymbol{p}_C^t \in \mathbb{R}^{N_C}$, where N_I, N_c denote respectively the number of classes of individual and collective actions. In the following sections, we will describe each of these components in more detail.

5.2.2 Joint Feature Representation

We build upon the Inception architecture [150] for getting our dense feature representation, since it does not only demonstrate good performance but is also more computationally

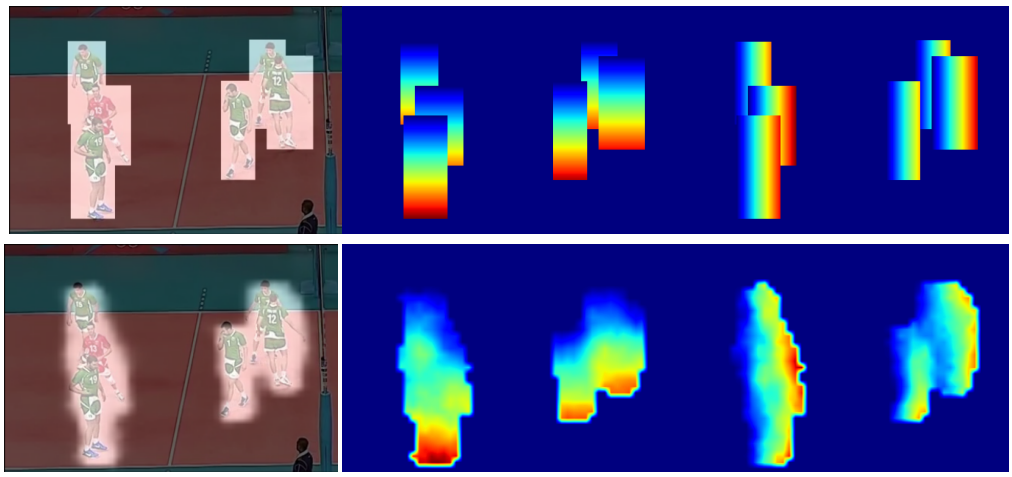


Figure 5.3: Example of ground truth (top) and predicted (bottom) maps. We show segmentation map $\mathbf{P}$ projected on the original image, followed by two out of four channels of the regression map $\mathbf{B}$, which encode respectively vertical and horizontal displacement from the location i to one of the bounding box corners.

efficient than some of the more popular competitors [92, 143].

One of the challenges when simultaneously dealing with multiple tasks is that representations useful for one task may be quite inefficient for another. In our case, person detection requires reasoning on the type of the object, whereas discriminating between actions can require looking at lower-level details. To tackle this problem, we propose using multi-stage features: instead of simply using the final convolutional layer, we produce our dense feature map $\mathbf{F} \in \mathbb{R}^{|\mathcal{I}| \times D}$ (here and later t is omitted for clarity) by concatenating multiple intermediate activation maps. Since they do not have fitting dimensions, we resize them to the fixed size $|\mathcal{I}| = H \times W$ via differentiable bilinear interpolation. Note that similar approaches have been very successful for semantic segmentation [63, 102], when one has to simultaneously reason about the object class and its boundaries.

5.2.3 Dense Detections

Given the output of the feature extraction stage, the goal of the detection stage is to generate a set of reliable detections, that is, a set of bounding box coordinates with their corresponding confidence scores. We do it in a dense manner, meaning that, given the feature map $\mathbf{F} \in \mathbb{R}^{|\mathcal{I}| \times D}$, we produce two dense maps $\mathbf{B} \in \mathbb{R}^{|\mathcal{I}| \times 4}$ and $\mathbf{P} \in \mathbb{R}^{|\mathcal{I}|}$, for bounding boxes coordinates and presence probability, respectively. Essentially, $\mathbf{P}$ represents a segmentation mask encoding which parts of the image contain people, and $\mathbf{B}$ represents the coordinates of the bounding boxes of the people present in the scene, encoded relative to the pixel locations. This is illustrated by Figure 5.3.

We can interpret this process of generating $\mathbf{P}, \mathbf{B}$ from $\mathbf{F}$ in several different ways. With respect to recent work on object detection [60, 125, 126], it can be seen as a fully-convolutional network that produces a dense set of object proposals, where each pixel of the feature map $\mathbf{F}$ generates a proposal. Alternatively, we can see this process as an advanced non-linear version of the Hough transform, similar to Hough Forests [10, 57]. In these methods, each patch of the image is passed through a set of decision trees, which produce a distribution over potential object locations. The crucial differences with the older methods are, first, leveraging deep neural network as a more powerful regressor and, second, the ability to use large contexts in the image, in particular to reason jointly about parts.

Let us now introduce $\mathbf{B}$ and $\mathbf{P}$ more formally, by defining how we convert the given ground truth object locations into dense ground truth maps $\hat{\mathbf{B}}, \hat{\mathbf{P}}$. For each image $\mathbf{I}$, the detection ground truth is given as a set of bounding boxes $\{(y_0, x_0, y_1, x_1)_1, \dots, \}$. To obtain the value for the specific location $i = (i_y, i_x) \in \mathcal{I}$ of the ground truth probability map $\hat{\mathbf{P}}$, we set $\hat{\mathbf{P}}_i = 1$ if $y_0 \leq i_y \leq y_1, x_0 \leq i_x \leq x_1$ for any of the ground truth boxes, and $\hat{\mathbf{P}}_i = 0$ otherwise. For the regression map, each location i represents a vector $\hat{\mathbf{B}}_i = (t_{y0}, t_{x0}, t_{y1}, t_{x1})$, where:

$$t_{y0} = (i_y - y_0)/s_y,\ t_{x0} = (i_x - x_0)/s_x\ , \tag{5.1}$$

$$t_{y1} = (y_1 - i_y)/s_x,\ t_{x1} = (x_1 - i_x)/s_y\ , \tag{5.2}$$

where s_y, s_x are scaling coefficients that are fixed, and can be taken either as the maximum size of the bounding box over the training set, or the size of the image. Ultimately, our formulation makes it possible to use ground truth instance-level segmentation masks to assign each i to one of the ground truth instances. However, since these masks are not available, and there can be multiple ground truth bounding boxes that contain i, we assign each i to the bounding box with the highest y_0 coordinate, as shown in Figure 5.3. Note that, $\hat{\mathbf{B}}_i$ are only defined only for $i : \hat{\mathbf{P}}_i = 1$, and the regression loss is constructed accordingly.

The mapping from $\mathbf{F}$ to $\mathbf{B}$, $\mathbf{P}$ is a fully-convolutional network, consisting of a stack of two 3×3 convolutional layers with 512 filters and a shortcut connection [65]. We use softmax activation function for $\mathbf{P}$ and ReLU for $\mathbf{B}$. The loss is defined as follows:

$$\begin{aligned} \mathcal{L}_{det} = &- \frac{1}{|\mathcal{I}|} \sum_i \hat{\mathbf{P}}_i \log \mathbf{P}_i + \\ &w_{reg} \frac{1}{\sum_i \hat{\mathbf{P}}_i} \cdot \sum_i \hat{\mathbf{P}}_i ||\hat{\mathbf{B}}_i - \mathbf{B}_i||_2^2\ , \end{aligned} \tag{5.3}$$

where w_{reg} is a weight that makes training focused more on classification or regression. For datasets where classification is easy, such as `volleyball` [69], we set it to $w_{reg} = 10$,

whereas for cluttered scenes with large variations in appearance lower values could be beneficial.

5.2.4 Inference for Dense Detection Refinement

The typical approach to get the final detections given a set of proposals is to re-score them using an additional recognition network and then run non-maxima suppression (NMS) [73,126]. This has several drawbacks. First, if the amount of the proposals is large, the re-scoring stage can be prohibitively expensive. Second, the NMS step itself is by no means optimal, and is susceptible to greedy decisions. Instead of this commonly used technique, we propose using a simple inference procedure that does not require re-scoring, and makes NMS in the traditional sense unnecessary. Our key observation is that instead of making similar hypotheses suppressing each other, one can rather make them refine each other, thus increasing the robustness of the final estimates.

To this end, we define a hybrid MRF on top of the dense proposal maps $\mathbf{B}^*$, which we obtain by converting $\mathbf{B}$ to the global image coordinates. For each hypothesis location $i \in \mathcal{I}$ we introduce two hidden variables, one multinomial Gaussian $\mathbf{X}_i \in \mathbb{R}^4$, and one categorical $A_i \in \mathcal{I}$. $\mathbf{X}_i$ encodes the "true" coordinates of the detection, and A_i encodes the assignment of the detection to one of the hypothesis locations in $\mathcal{I}$. Note that, although this assignment variable is discrete, we formulate our problem in a probabilistic way, through distributions, thus allowing a detection to be "explained" by multiple locations. The joint distribution over $\mathbf{X}_{1:|\mathcal{I}|}, A_{1:|\mathcal{I}|}$ is defined as follows:

$$P(\mathbf{X}_{1:|\mathcal{I}|}, A_{1:|\mathcal{I}|}) \propto \prod_{i,j} \exp\left(-\frac{\mathbb{1}[A_i = j] \cdot ||\mathbf{X}_i - \mathbf{X}_j||_2^2}{2\sigma^2}\right) , \tag{5.4}$$

where σ is the standard deviation parameter, which is fixed.

Intuitively, Eq. 5.4 jointly models the relationship between the bounding box predictions produced by the fully-convolutional network. The basic assumption is that each location $i \in \mathcal{I}$ on the feature map belongs to a single "true" detection location j, which can be equal to i, and the observation $\mathbf{X}_i$ should not be far from the observation $\mathbf{X}_j$ at this "true" location. The goal of inference is to extract those "true" locations and their corresponding predictions by finding the optimal assignments for A_i and values of $\mathbf{X}_i$. In other words, we want to compute marginal distributions $P(\mathbf{X}_i), P(A_i), \forall i \in \mathcal{I}$. Unfortunately, the exact integration is not feasible, and we have to resort to an approximation. We use the mean-field approximation, that is, we introduce the following factorized variational distribution:

$$Q(\mathbf{X}_{1:|\mathcal{I}|}, A_{1:|\mathcal{I}|}) = \prod_i \mathcal{N}(\mathbf{X}_i\,; \boldsymbol{\mu}_i, \sigma^2) \cdot \mathrm{Cat}(A_i\,; \boldsymbol{\eta}_i) , \tag{5.5}$$

where $\boldsymbol{\mu}_i \in \mathbb{R}^4$ and $\boldsymbol{\eta}_i \in \mathbb{R}^{|\mathcal{I}|}$ are the variational parameters of the Gaussian and categorical distributions respectively. Then, we minimize the KL-divergence between the variational distribution Eq. 5.5 and the joint Eq. 5.4, which leads to the following fixed-point updates for the parameters of $Q(\cdot)$:

$$\eta_{ij}^{\tau} \propto -\frac{||\boldsymbol{\mu}_i^{\tau-1} - \boldsymbol{\mu}_j^{\tau-1}||_2^2}{2\sigma^2}, \boldsymbol{\alpha}_i^{\tau} = \text{softmax}(\boldsymbol{\eta}_i^{\tau}) , \tag{5.6}$$

$$\hat{\boldsymbol{\mu}}_i^{\tau} = \sum_j \alpha_{ij} \boldsymbol{\mu}_j^{\tau-1} , \tag{5.7}$$

where $\tau \in \{1, \ldots, \mathcal{T}\}$ is the iteration number, $\boldsymbol{\alpha}_i^{\tau} \in \mathbb{R}^{|\mathcal{I}|}, \sum_j \alpha_{ij}^{\tau} = 1$ is the reparameterization of $\boldsymbol{\eta}_i^{\tau}$. The complete derivation of those updates is provided in Appendix C.

Starting from some initial $\boldsymbol{\mu}^0$, one can now use Eq. 5.6 and Eq. 5.7 until convergence. In practice, we start with $\boldsymbol{\mu}^0$ initialized from the estimates $\mathbf{B}^*$, thus conditioning our model on the observations, and only consider those $i \in \mathcal{I}$, for which the segmentation probability $\mathbf{P}_i > \rho$, where ρ is a fixed threshold. Furthermore, to get $\boldsymbol{\mu}^{\tau}$ we use the following smoothed update for a fixed number of iterations $\mathcal{T}$:

$$\boldsymbol{\mu}_i^{\tau} = (1 - \lambda) \cdot \boldsymbol{\mu}^{\tau-1} + \lambda \cdot \hat{\boldsymbol{\mu}}^{\tau} , \tag{5.8}$$

where λ is a damping parameter that can be interpreted as a step-size [8].

To get the final set of detections, we still need to identify the most likely hypothesis out of our final refined set $\boldsymbol{\mu}^{\mathcal{T}}$. Luckily, since we also have the estimates $\boldsymbol{\alpha}_i^{\mathcal{T}}$ for the assignment variables A_i, we can identify them using a simple iterative scheme similar to that used in Hough Forests [10]. That is, we identify the hypothesis with the largest number of locations assigned to it, then remove those locations from consideration, and iterate until there are no unassigned locations left. The number of assigned locations is then used as a detection score with a very nice interpretation: a number of pixels that "voted" for this detection.

5.2.5 Matching RNN for Temporal Modeling

Previous sections described a way to obtain a set of reliable detections from raw images. However, temporal information is known to be a very important feature when it comes to action recognition [95, 159]. To this end, we propose using a matching Recurrent Neural Network, that allows us to merge and propagate information in the temporal domain.

For each frame t, given a set of N detections $\mathbf{b}_n^t, n \in \{1, \ldots, N\}$, we first smoothly extract fixed-sized representations $\boldsymbol{f}_n^t \in \mathbb{R}^{K \times K \times D}$ from the the dense feature map $\mathbf{F}^t$, using bilinear interpolation. This is in line with the ROI-pooling [126], widely used in

object detection, and can be considered as a less generic version of spatial transformer networks [70], which were also successfully used for image captioning [73]. Those representations $\boldsymbol{f}_n^t$ are then passed through a fully-connected layer, which produces more compact embeddings $\boldsymbol{e}_n^t \in \mathbb{R}^{D_e}$, where D_e is the number of features in the embedded representation. These embeddings are then used as inputs to the RNN units.

We use standard Gated Recurrent Units (GRU [26]) for each person in the sequence, with a minor modification. Namely, we do not have access to the track assignments neither during training nor testing, which means that the hidden states $\boldsymbol{h}_n^t \in \mathbb{R}^{D_h}$ and $\boldsymbol{h}_n^{t+1} \in \mathbb{R}^{D_h}$, where D_h is the number of features in the hidden state, are not necessarily corresponding to the same person. Our solution to this is very simple: we compute the Euclidean distances between each pair of representations at step t and $t-1$, and then update the hidden state based on those distances. A naive version that works well when the ground truth locations are given, is to use bounding box coordinates $\mathbf{b}^t, \mathbf{b}^{t-1}$ as the matching representations, and then update $\boldsymbol{h}_n^t$ by the closest match $\boldsymbol{h}_{n^*}^{t-1}$:

$$n^* = \arg\min_m ||\mathbf{b}_n^t - \mathbf{b}_m^{t-1}||_2^2 \,, \tag{5.9}$$

$$\boldsymbol{h}_n^t = \mathrm{GRU}(\boldsymbol{e}_n^t, \boldsymbol{h}_{n^*}^{t-1}) \,. \tag{5.10}$$

Alternatively, instead of bounding box coordinates $\mathbf{b}^t$, one can use the embeddings $\boldsymbol{e}^t$. This allows the model to learn a suitable representation, which can be potentially more robust to missing/misaligned detections. Finally, instead of finding a *single* nearest-neighbor to make the hidden state update, we can use *all* the previous representations, weighted by the distance in the embedding space as follows:

$$w_{nm}^t \propto \exp(-||\boldsymbol{e}_n^t - \boldsymbol{e}_m^{t-1}||_2^2) \,, \sum_m w_{nm}^t = 1, \tag{5.11}$$

$$\hat{\boldsymbol{h}}^{t-1} = \sum_m w_{nm}^t \boldsymbol{h}_m^{t-1} \,, \tag{5.12}$$

$$\boldsymbol{h}_n^t = \mathrm{GRU}(\boldsymbol{e}_n^t, \hat{\boldsymbol{h}}^{t-1}) \,. \tag{5.13}$$

We experimentally evaluated all of these matching techniques, which we call respectively `boxes`, `embed` and `embed-soft`. We provide results in Section 5.3.

To get the final predictions $\boldsymbol{p}_C^t$ for collective activities, we max pool over the hidden representations $\boldsymbol{h}^t$ followed by a softmax classifier. The individual actions predictions $\boldsymbol{p}_{I,n}^t$ are computed by a separate softmax classifier on top of $\boldsymbol{h}_n^t$ for each detection n. The

loss is defined as follows:

$$\begin{aligned}\mathcal{L}_{CI} = &- \frac{1}{T \cdot N_C} \sum_{t,c} \hat{\boldsymbol{p}}^t_{C,c} \log \boldsymbol{p}^t_{C,c} \\ &- w_I \frac{1}{T \cdot N \cdot N_I} \sum_{t,n,a} \hat{\boldsymbol{p}}^t_{I,n,a} \log \boldsymbol{p}^t_{I,n,a} \ ,\end{aligned} \tag{5.14}$$

where T is the number of frames, N_C, N_I are the numbers of labels for collective and individual actions, N is the number of detections, and $\hat{\boldsymbol{p}}_*$ is the one-hot-encoded ground truth. The weight w_I allows us to balance the two tasks differently, but we found that the model is somewhat robust to the choice of this parameter. In our experiments, we set $w_I = 2$.

5.3 Evaluation

In this section, we report our results on the task of multi-person scene understanding and compare them to multiple baselines. We also compare our detection pipeline to multiple state-of-the-art detection algorithms on a challenging dataset for multi-person detection.

5.3.1 Datasets

We evaluate our framework on the recently introduced `volleyball` dataset [69], since it is the only publicly available dataset for multi-person activity recognition that is relatively large-scale and contains labels for people locations, as well as their collective and individual actions.

This dataset consists of 55 volleyball games with 4830 labelled frames, where each player is annotated with the bounding box and one of the 9 individual actions, and the whole scene is assigned with one of the 8 collective activity labels, which define which part of the game is happening. For each annotated frame, there are multiple surrounding unannotated frames available. To get the ground truth locations of people for those, we resort to the same appearance-based tracker as proposed by the authors of the dataset [69].

5.3.2 Baselines

We use the following baselines and versions of our approach in the evaluation:

- `Inception-scene` - Inception-v3 network [150], pre-trained on ImageNet and fine-tuned to predict collective actions on whole images, without taking into account locations of individuals.

- `Inception-person` - similar to previous baseline, but trained to predict individual actions based on high-resolution fixed-sized images of individual people, obtained from the ground truth detections.

- `HDTM` - A 2-stage deep temporal model model [69], consisiting of one LSTM to aggregate person-level dynamics, and one LSTM to aggregate scene-level temporal information. We report multiple versions of this baseline: the complete version which includes both scene-level and person-level temporal models, `scene`, which only uses scene-level LSTM, and `person`, which only uses person-level LSTM.

- `OURS-single` - A version of our model that does not use an RNN. We report results for ground truth locations, as well as detections produced by our detection pipeline.

- `OURS-temporal` - A complete version of our model with GRU units for temporal modeling. We report results both for ground truth locations and our detections, as well as results for different matching functions.

5.3.3 Implementation Details

All our models are trained using backpropagation using the same optimization scheme: for all the experiments and all datasets, we use stochastic gradient descent with ADAM [81], with the initial learning rate set to 10^{-5}, and fixed hypereparameters to $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$.

We train our model in two stages: first, we train a network on single frames, to jointly predict detections, individual, and collective actions. We then fix the weights of the feature extraction part of our model, and train our temporal RNN to jointly predict individual actions together with collective activities. Note that in fact our model is fully-differentiable, and the reason for this two-stage training is purely technical: backpropagation requires keeping all the activations in memory, which is not possible for a batch of image sequences. The total loss is simply a sum of the detection loss (Eq. 5.3) and the action loss (Eq. 5.14) for the first stage, and the action loss for the second stage. We use a temporal window of length $T = 10$, which corresponds to 4 frames before the annotated frame, and 5 frames after.

The parameters of the MRF are the same for all the experiments. We run inference on the bounding boxes with the probability $\mathbf{P}_i$ above the threshold $\rho = 0.2$, and set the standard deviation $\sigma = 0.005$, step size $\lambda = 0.2$, and the number of iterations $\mathcal{T} = 20$.

Our implementation is based on TensorFlow [1] and its running time for a single sequence of $T = 10$ high-resolution (720x1080) images is approximately 1.2s on a single Tesla-P100 NVIDIA GPU.

5.3.4 Multi-Person Scene Understanding

The quantitative results on the `volleyball` dataset are given in Table 5.1. Whenever available, we report accuracies both for collective action recognition and individual action recognition. For variants of our methods, we report two numbers: when the output of our detection pipeline was used (MRF), and the ground truth bounding boxes (GT). Our method is able to achieve state-of-the-art performance for collective activity recognition even without ground truth locations of the individuals and temporal reasoning. With our matching RNN, performance improvements are even more noticeable. The comparison to `Inception-person`, which was fine-tuned specifically for the single task of individual action recognition, indicates that having a joint representation which is shared across multiple tasks leads to an improvement in average accuracy on individual actions. When we use the output of our detections, the drop in performance is expected, especially since we did not use any data augmentation to make the action recognition robust to imperfect localization. For collective actions, having perfect localization is somewhat less important, since the prediction is based on multiple individuals. In Figure 5.4 we provide some visual results, bounding boxes and actions labels are produced by `OURS-temporal` model with `embed-soft` matching from raw image sequences from the test set.

Method	collective	individual
`Inception-scene` (GT)	75.5	-
`Inception-person` (GT)	-	78.1
`HDTM-scene` [69](GT)	74.7	-
`HDTM-person` [69](GT)	80.2	-
`HDTM` [69](GT)	81.9	-
`OURS-single` (MRF/GT)	83.3 / 83.8	77.8 / 81.1
`OURS-temporal` (MRF/GT)	87.1 / **89.9**	77.9 / **82.4**

Table 5.1: Results on the `volleyball` dataset. We report average accuracy for collective activity and individual actions. For `OURS-temporal` for the ground truth bounding boxes (GT) we report results with the `bbox` matching, and for the detections (MRF) we report results with the `embed` matching.

In Table 5.2 we compare different matching strategies. For the ground truth detections, as expected, simply finding the best match in the bounding box coordinates, `boxes`, works very well. Interestingly, using the `embed` and `embed-soft` matching are beneficial for the performance when detections are used instead of the ground truth. It is also understandable: appearance is more robust than coordinates, but it also means that our model is actually able to capture that robust appearance representation, which might not be absolutely necessary for the prediction in a single frame scenario. Note that, whereas for the collective actions the temporal data seems to help significantly, the improvement for the individual action estimation is very modest, especially for the detections. We

hypothesize that in order to discriminate better between individual actions, it is necessary to look at how the low-level details change, which could be potentially smoothed out during the spatial pooling, and thus they are hard to capture for our RNN.

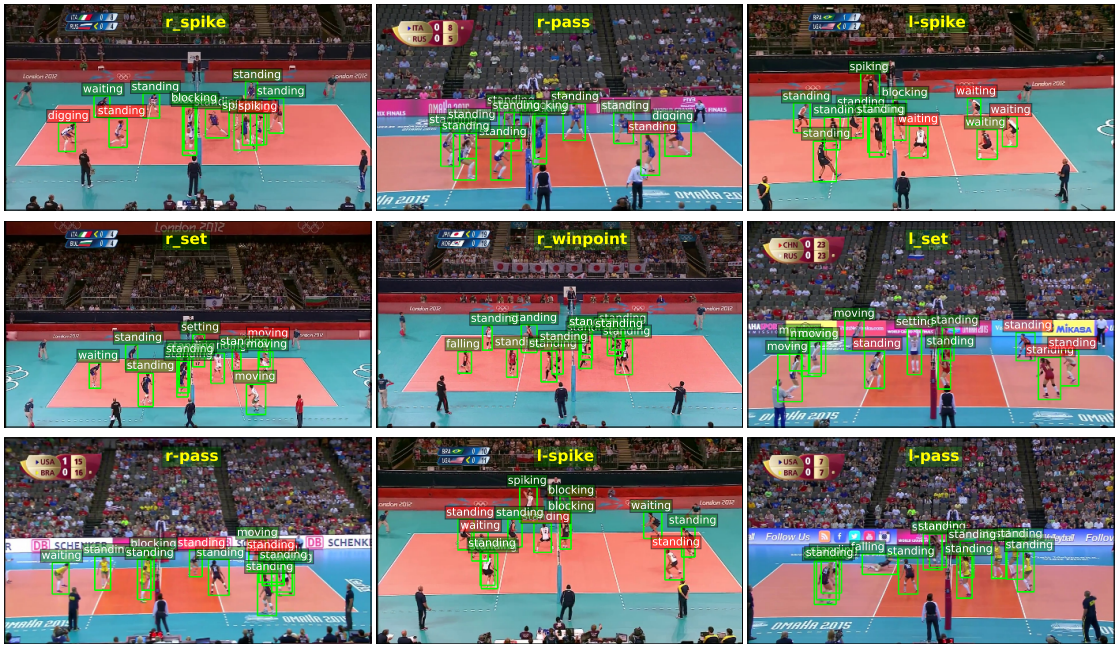


Figure 5.4: Examples of visual results (better viewed in color). Green boxes around the labels correspond to correct predictions, red correspond to mistakes. The bounding boxes in the images are produced by our detection scheme, and obtained in a single pass together with the action labels.

Method	collective	individual
`boxes` (MRF/GT)	82.0 / 89.9	68.6 / **82.4**
`embed` (MRF/GT)	87.1 / 90.0	77.9 / 81.9
`embed-soft` (MRF/GT)	86.2 / **90.6**	77.4 / 81.8

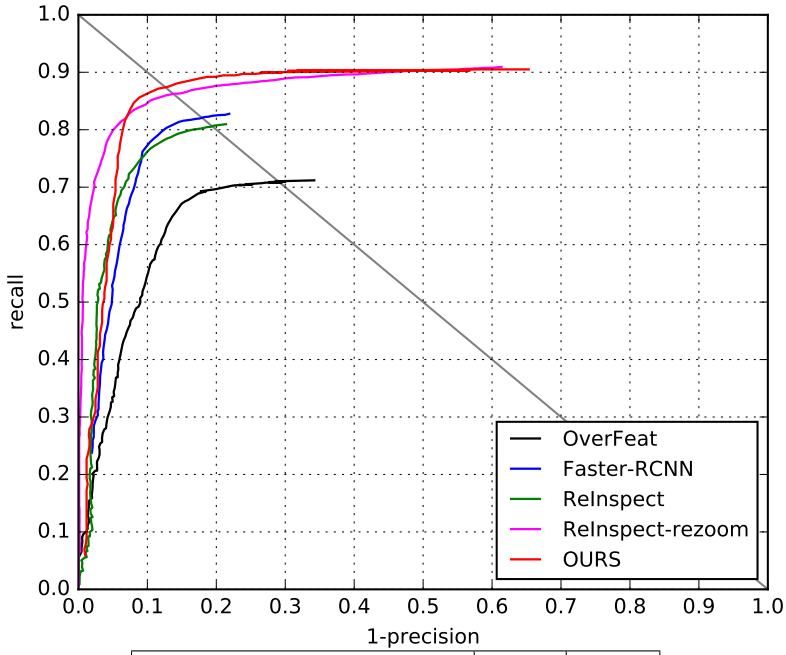
Table 5.2: Comparison of different matching strategies for the `volleyball` dataset. `boxes` corresponds to the nearest neighbour (NN) match in the space of bounding box coordinates, `embed` corresponds to the NN in the embedding space $\boldsymbol{e}$, and `embed-soft` is a soft matching in $\boldsymbol{e}$.

We also conducted experiments to see if our joint detection using MRF is beneficial, and compare it to the traditional non-maxima suppression, both operating on the same dense detection maps. The results for various matching strategies are given in Table 5.3. For all of them, our joint probabilistic inference leads to better accuracy than non-maxima suppression.

Method	collective	individual
`boxes` MRF	82.0	68.6
`boxes` NMS	77.0	68.1
`embed` MRF	**87.1**	**77.9**
`embed` NMS	85.2	76.2
`embed-soft` MRF	86.2	77.4
`embed-soft` NMS	85.1	75.7

Table 5.3: Comparative results of detection schemes on the `volleyball` dataset. We report the average accuracy for the collective and individual action recognition.

5.3.5 Multi-Person Detection



Method	AP	EER
`Overfeat` [139]	0.67	0.71
`Faster-RCNN` [126]	0.79	0.80
`ReInspect` [148]	0.78	0.81
`ReInspect-rezoom` [148]	**0.89**	0.85
`OURS`	0.88	**0.87**

Figure 5.5: Results for multi-person detection on the `brainwash` [148] dataset (better viewed in color).

For completeness, we also conducted experiments for multi-person detection using our dense proposal network followed by a hybrid MRF. Our main competitor is the `ReInspect` algorithm [148], which was specifically designed for joint multi-person detection. We trained and tested our model on the `brainwash` dataset [148], which contains more than

11000 training and 500 testing images, where people are labeled by bounding boxes around their heads. The dataset includes some highly crowded scenes in which there are a large number of occlusions.

Many of the bounding boxes in this dataset are extremely small and thus have very little image evidence, however, our approach allows us to simultaneously look at different feature scales to tackle this issue. We use 5 convolutional maps of the original Inception-v3 architecture to construct our dense representation **F**. We do not tune any parameters on the validation set, keeping them the same as for `volleyball` dataset.

In Figure 5.5 we report average precision (AP) and equal error rate (EER) [47], along with the precision-recall curves. We outperform most of the existing detection algorithms, including widely adopted `Faster-RCNN` [126], by a large margin, and perform very similarly to `ReInspect-rezoom`. One of the benefits of our detection method with respect to the `ReInspect`, is that our approach is not restricted only to detection, and can be also used for instance-level segmentation.

5.4 Discussion

We proposed a unified model for joint detection and activity recognition of multiple people. Our approach does not require any external ground truth detections nor tracks, and demonstrates state-of-the-art performance both on multi-person scene understanding and detection datasets. Future work will apply the proposed framework to explicitly capture and understand human interactions.

6 Face Modeling with Compositional Variational Autoencoders

Building robust and expressive face models is challenging because they must be able to capture deformations at many different scales. These range from large ones to represent the overall shape of specific person's face to small ones to capture subtle expressions such as a smirk or a frown.

Most existing methods can be roughly split into two categories depending on whether they use global linear models [16, 33, 75] or local ones [152, 165]. While the former are simple to use and usually robust to noise and mismatches, the underlying linear space is over-constrained and does not provide sufficient flexibility to represent high-frequency deformations. By contrast local models bring flexibility by separately modeling local deformations. However, they are also more vulnerable to noise and outliers, and can easily produce non-face shapes. Even recent hybrid methods that enforce global anatomical constraints [165] remain limited to person-specific settings and it is not clear how to extend them to capture facial features across multiple identities.

With the advent of Deep Learning, there have been several attempts at using deep nets for data-driven face reconstruction [41, 130, 153]. However, these methods still rely on global linear models, which precludes from performing required multi-scale modeling.

In this chapter, we propose a novel method to model multi-scale face geometry that learns the facial geometry from the data without making any restrictive linear assumptions. Our approach starts with the observation that both global and local linear models can be viewed as specific instances of autoencoders. They can therefore both be incorporated into a generic compositional architecture that combines the strengths of both local and global models, while being completely data-driven. In particular, our approach features a new Variational Autoencoder (VAE) with multiple layers of hidden variables that capture various level of geometrical details. In effect, some network layers capture the low-frequency geometry while others represent high-frequency details.

In the experimental evaluation we demonstrate our model's effectiveness on a variety of fitting tasks, including dense depth data, sparse 2D and 3D correspondences, as well as shape-from-shading reconstruction. We show that it can capture high-quality face geometry even when trained using a database featuring only 16 different people.

In short, our main contribution is a model that encodes facial geometry over a range of scales and generalizes to new identities and arbitrary expressions, while being learned from a small number of different people. The last point is important because creating databases of high-quality meshes that cover a wide range of human expressions and a large number of different identities is both expensive and time-consuming.

6.1 Related Work

One of the main motivations for this work is to demonstrate that it is possible to use deep probabilistic models and variational methods to learn meaningful geometric representations directly from the data. In this section, we therefore first review existing face models and several recent efforts on applying deep learning to data-driven face reconstruction. We then give a very brief introduction into deep probabilistic models with a focus on Variational Autoencoders.

6.1.1 Parametric Face Models

Many different global 3D face parameterizations have been proposed over the years. They include Active Appearance Models (AAM) [27], blendshapes [98], principal components analysis (PCA) derived from a set of training shapes [16, 96], and multilinear models [157]. They have been successfully used to overcome the ambiguities associated with monocular face tracking [15, 30, 33, 46, 56, 99, 138]. However, because they are designed to model the *whole* face at once, it is difficult to use them to represent small details without making them exceedingly large and unwieldy.

Local or region-based shape models have therefore also been proposed to remedy this problem. For example Joshi et al. [75] use a region-based blendshape model for keyframe facial animation and automatically determine the best segmentation using a physical model. Na and Jung [110] use local blendshapes for motion capture retargeting and devise a method for choosing the local regions and their corresponding weighting factors automatically. Tena et al. [152] learn a region-based PCA model based on motion capture data, which allows direct local manipulation of the face. Neumann et al. [112] extract sparse localized deformation components from an animated mesh sequence, for the purpose of intuitive editing as well as statistical processing of the face. Brunton et al. [18] rely on many localized multilinear models to reconstruct faces from noisy or occluded point

cloud data. All these approaches offer more flexibility than the globals models but at the cost of being less constrained to realistically represent human faces.

Wu et al. [165] propose a hybrid approach that combines a local 3D model made of many overlapping patches, which can be locally deformed, and a global model in the form of anatomical constraints that simulate the existence of a skull and jaw bone. This is effective, but it has to be tailored to each individual, and only considers bone structure, while ignoring other types of constraints.

6.1.2 Deep Learning for 3D Face Reconstruction.

Deep models have been successfully used for 3D face reconstruction. In [153], the authors propose a weakly-supervised approach to learning a CNN-based regressor from the space of images into a pre-defined semantic space, which includes global pose and facial expressions, as well as illumination and texture. Similarly, in [129], used a large dataset of artificially rendered face images to train a CNN that maps images into the space of facial geometry. Both these approaches, however, rely on a pre-defined geometry space based on a variation of a bilinear AAM model [27].

By contrast, applying deep generative models to learning a geometric representation has been largely overlooked. The approach of [45] is an exception that relies on deep restricted Boltzmann machines to model the shape of the face. However, that approach does not model the entire facial geometry, but is restricted to represent a sparse set of facial landmarks.

6.1.3 Deep Generative Models

Deep Generative Models, including Variational Autoencoders (VAEs) [82] and Generative Adversarial Networks (GANs) [39, 44, 61], are highly effective at learning complex high-dimensional distributions and have been put to good use for image synthesis and unsupervised learning. However, GANs are notoriously hard to train, which we noticed empirically in preliminary experiments. We therefore chose to rely on VAEs. We provide the basics of VAE below and will use the same formalism in the next section to describe how we use it for our purposes.

Let $\mathcal{M} = \{\mathbf{M}^{(1)}, \ldots, \mathbf{M}^{(M)}\}$ be a set of observations $\mathbf{M}^{(i)}$ which are distributed according to the generative distribution $p(\mathbf{M}^{(i)}, \mathbf{z}^{(i)}; \boldsymbol{\theta}_d) = p(\mathbf{M}^{(i)}|\mathbf{z}^{(i)}; \boldsymbol{\theta}_d) \cdot p(\mathbf{z}^{(i)}; \boldsymbol{\theta}_d)$, where $\mathbf{z}^{(i)}$ is a vector of latent (hidden) variables, and $\boldsymbol{\theta}_d$ are the parameters of the distribution. In theory, these parameters can be learned by maximizing the log-likelihood of the observed data

$$\log p(\mathbf{M}^{1:M}; \boldsymbol{\theta}_d) = \sum_{i=1}^{M} \log p(\mathbf{M}^{(i)}; \boldsymbol{\theta}_d) \ . \tag{6.1}$$

As discussed in Chapter 2, computing the actual log-likelihood is intractable for non-trivial models, and instead, we can resort to the following variational approximation:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{z}|\mathbf{M};\boldsymbol{\theta}_e)}[\log p(\mathbf{M}, \mathbf{z}; \boldsymbol{\theta}_d) - \log q(\mathbf{z}|\mathbf{M}; \boldsymbol{\theta}_e)] \ , \tag{6.2}$$

where we dropped the indices (i) for clarity and $\mathbb{E}_q[\cdot]$ denotes expectation with respect to the variational distribution q defined over hidden variables $\mathbf{z}$ and parameterized by $\boldsymbol{\theta}_e$. We can further rewrite Eq. 6.2 as

$$\mathcal{L} = \mathbb{E}_q[\log p(\mathbf{M}|\mathbf{z}; \boldsymbol{\theta}_d)] - \mathbb{E}_q[\log \frac{q(\mathbf{z}|\mathbf{M}; \boldsymbol{\theta}_e)}{p(\mathbf{z}; \boldsymbol{\theta}_d)}] \ , \tag{6.3}$$

where the left-hand term can be understood as a negative reconstruction error of the generative model (decoder) $p(\mathbf{M}|\mathbf{z})$ and the right-hand term is the KL divergence between the approximate posterior (encoder) $q(\mathbf{z}|\mathbf{M})$ and the prior $p(\mathbf{z})$, which acts as a reguralizer. Without this term, there would be no incentive to learn a smooth and meaningful representation for $\mathbf{z}$, which is crucial if we want to then traverse this space when doing model fitting. In the context of deep generative models, both the generative model $p(\mathbf{M}|\mathbf{z})$ and the approximate posterior $q(\mathbf{z}|\mathbf{M})$ are parameterized using deep neural networks. Distribution q is usually taken to be a diagonal Gaussian, but more sophisticated distributions have been investigated in [82, 127, 154].

6.2 Method

In this section, we first describe the mesh parameterization that enables us to efficiently apply CNNs to the face geometry. We then discuss an important insight behind our method, which is that both global and local linear models that are central to most state-of-the-art approaches to modeling 3D faces can be expressed as shallow auto-encoders. A natural way to increase their flexibility would therefore be to simply replace the linear encoders and decoders by non-linear ones. However, in practice, this would not be enough because model fitting requires a well-behaved parameter space that is well suited for optimization. We therefore show that the convolutional VAEs can be used for this purpose in the global case. Finally, since this results in a model that is more flexible than the original ones but still suffers from the limitations of all global ones, we introduce a compositional version of VAEs, which combines the strength of local and global models by explicitly representing various deformation levels.

6.2.1 Mesh Representation

Typically, face geometry is represented as a triangular mesh, or, more formally, as a pair $(\mathcal{V}, \mathcal{T})$, where $\mathcal{V} \in \mathbb{R}^{N\times 3}$ is a collection of 3D vertices and $\mathcal{T}$ is a set of triangles that defines the topology. Note that, we keep the same triangulation for all the faces and assume the shape variations are all captured by the $\mathcal{V}$ coordinates. Details on how to perform mesh registration are given in Section 6.4.1. Further, these coordinates are represented as a 3-channel image $\mathbf{M} \in \mathbb{R}^{H\times W\times 3}$ and the triangles in $\mathcal{T}$ by triplets of the vertex indices of the form $\{(i,j),(i+1,j),(i+1,j+1)\}$ and $\{(i,j),(i,j+1),(i+1,j+1)\}$, as shown in Figure 6.1. Importantly, this means that pixels that are neighbors in terms of pixel coordinates are also topological neighbors. This makes it natural to perform 2D convolutions on meshes and efficiently use the deep learning machinery.

Figure 6.1: Example of (mean-subtracted) UV parameterization of a face. From left-to-right: x, y, z coordinates.

6.2.2 Linear Face Models as Autoencoders

A global linear model such as the one of [16] represents all possible face shapes as linear combinations in a set of basis vectors. In [16], it was obtained by performing principal component analysis on a training database.

Formally, we can write

$$\boldsymbol{h} = \mathbf{W}_e \cdot \mathbf{M}\ ,\ \hat{\mathbf{M}} = \mathbf{W}_d \cdot \boldsymbol{h}\ , \tag{6.4}$$

where $\mathbf{W}_e \in \mathbb{R}^{k\times 3N}$, $\mathbf{W}_d \in \mathbb{R}^{3N\times k}$ are respectively encoding and decoding matrices, and $\boldsymbol{h} \in \mathbb{R}^k$ is a set of k linear coefficients, such that $\|\mathbf{M} - \hat{\mathbf{M}}(\boldsymbol{h})\|$ is mimimized in the space spanned my $\mathbf{W}_e$. The transformations of Eq. 6.4 can be implemented by a shallow linear auto-encoder, as shown in Figure 6.2 (a). Given the observations such as depth maps or the 2D positions of sparse landmarks, which we will denote $\mathbf{X}$, fitting a model to it can then be expressed as finding a set of parameters $\hat{\boldsymbol{h}}$ that maximizes the data likelihood

$p(\mathbf{X}|\mathbf{W}_d \cdot \boldsymbol{h})$.

Local linear models such as [152] give more flexibility than global ones by decoupling the parameters between different parts of the mesh. In practice, this means that $\boldsymbol{h}$ is factored into independent sets of parameters $\boldsymbol{h}_\rho$ for distinct patches $\mathbf{M}_\rho$ of the mesh. Assuming that all these parameters are expressed in the same bases $\boldsymbol{\theta}_e, \boldsymbol{\theta}_d$, these local models can be seen as shallow *convolutional* auto-encoders, whose space of potential deformations is captured by a convolutional feature map $\boldsymbol{h}$, as shown in Figure 6.2 (b). Bases $\boldsymbol{\theta}_e$ and $\boldsymbol{\theta}_d$ are then the parameters of the convolutional layers of respectively encoder and decoder, which are shared among all the patches.

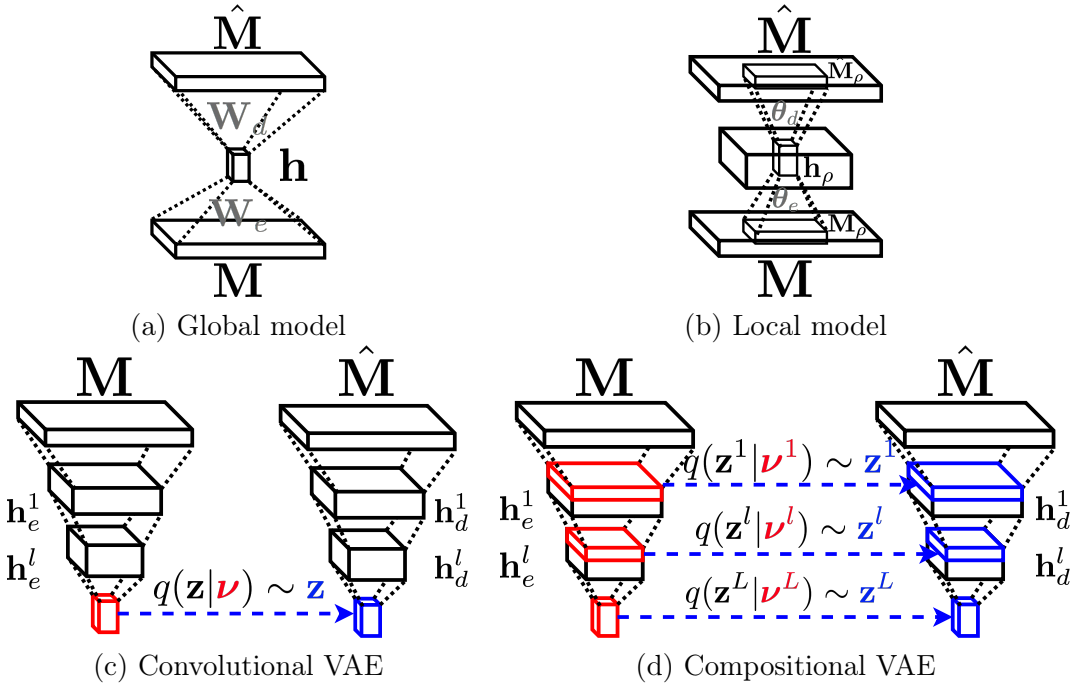


Figure 6.2: Autoencoding architectures for face geometry.

6.2.3 Convolutional Mesh VAE

Given that linear models can be viewed as linear auto-encoders, a natural way to extend them and potentially solve the problems discussed in Section 6.1, is to use non-linear versions of the encoders and decoders.

For global models, we therefore write

$$\boldsymbol{h} = E(\mathbf{M}; \boldsymbol{\theta}_e) \ , \ \hat{\mathbf{M}} = D(\boldsymbol{h}; \boldsymbol{\theta}_d) \ , \tag{6.5}$$

where $E(\cdot; \boldsymbol{\theta}_e)$ and $D(\cdot; \boldsymbol{\theta}_d)$ are multi-layer convolutional encoders and decoders, parame-

terized by weights $\boldsymbol{\theta}_e$ and $\boldsymbol{\theta}_d$ respectively, similarly to architectures in Figure 6.2 (c)-(d). In a similar manner as for the linear case, we can estimate $\boldsymbol{\theta}_e$ and $\boldsymbol{\theta}_d$ from the training data and then do model fitting by finding the parameter vector $\hat{\boldsymbol{h}}$ that maximizes $p(\mathbf{X}|D(\boldsymbol{h};\boldsymbol{\theta}_d))$.

The non-linear parameterization of Eq. 6.5 is more flexible than the one of Eq. 6.4. Unfortunately, it does not guarantee anymore that even small differences in the value of $\boldsymbol{h}$ from the values observed during training will not result in estimated shapes $\hat{\mathbf{M}} = D(\boldsymbol{h};\boldsymbol{\theta}_d)$ which are not representative of the true posterior, or, in other words, which are *not* face-like. To remedy this, we replace the simple auto-encoder of Eq. 6.5 by a *variational* auto-encoder based on the formalism described in Section 6.1.3, which ensures the smoothness of the learned space by enforcing a prior on the posterior q.

Namely, we parameterize the distribution over latent variables $q(\mathbf{z}|\mathbf{M};\boldsymbol{\theta}_e)$ and the generative model $p(\mathbf{M}|\mathbf{z};\boldsymbol{\theta}_d)$ in terms of a deep net encoder $E(\cdot)$ and decoder $D(\cdot)$ respectively. This yields a variational reformulation of Eq. 6.5:

$$\boldsymbol{\nu} = E(\mathbf{M};\boldsymbol{\theta}_e)\ ,\ \mathbf{z} \sim q(\mathbf{z}|\boldsymbol{\nu})\ ,\ \hat{\mathbf{M}} = D(\mathbf{z};\boldsymbol{\theta}_d)\ , \tag{6.6}$$

where $\boldsymbol{\nu}$ are the parameters of the approximate posterior, which is assumed to be a diagonal Gaussian. In practice, evaluating $\hat{\mathbf{M}}$ now requires sampling from $q(\mathbf{z}|\boldsymbol{\nu})$, which is not a differentiable operation. This was addressed in [83] by representing $\mathbf{z}$ as a deterministic variable that depends on $\boldsymbol{\nu}$ and auxiliary noise, which makes it possible to minimize the lower bound $\mathcal{L}$ of Eq. 6.2 and Eq. 6.3 using stochastic gradient descent. In Section 2.4, we discuss this solution and related background in more detail.

6.2.4 Compositional Mesh VAE

The non-linear parameterization of Eq. 6.6 is more flexible than the linear one of Eq. 6.4 while still providing a latent space that is smooth and easy to optimize over. However, both formulations still depend on a single low-dimensional vector, namely $\boldsymbol{h}$ in Eq. 6.4 and $\mathbf{z} \sim q(\cdot|\boldsymbol{\nu})$ in Eq. 6.6, to represent the shape, which makes it difficult to capture high-frequency deformations.

In this section, we propose a solution to this difficulty by introducing *multiple* layers of hidden variables $\mathbf{z}^{1:L}$, where each individual layer models a separate level of detail. Intuitively, the goal of the encoder is then to gradually *decompose* the input mesh $\mathbf{M}$ into those variables, such that the decoder can then *compose* those individual representations back into a final reconstruction $\hat{\mathbf{M}}$. The higher-level layers, that is, those corresponding to lower l-s, have more degrees of freedom and more local control with smaller receptive field, are therefore well suited to represent the high-frequency geometric components,

whereas the lower-level layers have more control over the global shape. This will be demonstrated at the end of the evaluation section.

Formally, the joint distribution for the observed meshes $\mathbf{M}$ and latent variables $\mathbf{z}^{1:L}$ can now be written as

$$p(\mathbf{M}, \mathbf{z}^{1:L}) = p(\mathbf{M}|\hat{\mathbf{M}}(\mathbf{z}^{1:L})) \cdot \prod_{l=1}^{L} p(\mathbf{z}^l|\boldsymbol{\xi}^l) \ , \tag{6.7}$$

where $\boldsymbol{\xi}^l$ are the parameters of the prior, and the approximate posterior q is factorized over layers l as

$$q(\mathbf{z}^{1:L}|\mathbf{M}; \boldsymbol{\theta}_e) = \prod_{l=1}^{L} q(\mathbf{z}^l|\boldsymbol{\nu}^l) \ . \tag{6.8}$$

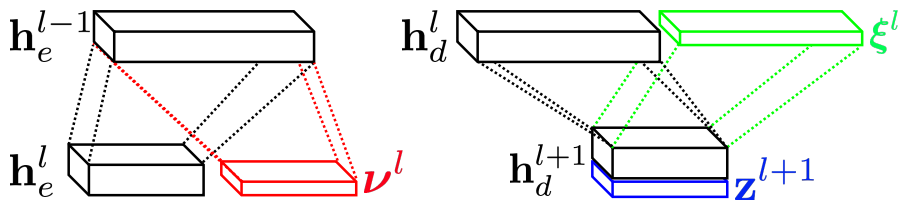


Figure 6.3: Compositional VAE layers. Encoder (left): given activations $\boldsymbol{h}_e^{l-1}$ we output the lower-dimensional activations $\boldsymbol{h}_e^l$ along with the *posterior* parameters $\boldsymbol{\nu}^l$. Decoder (right): given activations $\boldsymbol{h}_d^{l+1}$ and a sample $\mathbf{z}^{l+1}$ we output the higher-dimensional activation $\boldsymbol{h}_d^l$ along with the *prior* parameters $\boldsymbol{\xi}^l$.

To account for the new factorized structure of our latent space, we expand the formulation of Eq. 6.6 and write

$$\begin{aligned} \boldsymbol{h}_e^l, \boldsymbol{\nu}^l &= E^l(\boldsymbol{h}_e^{l-1}; \boldsymbol{\theta}_e^l) \ , \\ \mathbf{z}^l &\sim q(\mathbf{z}^l|\boldsymbol{\nu}^l) \ , \\ \boldsymbol{h}_d^l, \boldsymbol{\xi}^l &= D^l(\boldsymbol{h}_d^{l+1}, \mathbf{z}^{l+1}; \boldsymbol{\theta}_d^l) \ , \end{aligned} \tag{6.9}$$

where $\boldsymbol{\nu}^l \in \mathbb{R}^{H^l \times W^l \times C^l}$ and $\boldsymbol{\xi}^l \in \mathbb{R}^{H^l \times W^l \times C^l}$ are parameters of the approximate posterior $q(\mathbf{z}^l|\boldsymbol{\nu}^l)$ and prior $p(\mathbf{z}^l|\boldsymbol{\xi}^l)$, respectively, which we take to be diagonal Gaussians as in the original VAE [83]. During training, the KL term of Eq. 6.3 ensures that $q(\mathbf{z}^l|\boldsymbol{\nu}^l)$ stays close to the prior $p(\mathbf{z}^l|\boldsymbol{\xi}^l)$, which encourages the model to learn a more well-behaved representation for $\mathbf{z}^l$. Note that, for $\mathbf{z}^L$, we do not have to predict $\boldsymbol{h}_e^L$, and the corresponding prior is set to zero-mean unit-variance $\boldsymbol{\xi}^L = (\mathbf{0}, \mathbf{I})$. Figure 6.3 shows a graphical illustration of Eq. 6.9, and Figure 6.2 (d) depicts the whole architecture. Note that, the overall architecture is quite similar to the U-Net [133], which is widely

used for semantic segmentation, with an important difference that in our model the skip connections are probabilistic.p

Finally, substituting Eqs. 6.7 and 6.8 into the lower bound of Eq. 6.3 gives us the training objective that we can optimize given a training set using SGD. We give additional details on this procedure in Section 6.4.2.

6.3 Model Fitting

The compositional VAE model described above is designed to effectively encode the facial deformations in different layers of its hidden variables. An important property that is a result of the factorized structure and the variational nature of the model is its ability to extrapolate, which is especially useful for face model fitting given 3D or 2D constraints. In what follows, we describe the model fitting procedure in different application scenarios, ranging from depth map-based face fitting to shading-based face reconstruction from just a single image.

Namely, given generic image data $\mathbf{X}$ and the pre-trained decoder D, our goal is to find parameter vectors $\mathbf{z}^{1:L}$ such that decoded mesh whose shape is given by $\hat{\mathbf{M}} = D(\mathbf{z}^{1:L})$ fits the data as well as possible. Formally, this is equivalent to solving a MAP problem, that is, maximizing

$$\log p(\mathbf{X}|\hat{\mathbf{M}}(\mathbf{z}^{1:L})) + \sum_{l=1}^{L} \log p(\mathbf{z}^l|\boldsymbol{\xi}^l(\mathbf{z}^{l+1})) \;, \tag{6.10}$$

wrt $\mathbf{z}^{1:L}$, where $p(\mathbf{X}|\hat{\mathbf{M}})$ is the probability of observing $\mathbf{X}$ if the mesh shape is given by $\hat{\mathbf{M}}$. Note that the prior probability terms act as regularizers that prevent the model parameters from straying too far away from values observed in the training data. While this may be advantageous in the presence of noise, it also limits the ability of the model to extrapolate. In the results section, we will therefore compare results with different combinations of these terms across various types of constraints and noise levels. In practice, we use gradient descent to iteratively optimize Eq. 6.10. Below, we describe the formulation of the data term $p(\mathbf{X}|\hat{\mathbf{M}})$ for different types of input data.

3D to 3D correspondences. The simplest case is when we know the position $\mathbf{M}_i$ of a subset $\mathcal{I}$ of vertices up to some precision, for example obtained from a multi-view setup. Assuming a Gaussian error distribution with unit variance and conditional independence of individual observations, we write

$$\sum_{i\in\mathcal{I}} \log p(\mathbf{M}_i|\hat{\mathbf{M}}_i) \propto -\sum_{i\in\mathcal{I}} ||\mathbf{M}_i - \hat{\mathbf{M}}_i||_2^2 \;. \tag{6.11}$$

2D to 3D correspondences. In realistic scenarios, 3D to 3D correspondences are rarely available but 2D to 3D ones can be established by matching sparse facial landmarks in an image. Therefore, let $\mathcal{I}$ now be the set of vertices $\mathbf{M}_i$ for which we have 2D projections $\mathbf{P}_i \in \mathbb{R}^2$. Given camera intrinsic $\mathbf{K} \in \mathbb{R}^{3\times3}$ and extrinsic $\mathbf{R}|\mathbf{t} \in \mathbb{R}^{3\times4}$ parameters, and making the same Gaussian IID assumptions about the observations, we can write:

$$\sum_{i\in\mathcal{I}} \log p(\mathbf{P}_i|\hat{\mathbf{M}}_i) \propto -\sum_{i\in\mathcal{I}} ||\mathbf{P}_i - \Pi_{\mathbf{K},\mathbf{R}|\mathbf{t}}\hat{\mathbf{M}}_i||_2^2 \,, \tag{6.12}$$

where $\Pi_{\mathbf{K},\mathbf{R}|\mathbf{t}}\hat{\mathbf{M}}_i$ are the 2D projections of the model vertices.

Depth maps. Depth cameras have now become an inexpensive and widely available means for face capture. Furthermore, high-quality depth maps can be obtained by stereo matching of high-resolution RGB images. Let $\mathbf{D} \in \mathbb{R}^{H_D\times W_D}$ be such a depth map. We now need to define $p(\mathbf{D}|\hat{\mathbf{M}})$. Ignoring differentiability for a moment, we consider the set of vertices visible from the depth camera point of view $\mathcal{I}_V \subset H \times W$, compute their image coordinates $(\hat{u}_i, \hat{v}_i)$ in the depth map coordinate frame defined by $\mathbf{K}, \mathbf{R}|\mathbf{t}$. Then, we evaluate the difference between the depth value stored at those coordinates $\mathbf{D}_i = \mathbf{D}_i(\hat{u}_i, \hat{v}_i)$ and the one that projected from the 3D vertex position using camera extrinsics $\hat{\mathbf{D}}_i = (\mathbf{R}\cdot\hat{\mathbf{M}}_i + \mathbf{t})_z$. Under the same Gaussian assumptions as before, this allows us to write

$$\sum_{i\in\mathcal{I}_V} \log p(\mathbf{D}_i|\hat{\mathbf{M}}_i) \propto -\sum_{i\in\mathcal{I}_V} ||\mathbf{D}_i - \hat{\mathbf{D}}_i||_2^2 \,. \tag{6.13}$$

Unfortunately, self-occlusions make visibility non-differentiable. To overcome this difficulty, we compute $\mathcal{I}_V$ by rendering the mask of visible vertex indices using OpenGL during forward passes and keep $\mathcal{I}_V$ fixed during the backward passes. Furthermore, in order for us to be able to propagate gradients not only through the values of depth, but also through the image coordinates $(\hat{u}, \hat{v})$, we employ a bilinear kernel

$$\mathbf{D}_i = \sum_{u,v} \mathbf{D}(u,v)\max(0, 1-|u-\hat{u}|)\max(0, 1-|v-\hat{v}|) \,, \tag{6.14}$$

to perform the differentiable sampling, as in [70].

Shape from Shading Constraints. Another compelling but very challenging application is to fit face model to a single RGB image. Whereas the rough expression can be estimated using sparse 2D-3D correspondences, they are not sufficient to capture identity-specific high-frequency detail. One approach to overcome this is using image formation models. Let $\mathbf{I} \in \mathbb{R}^{H_I\times W_I\times3}$ be an RGB image. Our goal is now to define $p(\mathbf{I}|\hat{\mathbf{M}})$. We assume a simple Lambertian model, with a single 3-channel light source

parameterized by $\mathbf{L} \in \mathbb{R}^{3\times3}$. Further, we use the mesh $\hat{\mathbf{M}}$ to compute vertex normals $\hat{\mathbf{N}}$, which amounts to computing a cross product between two sets of vectors. Then, the model intensity can be computed as $\hat{\mathbf{I}}_i = \mathbf{T}_i \cdot \mathbf{L} \cdot \hat{\mathbf{N}}_i$, given the texture $\mathbf{T}_i$. Computing the texture is a highly non-trivial task, and here we simply set it to be uniform white, assuming that to some extent the albedo can be captured by $\mathbf{L}$. We now can write

$$\sum_{i\in\mathcal{I}_V} \log p(\mathbf{I}_i|\hat{\mathbf{M}}_i) \propto -\sum_{i\in\mathcal{I}_V} ||\mathbf{I}_i - \hat{\mathbf{I}}_i||_2^2 \tag{6.15}$$

where we used same approach for sampling and computing $\mathcal{I}_V$ as for the depth maps. Moreover, we also use a similar trick for computing $\mathbf{L}$: at every forward pass, we use the current estimate of $\hat{\mathbf{N}}$ to solve Eq. 6.15 for $\mathbf{L}$, and then keep it fixed during the backward pass.

6.4 Evaluation

We start with a description of our face geometry dataset and give some implementation details. We then present quantitative results on several benchmarks and demonstrate qualitatively that our model can be used both to fit noisy depth maps and to perform shape-from-shading. Finally, we present experiments designed to explore the learned latent space and showcase its decompositional power.

6.4.1 Dataset

A face geometry dataset aligned with a reference topology is required to train and evaluate our model. However, none of the publicly available face shape datasets [19, 21] offer truly high-resolution models, which would not allow us to fully test descriptive power of our compositional model. We thus built a new one that comprises high-quality face geometries using a multi-view camera setup similar to [11] and performing stereo-based 3D face reconstruction. We captured 20 different people, each performing a set of expressions similar to those of blendshapes of [75]. This resulted in 2140 high-quality meshes. To create a uniform face topology, we first defined a generic neutral face template mesh with a precomputed UV map. This generic mesh was then aligned to the mesh for each subject with their expression being neutral. To this end, we performed non-rigid mesh deformation [146] with facial landmark constraints, which were detected on the corresponding RGB images from the multi-view setup [161].

Given those topologically aligned neutral meshes for each individual, we further aligned them to identity-specific peak expression scans using facial landmarks, geometrical constraints, and optical flow-based constraints. This produced fully-aligned meshes, which are all registered to the same topology represented as a UV map of size $H \times W = 256 \times 256$.

Finally, we removed from all mesh coordinates the global rotation and translation of the head. Figure 6.4 depicts some of the fully-registered meshes.

Figure 6.4: Samples from the dataset.

In all of our numerical experiments, we use a total of 1712 meshes of 16 randomly chosen subjects for training, and 428 meshes of the remaining 4 subjects for testing.

6.4.2 Implementation Details

All the models are trained using stochastic gradient descent with ADAM [81] optimizer with step size 1×10^{-4} and the hyperparameters $\beta_1 = 0.9, \beta_2 = 0.999$. For the convolutional models, we use identical architecture with 5 residual blocks, with down(up)-sampling after each block of the encoder(decoder). Each block consists of two 4x4 convolutional layers with ELU non-linearities, with weights initialized from a zero-mean Gaussian distribution with standard deviation 0.001. The final 8×8 convolutional representation is mapped to the bottleneck representation using a fully connected layer. Both linear and convolutional VAE models use 128-dimensional bottleneck. For the compositional VAE, we use 64-dimensional bottleneck $\mathbf{z}^6$, all the remaining convolutional maps $\mathbf{z}^5, \ldots, \mathbf{z}^1$-s have 16 channels and the size of the corresponding activation layers. When training variational models, we employ the free-bits technique of [82] with $\lambda = 4$, as we found that it leads to better convergence.

Given a pre-trained model, the model fitting is done by optimizing Eq. 6.10 with one of the data terms from Section 6.3 using gradient descent with ADAM optimizer. For noisy depth maps from Section 6.4.4, we found that using a more robust L1 loss leads to better results. For the 2D-3D and 3D-3D fitting results presented in the following section, the optimization takes around 3-4s per image, and for depth fitting it takes around 8s, on a single NVidia P100 GPU.

6.4.3 Quantitative Evaluation

In this section, we evaluate quantitatively the behavior of our model and compare it to that of baselines on synthetically generated 3D to 3D correspondences, 2D to 3D correspondences, and depth maps. In all three cases, we perform the fitting as described in Section 6.3 and will demonstrate in the following section that our approach works equally well on real stereo and shape-from-shading data.

Our baselines include the traditional linear model, introduced in Section 6.2.2, as well as a the deep convolutional VAE from Section 6.2.3. We will refer to them as `VAE` and `LINEAR` in the result tables below.

We also compare multiple variants of our approach depending on how we handle the prior terms $\log p(\mathbf{z}^l|\boldsymbol{\xi}^l(\mathbf{z}^{l+1}))$ of Eq. 6.10. We denote them as $\mathbf{z}^l$ for simplicity and can either use them or ignore them. More specifically, we report our results that range from using only $\mathbf{z}^1$ (less priors) to $\mathbf{z}^{1:4}$ (more priors). Recall from Section 6.2.4 that the lower values of l denote layers that influence most the overall shape and the higher values the fine details. This means that we progressively make constraints more and more global.

Method	0.2%	0.5%	2%	10%
`LINEAR`	2.795	1.309	1.016	0.980
`VAE`	1.678	1.317	1.176	1.139
`OURS` $\mathbf{z}^1$	1.470	1.079	**0.596**	**0.247**
`OURS` $\mathbf{z}^{1:2}$	1.468	1.121	0.609	0.336
`OURS` $\mathbf{z}^{1:3}$	1.396	1.020	0.616	0.467
`OURS` $\mathbf{z}^{1:4}$	**1.320**	**0.986**	0.775	0.717

Table 6.1: Results for model fitting with 3D-3D correspondences. RMSE in mm for different proportions of constrained vertices.

3D to 3D correspondences. In Table 6.1, we report the average RMSE in mm when constraining the 3D position of a subset of mesh vertices, as a function of the proportion of vertices being fixed. While these are chosen randomly for each subsampling level, the error is measured for *all* mesh vertices. All variants of our full compositional model outperform `LINEAR` and `VAE`, even when constraining as few as 0.2% of the vertices, which amounts to about 60 3D to 3D correspondences. This suggests that the performance boost is not only attributable to the increased flexibility of our representation but also to the fact it captures the right priors about face geometry. Unsurprisingly, the fewer correspondences we have, the more important the global shape constraints become, as evidenced by the fact that we get the best results when using the priors for all the layers in the 0.2% case but only the ones on the fine details in the 2% and 10% cases.

Method	0.2%	0.5%	2%	10%
LINEAR	4.381	3.691	3.394	3.302
VAE	3.606	3.183	3.114	3.077
OURS $\mathbf{z}^1$	2.690	2.521	**2.390**	**2.330**
OURS $\mathbf{z}^{1:2}$	2.660	2.521	2.396	2.343
OURS $\mathbf{z}^{1:3}$	2.606	**2.512**	2.431	2.396
OURS $\mathbf{z}^{1:4}$	**2.586**	2.545	2.472	2.453

Table 6.2: Results for model fitting with 2D-3D correspondences. RMSE in mm for different proportions of constrained vertices.

2D to 3D correspondences. In Table 6.2, we present fitting results obtained by constraining some mesh vertices to project at the right location in one of the camera views. As before, we report results obtained by constraining in this fashion from 0.2% to 10% of the vertices. Due to 2D-3D ambiguities, this is a more difficult that exploiting 3D to 3D correspondences and the accuracies for all methods are worse than those reported in Table 6.2. Nevertheless all variants of our approach still outperform the baselines and we observe again that, the sparser the data is, the more important it is to account for the priors at all four levels of our architecture.

Method	$\sigma^2 = 1$	$\sigma^2 = 2$	$\sigma^2 = 3$
LINEAR	3.908	3.924	3.953
VAE	3.167	3.199	3.249
OURS $\mathbf{z}^1$	3.032	3.142	3.252
OURS $\mathbf{z}^{1:2}$	**3.020**	**3.114**	3.215
OURS $\mathbf{z}^{1:3}$	3.079	3.127	**3.191**
OURS $\mathbf{z}^{1:4}$	3.110	3.150	3.226

Table 6.3: Results for model fitting with depth data. RMSE in mm for different noise levels.

Depth maps. We generate synthetic depth maps from the ground truth data and corrupt them by adding different levels of IID Gaussian noise. We report our results in Table 6.3. Since the correspondences must be established and computing visibility is a non-differentiable operation as discussed in Section 6.3, fitting is more difficult than before. As a result, our method still outperforms the baselines but by a smaller margin. In this case, the best variants of our model are those that enforce priors up to $\mathbf{z}^3$. In other words, in the presence of noisy but dense data, over-constraining the model can be less beneficial.

6.4.4 Qualitative Results

We now turn to more realistic image data to demonstrate the power of our model. To this end, we first captured two additional subjects using a small 3-camera setup, and

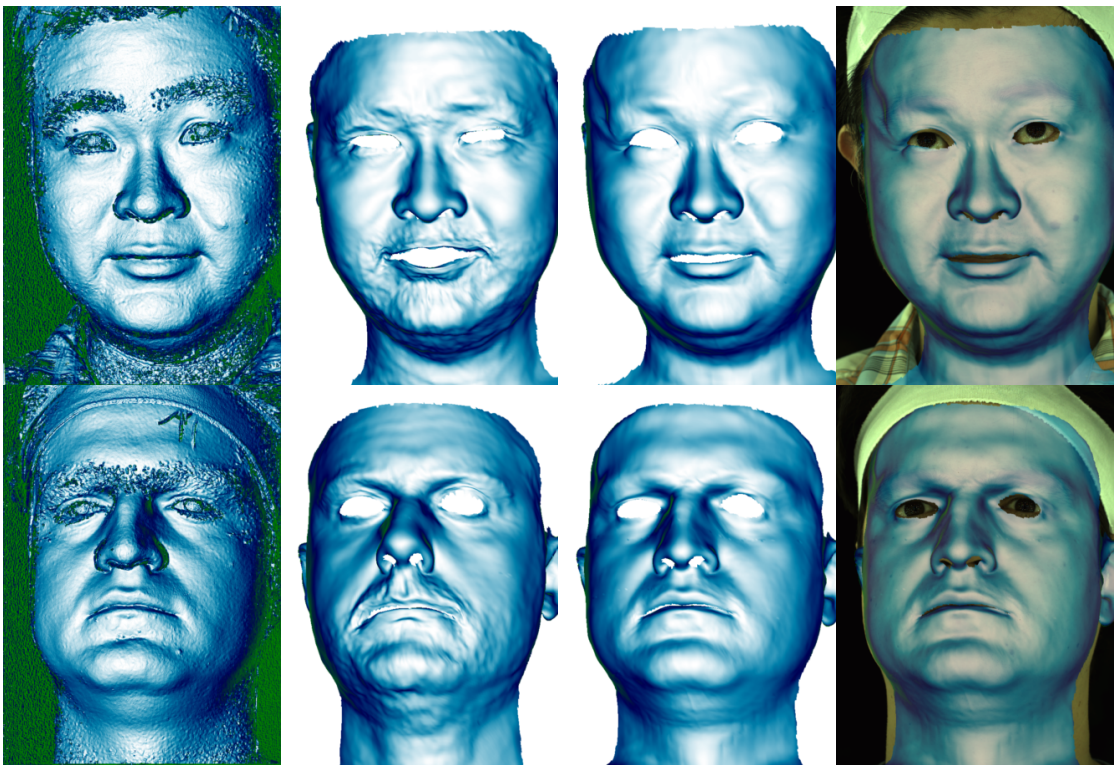


Figure 6.5: Visual results for fitting noisy depth maps. From left-to-right: input depth map, rendered mesh (`LINEAR`), rendered mesh (`OURS`), rendered mesh (`OURS`) overlaid with the image.

used stereo to compute noisy depth maps that are representative of what can expect in a real world environment. Figure 6.5 depicts our results alongside those of LINEAR. Our method correctly captures not only the overall head shape but also fine details whereas LINEAR introduces numerous artifacts instead.

Figure 6.6: Visual results for shape-from-shading for images from [135]. From left-to-right: rendered mesh (LINEAR) rendered mesh (OURS).

In Figure 6.6, we demonstrate the ability of our model to capture an unusual expression—that of the woman of the top—or face—that of the man at the bottom—using images from 300-W dataset [135]. We initialize the process by using the 2D landmarks provided by [135], to compute the head pose and general expression, and then solve the MAP of Eq. 6.10 with the data term of Eq. 6.15. For comparison purposes, we also used LINEAR, which again produced unwanted artifacts.

6.4.5 Exploring the Latent Space

We start with an experiment that demonstrates the spatial extent the changes in a single hidden variable at different levels have on the output. For that, we first fix all the variables $\mathbf{z}^{1:L}$ to the values corresponding to the mean face, and then vary a single location in $\mathbf{z}^{l}$ from the minimum to the maximum value for that variable across the dataset. The results of those variations are shown in Figure 6.7. Naturally, the variables from the

Figure 6.7: Visualizing the receptive field: how changing the value of a single variable affects the output. Heatmaps represent the MSE between the deformed mesh and the original in the UV space. From left-to-right: $\mathbf{z}^6$ - $\mathbf{z}^2$.

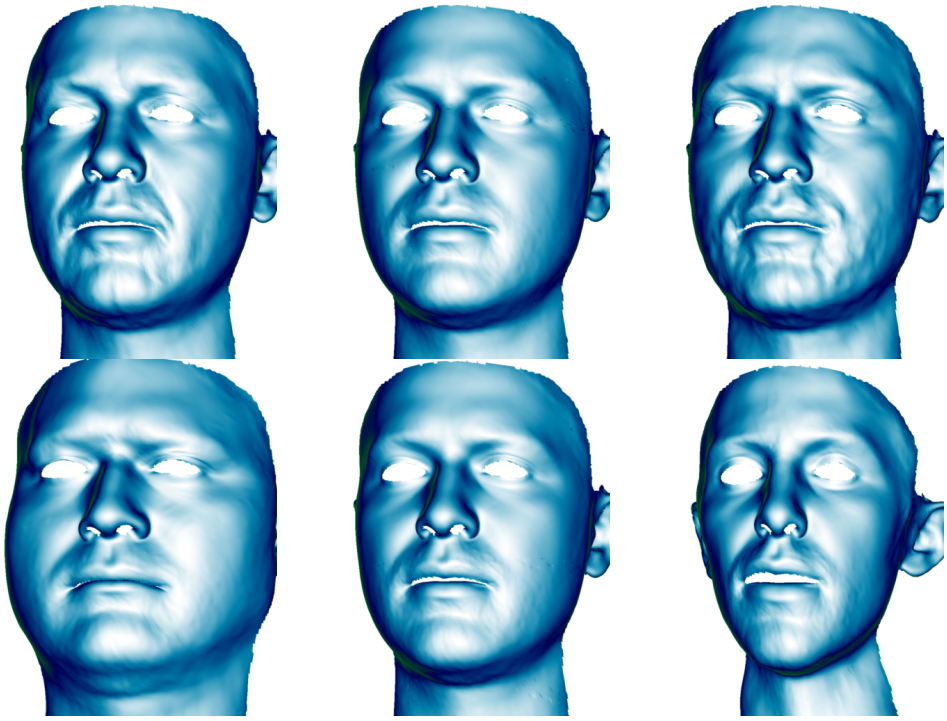


Figure 6.8: Visualizing the effect of varying the first PCA component of $\mathbf{z}^{1:2}$ (top) and $\mathbf{z}^{4:5}$ (bottom) representations.

layers which are closer to the bottleneck have global influence on the mesh, and as we go closer to the output, their effective receptive field gradually shrinks.

Further, we explore the learned space by looking at the kind of details that different subsets of variables $\mathbf{z}^{1:L}$ are capturing. PCA is a classical approach for this kind of exploratory analysis. Namely, we first compute the projections $\hat{\mathbf{z}}^{1:L}$ for all the meshes in the dataset by optimizing the posterior of Eq. 6.10, and then compute the PCA basis via SVD for a subset of variables of interest. We report visual results of varying first principal components of $\mathbf{z}^{1,2}$ and $\mathbf{z}^{5,6}$ in Figure 6.8. As can be seen from this illustration, the higher layers $\mathbf{z}^{1,2}$, which have smaller receptive field size and more degrees of freedom, capture high-frequency deformations, such as beards and wrinkles. On the other hand, the lower layers $\mathbf{z}^{5,6}$ evidently capture global details, such as the general shape of the head.

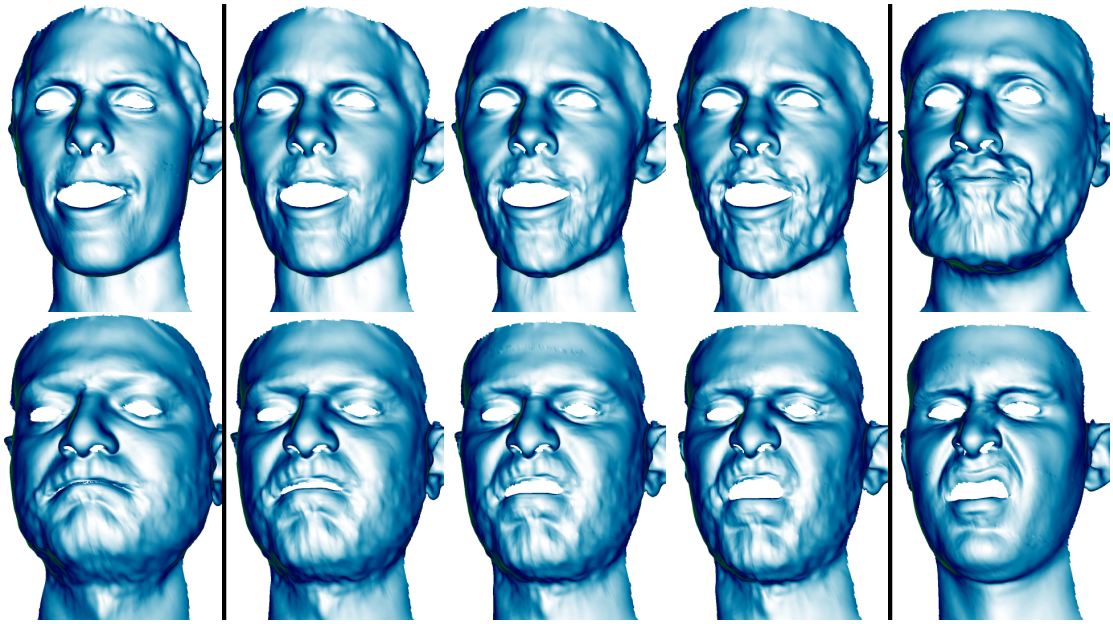


Figure 6.9: Visual results for detail transfer. The leftmost and rightmost columns are the two original meshes. Top: interpolating $\mathbf{z}^{1,2}$ while keeping $\mathbf{z}^{5,6}$ details fixed. Bottom: interpolating $\mathbf{z}^{5,6}$ while keeping $\mathbf{z}^{1,2}$ fixed.

An alternative way to explore the latent space, which is usually employed in deep generative model literature, is to directly traverse the space between the projections of the data samples. To do that we select several random pairs of meshes and find the corresponding values of $\hat{\mathbf{z}}^{1:L}$ by optimizing Eq. 6.10. Given those, we then can interpolate the values of a certain subset of variables between two projections, while keeping all the others fixed. The visual demonstration of this detail transfer process for $\mathbf{z}^{1,2}$ and $\mathbf{z}^{5,6}$ is shown in Figure 6.9. We see that higher layers $\mathbf{z}^{1,2}$ are capturing higher-frequency details, e.g. beards and small variations in eyelids and lips, whereas the lower layers $\mathbf{z}^{5,6}$ are

capturing the overall shape of the head and the general expression. This indicates that the model indeed separates the geometrical details into different semantically meaningful layers of representation. We provide additional detail transfer results in Appendix D.

6.5 Conclusion

We proposed a novel data-driven parameterization for face geometry, and demonstrated its versatility on a variety of model fitting tasks. An exciting direction for future work is investigating alternative architectures for the decoders, such as PixelRNN, and learning to predict hidden representations directly from the images, without a need for optimization. We believe that applying modern generative modeling techniques to geometry data is a very promising field, especially since, unlike for natural images, there exist more straightforward ways to evaluate the quality of the latent space.

7 Concluding Remarks

7.1 Summary

In this thesis, we develop multiple methods for a diverse range of computer vision tasks focused on human modeling and scene understanding. All of the proposed methods benefit from a powerful class of approximate inference methods - variational inference.

In Chapter 3 we introduce DPOM, a probabilistic approach for detecting humans in depth images. Majority of existing methods for depth-based detection rely on heuristics and rarely perform explicit occlusion reasoning [71, 108, 147], and thus are prone to failures which are hard to explain and predict. This motivates our approach, which is based on a principled generative model for depth images, which enables joint reasoning and explicitly takes into account possible occlusions. This makes DPOM particularly useful in crowded scenes when occlusions are common. Since the exact inference in the model is computationally intractable, we develop an efficient approximate algorithm based on variational methods. In our experiments, we demonstrate that our approach performs better than the state-of-the-art on multiple challenging datasets.

In Chapter 4 we propose a novel algorithm for mean-field variational inference, which supports a large class of discrete probabilistic models, such as POM [52], DPOM and, more generally, CRF models with repulsive and higher-order terms. The main motivation for our approach comes from the fact that existing algorithms for mean-field variational inference are either too inefficient for realistic vision applications, or suffer from non-convergence for many useful models. Our method is based on a special variant of proximal gradient descent, and, unlike existing methods, it is very efficient and provably convergent for a large class of models. We show experimentally that using our algorithm yields improved performance on multiple tasks, such as human detection and semantic segmentation.

In Chapter 5 we develop a unified framework for multi-human scene understanding. Most

of the recent methods that tackle the problem [31, 69, 123] do not aim at solving the entire problem end-to-end, and rely on fragmented pipelines, using off-the-shelf algorithms for detection and tracking, which ultimately results in inefficiencies. In order to tackle these issues, we propose an architecture that jointly localizes humans and produces estimates of their individual actions and their collective activity, all in a single forward pass through a neural network. The detection part of our architecture relies on a variational scheme that refines detection proposals *jointly* rather then using greedy post-processing, ultimately making detections more robust. We demonstrate the efficiency of our methods both on multi-person scene understanding and detection datasets.

Finally, in Chapter 6 we introduce Compositional VAE, a new way of learning non-linear facial geometries using deep probabilistic models. The core insight that motivates our method is that most of the existing facial models can be formulated as shallow linear autoencoders, and hence a natural way to increase their flexibility is to switch to their non-linear, deeper counterparts. We thus propose a model that is based on a deep variational autoencoder, which features multiple levels of latent variables that naturally decompose facial geometries into varying levels of deformations. The *variational* part is important because ultimately we want to use the model for fitting tasks, which require a well-behaved parameter space. In our experiments, we show that our model can be used for a variety of fitting tasks, when trained on a modestly sized database of 16 individuals. Additionally, we show that our model can be used for detail transfer between individuals.

7.2 Limitations and Future Work

In this section, we discuss the main limitations of the proposed methods and propose potential directions for the future work.

Weakly Supervised Depth-Based Detection One of the main strengths of our DPOM model introduced in Chapter 3 is that ultimately it works without any training data. However, this comes at a price - our proposed object model is rather crude, and its discriminative power is limited. A natural direction to improve model's capabilities is thus to introduce a more flexible data-driven model, e.g. by reusing ideas discussed in Chapter 6. Unfortunately, even though depth sensors are getting cheaper, depth data is still not as widely available as normal images (which can be simply crawled from the Web), and thus collecting large-scale datasets of depth images is still very expensive. Luckily, structured models like DPOM are particularly useful [114] in weakly-supervised settings, ultimately allowing us to estimate model parameters with fewer training data or by using weak supervision, such as the geometry of the environment (which is relatively easy to estimate from depth maps) or temporal consistency. In particular, a similar

variational approach *for RGB-based* multi-view setups has been proven successfull [9].

Recognizing Social Interactions In Chapter 5 we introduced a unified framework for multi-human scene understanding. Although our method already produces a comprehensive interpretation of what is happening in the scene, it is still quite limited. Namely, our model assumes that action labels can be assigned individually to each human in the scene. In other words, an action label can be seen as an *unary* predicate. However, a lot of social interactions involve at least two humans, in other words, they can be seen as *binary* predicates. Thus, one exciting direction for further research is building models that are capable of explicitly capturing such binary (and possibly higher-order) interactions. The main obstacle to this is collecting training data, since exhaustive annotations for binary interactions ultimately require labels *for each pair* of objects in the scene. Nevertheless, recently there have been some progress in this direction [90, 91].

From Detection to Instance Segmentation The detection part of the system that we presented in Chapter 5 is based on an efficient and principled way to do joint reasoning on a set of proposals. However, it still relies on *bounding boxes*, which is quite an arbitrary way to formulate the object detection problem. We can speculate that the main reason for using bounding boxes is that the training data is relatively easy to collect. However, even for human annotators the task of labelling images with bounding boxes is quite ambiguous: it is not clear if the box should be around the whole body or only its visible parts, a problem which becomes much more pronounced when there are a lot of occlusions. Luckily, now there are multiple large-scale datasets that provide fine-grained instance segmentation labels [28, 100], and thus it seems natural to extend existing detection methods to benefit from them. As discussed in Section 5.2, our method can easily adapted to this task.

Deep Probabilistic Models for Geometry In Chapter 6 we present a deep probabilistic model for learning facial geometries. The proposed model is quite generic, and thus can also be applied to other human parts or to other kinds of objects, e.g. cars. However, the core assumption behind the model is that the overall *topology* of the objects of interest is fixed, which is a valid assumption for the shapes like faces or body parts, but is not reasonable if we want to build models for arbitrary objects. One very promising direction thus is to build a separate model for the topology itself: it will ultimately allow us to build models for arbitrary meshes, and potentially benefit from transfer learning across different kinds of 3D data. However, designing such models seems to be a highly non-trivial task, as it requires jointly modeling discrete (vertices and edges) and continuous (vertex coordinates) entities. A more practical direction is to build models for arbitrary fixed

topologies, in other words, model coordinates of the vertices and provide topologies as fixed inputs. For example, one could extend PointNet-style models [121, 122], which work on point clouds, to take into account the neighbourhood structure from the given mesh topologies.

A Appendix for Chapter 3

In this appendix, we provide the derivation of the mean-field updates of Chapter 3. Please refer to Table 3.1 for the notations.

By definition X_k is a Bernoulli variable ($Q(X_k = 1) + Q(X_k = 0) = 1$), and keeping in mind Eq. 3.7:

$$Q(X_k = 1) = \frac{\exp\left(\mathbb{E}[\log P(\mathbf{D}, \mathbf{X}, \mathbf{M})|X_k = 1]\right)}{\tilde{Z}_{X_K}} \tag{A.1}$$

$$Q(X_k = 0) = \frac{\exp\left(\mathbb{E}[\log P(\mathbf{D}, \mathbf{X}, \mathbf{M})|X_k = 0]\right)}{\tilde{Z}_{X_K}} \tag{A.2}$$

which allows us to find the normalizing factor $\tilde{Z}_{X_K}$ and get the following update on $\rho_k = Q(X_k = 1)$ (Eq. 3.8):

$$\begin{aligned} \rho_k = \sigma(\quad & \mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D}, \mathbf{X}, \mathbf{M}|X_k = 1] - \\ & \mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D}, \mathbf{X}, \mathbf{M}|X_k = 0]) \end{aligned} \tag{A.3}$$

Let's take a closer look at the conditional expectation of the joint $\mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D}, \mathbf{X}, \mathbf{M})|X_k = \xi]$, $\xi \in \{0, 1\}$ (omitting $\mathbf{M}$ for clarity):

$$\begin{aligned} & \mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D}, \mathbf{X})|X_k = \xi] = \\ & \mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D}|\mathbf{X})P(\mathbf{X})|X_k = \xi] = \\ & \textstyle\sum_{i\in\mathcal{S}_k} \mathbb{E}[\log P(D_i|\mathbf{X})|X_k = \xi] + \sum_{l\in\mathcal{K}} \mathbb{E}[\log P(X_l)|X_k = \xi] \end{aligned}$$

where we used an assumption of conditional independence between pixels, and also assumption of independence of occupancies a-priori.

We now need to evaluate the expectation of the observation likelihood $\log P(D_i|\mathbf{X})$ conditioned on $X_k = \xi$ for $\xi \in \{0, 1\}$. Under our generative model, each pixel is either generated by one of the silhouettes, or by the background. If it was generated by some silhouette $\mathcal{S}_l$, then, first, all the silhouettes that are closer to the camera should be absent (which happens with probability $\tau_{l,i}$), and, second, the silhouette itself should be present (which happens independently with probability ρ_l). Otherwise, all the silhouettes are absent (probability $\tau_{|\mathcal{K}|,i}$), and pixel was generated by the background. Now, let's write down the expected log-likelihood for observing some value d at pixel $i \in \mathcal{L}$ (omitting segmentation masks for clarity):

$$\mathbb{E}_{Q(\mathbf{X})}[\log P(D_i = d|\mathbf{X})] = \sum_{l \in \mathcal{K}} \tau_{l,i}\rho_l \log \theta_{li}(d) + \tau_{|\mathcal{K}|,i} \log \theta_{bg,i}(d)$$

When conditioned on $X_k = 1$ the expectation will be:

$$\sum_{l<k} \tau_{l,i}\rho_l \log \theta_{li}(d) + \tau_{k-1,i} \log \theta_{ki}(d)$$

And conditioned when on $X_k = 0$:

$$\sum_{l<k} \tau_{l,i}\rho_l \log \theta_{li}(d) + \frac{\sum_{l>k} \tau_{l,i}\rho_l \log \theta_{li}(d) + \tau_{|\mathcal{K}|,i} \log \theta_{bg,i}(d)}{(1 - \rho_k)}$$

Now, one can evaluate $\mathbb{E}_{Q(\mathbf{X}\setminus X_k)}[\log P(\mathbf{D}, \mathbf{X})|X_k = \xi]$ (expectations for the priors is trivial, see e.g. [52]). If one substitutes these into Eq. A.3, one will get Eq. 3.10:

$$\begin{aligned}
&\rho_k = \sigma(\log \tfrac{\epsilon}{1-\epsilon} + \\
&\textstyle\sum_{i \in \mathbf{M}_k} \tau_{k,i} \log \theta_{k,i} - \\
&\textstyle\sum_{i \in \mathbf{M}_k} \frac{1}{1-\rho_k} (\sum_{l>k, i \in \mathbf{M}_l} \tau_{l,i}\rho_l \log \theta_{l,i} + \tau_{|\mathcal{K}|,i} \log \theta_{bg,i}))
\end{aligned}$$

B Appendix for Chapter 4

B.1 Proximal Gradient Mean-Field Inference

In this section, we derive the closed-form update rule for the proximal gradient descent introduced in Section 4.2 of the thesis.

Let us now consider the proximal gradient update,

$$\underset{q\in\mathcal{M}}{\text{minimize}} \left\{ \langle \mathbf{q}, \nabla\mathcal{E}(\mathbf{q}^t)\rangle - \mathcal{H}(\mathbf{q}) + \mathbf{D}^t \odot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t) \right\} , \tag{B.1}$$

where the first and the second terms are the expected energy and negative entropy respectively, and the last term is the proximal term. It can be written as

$$\mathbf{D}^t \odot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t) = \sum_{i,l} d_{i,l} \cdot q_{i,l} \log \frac{q_{i,l}}{q^t_{i,l}} , \tag{B.2}$$

where $\mathbf{D}^t$ is a diagonal matrix with non-zero elements $d_{i,l}$.

Our goal is to derive a closed-form update for all the mean parameters $q_{i,l}$, or, alternatively, for all the natural parameters $\theta_{i,l}$. We can write down the partial derivative of the expected energy with respect to any $q_{i,l}$ as

$$\nabla\mathcal{E}(\mathbf{q}^t)_{i,l} = \frac{\partial\mathcal{E}(\mathbf{q}^t)}{\partial q_{i,l}} = \mathbb{E}_{Q(\mathbf{X}|\mathbf{q}^t_{\neg i})}[\log p(\mathbf{X}|\mathbf{I})|X_i = l] . \tag{B.3}$$

Note, that both our objective $\mathcal{F}$ *and* the constraints $\mathbf{q} \in \mathcal{M}$ are separable over the variables $X_1, \ldots, X_N$, which makes it possible to minimize independently for each X_i. In

other words, our goal is to solve for all i

$$\underset{\mathbf{q}_i}{\text{minimize}} \quad \sum_l q_{i,l} \nabla \mathcal{E}(\mathbf{q}^t)_{i,l} + \sum_l q_{i,l} \log q_{i,l} + d_i^t \sum_l q_{i,l} \log \frac{q_{i,l}}{q_{i,l}^t} \,, \tag{B.4}$$

$$\text{subject to} \quad \sum_l q_{i,l} = 1 \tag{B.5}$$

We can now make each problem to unconstrained by introducing the Lagrangian

$$\begin{aligned} \mathcal{L}(\mathbf{q}_i, \mu_i) = &\sum_l q_{i,l} \nabla \mathcal{E}(\mathbf{q}^t)_{i,l} + \sum_l q_{i,l} \log q_{i,l} \,, \\ &+ d_i^t \sum_l q_{i,l} \log \frac{q_{i,l}}{q_{i,l}^t} - \mu_i \left(\sum_l q_{i,l} - 1 \right) \,, \end{aligned} \tag{B.6}$$

where μ_i is a corresponding Lagrange multiplier.

We then differentiate it with respect to $q_{i,l}$, $\forall i, l$

$$(1 + d_i^t) \log q_{i,l}^\star = \mathbb{E}_{Q(\boldsymbol{X}|\mathbf{q}_{-i})}[\log p(\mathbf{X}|\mathbf{I})|X_i = l] + d_i^t \log q_{i,l}^t + \mu_i \,, \tag{B.7}$$

which in turn leads to the update rule

$$q_{i,l}^{t+1} \propto \exp\left[\eta_i^t \cdot \mathbb{E}_{Q(\boldsymbol{X}|\mathbf{q}_{-i})}[\log p(\mathbf{X}|\mathbf{I})|X_i = l] + (1 - \eta_i^t) \cdot \log q_{i,l}^t \right] \,, \tag{B.8}$$

where $\eta_i^t = \frac{1}{1+d_i^t}$, and normalization constant can be obtained from μ_i.

B.2 Proving Convergence

In this section, we prove that the fixed step-size algorithm introduced in Section 4.2.2 guarantess convergence. In the remainder of this appendix, we will work under the assumption that

$$\forall i, t \;\; \exists d_i^t \text{ s.t. } \forall l \; d_{i,l}^t = d_i^t,$$

which is verified for the fixed and adaptive step size and methods described in Chapter 4. We will therefore replace $d_{i,l}$ by d_i in the subsequent derivations. Note that, this property does not hold for `OURS-ADAM`. Nevertheless, as shown in the experimental evaluation, in practice it tends to converge faster and to a better minima.

Lemma B.2.1 *The gradient of the proximal term at the current iteration point* $\nabla_{\mathbf{q}} \mathbf{D}^t \odot KL(\mathbf{q}||\mathbf{q}^t)|_{\mathbf{q}=\mathbf{q}^t}$ *is orthogonal to* $\mathcal{M}$.

Proof Let's write down the gradient:

$$\nabla_{\mathbf{q}}\mathbf{D}^t \odot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t) = (d_1^t \cdot \nabla_{\mathbf{q}_1}\mathrm{KL}(\mathbf{q}_1||\mathbf{q}_1^t), \ldots, d_N^t \nabla_{\mathbf{q}_N}\mathrm{KL}(\mathbf{q}_N||\mathbf{q}_N^t)) \ , \tag{B.9}$$

with each component containing:

$$\nabla_{\mathbf{q}_i}\mathrm{KL}(\mathbf{q}_i||\mathbf{q}_i^t) = (\log\frac{q_{i,1}}{q_{i,1}^t} + 1, \ldots, \log\frac{q_{i,M}}{q_{i,M}^t} + 1) \ . \tag{B.10}$$

The partial gradient at the current iteration point $\mathbf{q}_i^t$ is the all-ones vector:

$$\nabla_{\mathbf{q}_i}\mathrm{KL}(\mathbf{q}_i||\mathbf{q}_i^t)|_{\mathbf{q}_i=\mathbf{q}_i^t} = (1, \ldots, 1) \ , \tag{B.11}$$

which is obviously orthogonal to the hyperplane defined by the constraint $\sum_l q_{i,l} = 1$. Thus, $d_i^t \nabla_{\mathbf{q}_i}\mathrm{KL}(\mathbf{q}_i||\mathbf{q}_i^t)|_{\mathbf{q}_i=\mathbf{q}_i^t}$ is also orthogonal to this hyperplane, and we easily obtain the orthogonality of the product vector $\nabla_{\mathbf{q}}\mathbf{D}^t \odot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t)|_{\mathbf{q}=\mathbf{q}^t}$ to $\mathcal{M}$.

Lemma B.2.2 *For all* $\mathbf{q}^t$ *in* $\mathcal{M}$*,*

$$\forall \mathbf{q} \in \mathcal{M},\ \mathbf{D}^t \cdot KL(\mathbf{q}^{t+1}||\mathbf{q}^t) \geq \frac{L}{2}\|\mathbf{q} - \mathbf{q}^t\|_2^2 \ .$$

Proof We start with noting that the Hessian of the KL-proximal term is diagonal with

$$\forall \mathbf{q} \in \mathcal{M},\ \frac{\partial^2 \mathbf{D}^t \cdot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t)}{\partial q_{i,l}^2}|_{\mathbf{q}} = \frac{d_{i,l}^t}{q_{i,l}} \geq L \ . \tag{B.12}$$

Therefore, the proximal term is L-strongly convex on $\mathcal{M}$. For all $\mathbf{q}^t$ in $\mathcal{M}$,

$$\forall \mathbf{q} \in \mathcal{M},\ \mathbf{D}^t \cdot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t) \geq \langle \nabla_{\mathbf{q}}\mathbf{D}^t \odot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t)|_{\mathbf{q}=\mathbf{q}^t}, \mathbf{q} - \mathbf{q}^t \rangle + \frac{L}{2}\|\mathbf{q} - \mathbf{q}^t\|_2^2 \ . \tag{B.13}$$

The first term of the right hand side is null according to the orthogonality property B.2.1. Which leads to

$$\forall \mathbf{q} \in \mathcal{M},\ \ \mathbf{D}^t \cdot \mathrm{KL}(\mathbf{q}^{t+1}||\mathbf{q}^t) \geq \frac{L}{2}\|\mathbf{q} - \mathbf{q}^t\|_2^2 \ . \tag{B.14}$$

Further, we now demonstrate, that under certain assumptions, applying updates of Eq. B.8 leads to a decrease in objective at each iteration.

Theorem B.2.3 *If* $\mathcal{E}$ *is L-Lipschitz gradient on* $\mathcal{M}$*, and that* d_i^t*s are chosen such that* $d_i^t \geq L$*,* $\forall t, i$*. Then the objective function is decreasing at each step.*

Proof Let us assume that $\mathcal{E}$ is L-Lipschitz gradient on $\mathcal{M}$ and that $d_i^t \geq L$, $\forall t, i$. Then, we can show that the value of the objective function $\mathcal{E}(\mathbf{q}^{t+1}) - \mathcal{H}(\mathbf{q}^{t+1})$ at step $t+1$ has to be smaller than $\mathcal{E}(\mathbf{q}^t) - \mathcal{H}(\mathbf{q}^t)$

$$\mathcal{E}(\mathbf{q}^t) - \mathcal{H}(\mathbf{q}^t) \geq \arg\min_{\mathbf{q}} \left[\mathcal{E}(\mathbf{q}^t) + \langle(\mathbf{q} - \mathbf{q}^t), \nabla\mathcal{E}(\mathbf{q}^t)\rangle - \mathcal{H}(\mathbf{q}) + \mathbf{D}^t \cdot \mathrm{KL}(\mathbf{q}||\mathbf{q}^t)\right] \tag{B.15}$$

$$\geq \mathcal{E}(\mathbf{q}^t) + \langle(\mathbf{q}^{t+1} - \mathbf{q}^t), \nabla\mathcal{E}(\mathbf{q}^t)\rangle - \mathcal{H}(\mathbf{q}^{t+1}) + \mathbf{D}^t \cdot \mathrm{KL}(\mathbf{q}^{t+1}||\mathbf{q}^t) \tag{B.16}$$

$$\geq \mathcal{E}(\mathbf{q}^t) + \langle(\mathbf{q}^{t+1} - \mathbf{q}^t), \nabla\mathcal{E}(\mathbf{q}^t)\rangle - \mathcal{H}(\mathbf{q}^{t+1}) + \frac{L}{2}\|\mathbf{q}^{t+1} - \mathbf{q}^t\|_2^2 \tag{B.17}$$

$$\geq \mathcal{E}(\mathbf{q}^{t+1}) - \mathcal{H}(\mathbf{q}^{t+1}) \tag{B.18}$$

where step Eq. B.16 comes from the fact that by definition $\mathbf{q}^{t+1}$ realizes the minimum, Eq. B.17 holds by strong-convexity lower bound B.2.2 and Eq. B.18 holds by L-Lipschitz gradient property of $\mathcal{E}$.

B.3 Adaptive Steps

We now formally justify the update rule used in Section 4.2.3 of the thesis. In the proof of Lemma B.2.2, in Eq. B.12, we used the fact that $\frac{1}{q_{i,l}} \geq 1$. This bound is correct, but can be too large since ultimately $q_{i,l}$ can be very close to 0. This leads to the choice of $d_i = L$, $\forall i$, which ensures $\frac{d_i}{q_{i,l}} \geq L$.

Alternatively, one can choose a smaller value $d_i = L \max(q_{i,0}^t, \ldots, q_{i,L_i-1}^t)$, which also ensures that $\frac{d_i}{q_{i,l}^t} \geq L$, $\forall i, l$ but unfortunately the gain is still very minor.

However, all the previous bounds ignore the fact that all our variables lie on the simplex $\mathcal{M}$. We will now show, that one can obtain a proximal term that provides a much closer local upper-bound for the objective function.

Let us start by writing a second-order Taylor expansion of the KL-proximal term for

variable i around the current iteration point. This yields

$$\begin{aligned} d_i^t \cdot \mathrm{KL}(\mathbf{q}^{t+1}||\mathbf{q}^t) =& d_i^t \langle \nabla_{\mathbf{q}_i} \mathrm{KL}(\mathbf{q}_i||\mathbf{q}_i^t)|_{\mathbf{q}_i=\mathbf{q}_i^t}, \mathbf{q}_i^{t+1} - \mathbf{q}_i^t \rangle \\ &+ \frac{d_i^t}{2} \sum_l \frac{(q_{i,l}^{t+1} - q_{i,l}^t)^2}{q_{i,l}^t} + o(\|\mathbf{q}_i^{t+1} - \mathbf{q}_i^t\|_2^2) \qquad \text{(B.19)} \\ =& \frac{d_i^t}{2} \sum_l \frac{(q_{i,l}^{t+1} - q_{i,l}^t)^2}{q_{i,l}^t} + o(\|\mathbf{q}_i^{t+1} - \mathbf{q}_i^t\|_2^2) \ , \qquad \text{(B.20)} \end{aligned}$$

where we used Lemma B.2.1 to get Eq. B.20.

For a derivation similar to Eq. B.15-Eq. B.18 to hold (up to a second order approximation), we need to choose d_i^t so that

$$d_i^t \sum_l \frac{(q_{i,l}^{t+1} - q_{i,l}^t)^2}{q_{i,l}^t} \geq L \|\mathbf{q}_i^{t+1} - \mathbf{q}_i^t\|_2^2 \ .$$

However, we also should take into account the fact that both $\mathbf{q}^{t+1}$ and q^t lie in $\mathcal{M}$, and therefore $\sum_l q_{i,l}^{t+1} - q_{i,l}^t = 0$. Finally, one can choose $\frac{L}{d_i^t}$ as the optimum of the following program:

$$\begin{aligned} \underset{\delta}{\text{minimize}} \quad & \sum_l \frac{\delta_l^2}{q_{i,l}^t} \ , \\ \text{subject to} \quad & \sum_l \delta_l = 0 \ , \qquad \text{(B.21)} \\ & \sum_l \delta_l^2 = 1 \ . \end{aligned}$$

C Appendix for Chapter 5

In this appendix we provide derivations for the mean-field updates of Chapter 5 (Eq. 5.6-5.7)

Recall that our goal is to refine a dense set of detection hypotheses produced by a fully-convolutional neural network, by doing inference in the following model:

$$P(\mathbf{X}, \mathbf{A}) \propto \prod_{i,j} \exp\left(-\frac{\mathbb{1}[A_i = j] \cdot ||\mathbf{X}_i - \mathbf{X}_j||_2^2}{2\sigma^2}\right) , \tag{C.1}$$

where for each hypothesis location $i \in \mathcal{I} = \{1, \dots, H \times W\}$ we introduce two hidden variables, one multivariate Gaussian $\mathbf{X}_i \in \mathbb{R}^4$, and one categorical $A_i \in \mathcal{I}$. Note that, we use $\mathbb{1}[A_i = j]$ to denote the function that equals one if $A_i = j$ and zero otherwise. $\mathbf{X}_i$ encodes the "true" coordinates of the detection, and A_i encodes the assignment of the detection to one of the hypothesis locations in $\mathcal{I}$. Ultimately, we want to estimate the marginal distributions $P(\mathbf{X}_i), P(A_i), \forall i \in \mathcal{I}$. However, integrating Eq. C.1 directly is not tractable, and thus we resort to mean-field approximation. We introduce the following parameterized fully-factorized distribution:

$$Q(\mathbf{X}, \mathbf{A}) = \prod_i Q(\mathbf{X}_i)Q(A_i) = \prod_i \mathcal{N}(\mathbf{X}_i\,; \boldsymbol{\mu}_i, \sigma^2)\mathrm{Cat}(A_i\,; \boldsymbol{\eta}_i) , \tag{C.2}$$

where $\boldsymbol{\mu}_i \in \mathbb{R}^4$ and $\boldsymbol{\eta}_i \in \mathbb{R}^{|\mathcal{I}|}$ are the variational parameters of the Gaussian and categorical distributions respectively. The categorical distribution is a generalization of the Bernoulli distribution and is defined as follows:

$$\mathrm{Cat}(A_i\,; \boldsymbol{\eta}_i) = \prod_j \left(\frac{e^{\eta_{ij}}}{\sum_k e^{\eta_{ik}}}\right)^{\mathbb{1}[A_i=j]} = \prod_j \alpha_{ij}^{\mathbb{1}[A_i=j]} , \tag{C.3}$$

where $\alpha_{ij} = \frac{e^{\eta_{ij}}}{\sum_k e^{\eta_{ik}}}$ is a probability of $A_i = j$. Note that, we are using two ways to parameterize this distribution: through *natural* parameters $\boldsymbol{\eta}_i$ and *mean* parameters $\boldsymbol{\alpha}_i$.

Both of them are equivalent, but the former is somewhat more convenient for deriving the updates for $Q(A_i)$, whereas the later is used in the updates for $Q(\mathbf{X}_i)$.

The goal of the variational inference is to find the values of the parameters, such that $Q(\mathbf{X}, \mathbf{A})$ closely approximates $P(\mathbf{X}, \mathbf{A})$. This is done by minimizing the Kullback-Leibler (KL) divergence between the two:

$$KL(Q||P) = -\int_{\mathbf{X},\mathbf{A}} Q(\mathbf{X}, \mathbf{A}) \log \frac{P(\mathbf{X}, \mathbf{A})}{Q(\mathbf{X}, \mathbf{A})} \,. \tag{C.4}$$

As we saw in Section 2.3, optimizing Eq. C.4 can be done by iteratively applying the following updates for each of the approximate marginals $Q(\mathbf{X}_i), Q(A_i)$:

$$\begin{aligned} \log Q^{\tau}(\mathbf{X}_i) &\propto \mathbb{E}_{Q^{\tau-1}_{\neg \mathbf{X}_i}}[\log P(\mathbf{X}, \mathbf{A})] \,, \\ \log Q^{\tau}(A_i) &\propto \mathbb{E}_{Q^{\tau-1}_{\neg A_i}}[\log P(\mathbf{X}, \mathbf{A})] \,, \end{aligned} \tag{C.5}$$

where $\mathbb{E}[\cdot]$ denotes expectation, and $Q^{\tau-1}_{\neg \mathbf{X}_i}$ indicates that the expectation is computed w.r.t. approximate distribution over all the variables *except* $\mathbf{X}_i$ (similarly for A_i). We now can substitute Eq. C.1 and Eq. C.2 into Eq. C.5, to get the fixed-point updates for the variational parameters.

Let us start with the parameters of $Q(A_i)$, using that $\mathbb{E}[\mathbf{X}_i]_{Q^{\tau}} = \boldsymbol{\mu}_i^{\tau}$:

$$\log Q^{\tau}(A_i = j) \propto \eta_{ij} \propto -\mathbb{E}_{Q^{\tau-1}_{\neg A_i}}[\frac{\mathbb{1}[A_i = j] \cdot ||\mathbf{X}_i - \mathbf{X}_j||_2^2}{2\sigma^2}] \tag{C.6}$$

$$= -\frac{||\,\mathbb{E}_{Q^{\tau-1}_{\neg A_i}}[\mathbf{X}_i] - \mathbb{E}_{Q^{\tau-1}_{\neg A_i}}[\mathbf{X}_j]||_2^2}{2\sigma^2} \tag{C.7}$$

$$= -\frac{||\boldsymbol{\mu}_i^{\tau-1} - \boldsymbol{\mu}_j^{\tau-1}||_2^2}{2\sigma^2} \,. \tag{C.8}$$

Let's now derive the updates for $Q(\mathbf{X}_i)$. Noting that $\mathbb{E}_{Q^{\tau}}[\mathbb{1}[A_i = j]] = \alpha_{ij}^{\tau}$, $\sum_j \alpha_{ij}^{\tau} = 1$,

we can write:

$$\log Q^{\tau}(\mathbf{X}_i) = -\mathbb{E}_{Q^{\tau-1}_{\neg \mathbf{X}_i}}\left[\frac{\mathbb{1}[A_i = j] \cdot ||\mathbf{X}_i - \mathbf{X}_j||_2^2}{2\sigma^2}\right] + \text{const} \tag{C.9}$$

$$= -\sum_j \alpha_{ij} \mathbb{E}_{Q^{\tau-1}_{\neg \mathbf{X}_i}}\left[\frac{||\mathbf{X}_i - \mathbf{X}_j||_2^2}{2\sigma^2}\right] + \text{const} \tag{C.10}$$

$$= -\sum_j \alpha_{ij} \frac{1}{2\sigma^2} \mathbb{E}_{Q^{\tau-1}_{\neg \mathbf{X}_i}}\left[\mathbf{X}_i^T\mathbf{X}_i - 2\mathbf{X}_i^T\mathbf{X}_j + \mathbf{X}_j^T\mathbf{X}_j\right] + \text{const} \tag{C.11}$$

$$\approx -(\sum_j \alpha_{ij})\frac{1}{2\sigma^2}\mathbf{X}_i^2 + \frac{(\sum_j \alpha_{ij}\boldsymbol{\mu}_j^{\tau-1})^T}{\sigma^2}\mathbf{X}_i + \text{const} \tag{C.12}$$

$$=(-\frac{1}{2\sigma^2})\mathbf{X}_i^T\mathbf{X}_i + \frac{1}{\sigma^2}(\sum_j \alpha_{ij}\boldsymbol{\mu}_j^{\tau-1})^T\mathbf{X}_i + \text{const}\ , \tag{C.13}$$

where the approximation in Eq. C.12 comes from the fact that we substitute $\mathbf{X}_j$ with $\boldsymbol{\mu}_j^{\tau-1}$, for $j : j = i$.

Note that, since $Q^{\tau}(\mathbf{X}_i)$ is in exponential family and is also Gaussian with parameters $\boldsymbol{\mu}_i^{\tau}, \sigma^2$, we can always represent its log-PDF as follows:

$$\log Q^{\tau}(\mathbf{X}_i) = (-\frac{1}{2\sigma^2})\mathbf{X}_i^T\mathbf{X}_i + \frac{1}{\sigma^2}\boldsymbol{\mu}_i^T\mathbf{X}_i + \text{const}. \tag{C.14}$$

Given Eq. C.13 and Eq. C.14, we can simply match the first and the second moments, and get the following update for the variational parameters of $Q(\mathbf{X}_i)$:

$$\boldsymbol{\mu}_i^{\tau} = \sum_j \alpha_{ij}\boldsymbol{\mu}_j^{\tau-1}\ . \tag{C.15}$$

As mentioned in the Chapter 5, in practice we run inference for a fixed number of iterations $\mathcal{T}$, starting from initial $\boldsymbol{\mu}^0$ which we set to the dense proposals obtained from our fully-convolutional detection network, and use the following smoothed updates for $\boldsymbol{\mu}^{\tau}$:

$$\boldsymbol{\mu}_i^{\tau} = \lambda\boldsymbol{\mu}_i^{\tau-1} + (1-\lambda)\sum_j \alpha_{ij}\boldsymbol{\mu}_j^{\tau-1}\ , \tag{C.16}$$

where λ is a damping parameter, which we keep fixed, although ultimately one can interpret it as a step-size, as discussed in Chapter 4, and use more powerful first-order optimization schemes, such as ADAM [81].

In practice, we observe that the estimates of the variational parameters converge within 10-100 iterations, depending on the choice of σ. We leave the formal analysis of the

convergence properties of our approach as future work. Note that, the standard mean-field proofs do not strictly apply because we update all parameters in parallel, and the existing proofs for parallel updates, such as that in Appendix B, are derived for discrete random fields.

D Appendix for Chapter 6

In this appendix, we provide additional results for the Compositional VAE. The representations that our model learns naturally decomposes geometry into meaningful level of detail. This allows us to do detail transfer: i.e. transfer either high-frequency detail (Figure D.1), such as beards or wrinkles, while keeping the identity and general expression; or low-frequency detail (Figure D.2), such as the general expressions, while keeping the smaller skin deformations. Note that, these images are produced purely by doing interpolation in the learned latent space, without any manually specified geometrical priors.

Figure D.1: Transferring high frequency detail. We visualize the change of shapes when interpolating high frequency details while keeping low frequeoncy details fixed. The leftmost and rightmost columns are the two original meshes.

Figure D.2: Transferring low frequency detail. We visualize the change of shapes when interpolating low frequency detail while keeping high frequency details fixed. The leftmost and rightmost columns are the two original meshes.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] S.-I. Amari. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10(2):251–276, 1998.

[3] M. R. Amer, P. Lei, and S. Todorovic. Hirf: Hierarchical random field for collective activity recognition in videos. In *European Conference on Computer Vision*, 2014.

[4] C. Andrieu, N. De Freitas, A. Doucet, and M. I. Jordan. An introduction to mcmc for machine learning. *Machine Learning*, 50(1-2):5–43, 2003.

[5] T. Bagautdinov, A. Alahi, F. Fleuret, P. Fua, and S. Savarese. Social Scene Understanding: End-To-End Multi-Person Action Localization and Collective Activity Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[6] T. Bagautdinov, P. Fua, and F. Fleuret. Probability Occupancy Maps for Occluded Depth Images. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[7] T. Bagautdinov, C. Wu, J. Saragih, P. Fua, and Y. Sheikh. Modeling Facial Geometry Using Compositional VAEs. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[8] P. Baqué, T. Bagautdinov, F. Fleuret, and P. Fua. Principled Parallel Mean-Field Inference for Discrete Random Fields. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[9] P. Baqué, F. Fleuret, and P. Fua. Deep Occlusion Reasoning for Multi-Camera Multi-Target Detection. In *International Conference on Computer Vision*, 2017.

[10] O. Barinova, V. Lempitsky, and P. Kohli. On Detection of Multiple Object Instances Using Hough Transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9), 2012.

[11] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross. High-quality single-shot capture of facial geometry. *ACM Transactions on Graphics*, 29(4):40:1–40:9, July 2010.

[12] H. BenShitrit, J. Berclaz, F. Fleuret, and P. Fua. Multi-Commodity Network Flow for Tracking Multiple People. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1614–1627, 2014.

[13] K. Bernardin and R. Stiefelhagen. Evaluating Multiple Object Tracking Performance: the Clear Mot Metrics. *EURASIP Journal on Image and Video Processing*, 2008, 2008.

[14] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[15] M. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *International Conference on Computer Vision*, 1995.

[16] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *ACM SIGGRAPH*, pages 187–194, 1999.

[17] Y. Boykov, O. Veksler, and R. Zabih. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[18] A. Brunton, T. Bolkart, and S. Wuhrer. Multilinear Wavelets: A Statistical Shape Space for Human Faces. In *European Conference on Computer Vision*, 2014.

[19] A. Bulat and G. Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *International Conference on Computer Vision*, 2017.

[20] N. D. Campbell, K. Subr, and J. Kautz. Fully-connected crfs with non-parametric pairwise potential. In *Conference on Computer Vision and Pattern Recognition*, pages 1658–1665. IEEE, 2013.

[21] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, 2014.

[22] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In *International Conference for Learning Representations*, 2015.

[23] W. Choi and S. Savarese. A unified framework for multi-target tracking and collective activity recognition. In *European Conference on Computer Vision*, 2012.

[24] W. Choi and S. Savarese. Understanding collective activities of people from videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1242–1257, 2014.

[25] W. Choi, K. Shahid, and S. Savarese. Learning context for collective activity recognition. In *Conference on Computer Vision and Pattern Recognition*, 2011.

[26] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv Preprint*, 2014.

[27] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[28] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[29] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[30] D. DeCarlo and D. Metaxas. The Integration of Optical Flow and Deformable Models with Applications to Human Face Shape and Motion Estimation. In *Conference on Computer Vision and Pattern Recognition*, 1996.

[31] Z. Deng, A. Vahdat, H. Hu, and G. Mori. Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition. In *Conference on Computer Vision and Pattern Recognition*, June 2016.

[32] J. J. DiCarlo, D. Zoccolan, and N. C. Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012.

[33] M. Dimitrijevic, S. Ilic, and P. Fua. Accurate face models from uncalibrated and ill-lit video sequences. In *Conference on Computer Vision and Pattern Recognition*, 2004.

[34] P. Dollár. Piotr's Computer Vision Matlab Toolbox (PMT). http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html.

[35] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast Feature Pyramids for Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[36] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral Channel Features. In *British Machine Vision Conference*, pages 1–11, 2009.

[37] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: A Benchmark. In *Conference on Computer Vision and Pattern Recognition*, June 2009.

[38] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[39] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. In *International Conference for Learning Representations*, 2017.

[40] T. D'Orazio, M. Leo, N. Mosca, P. Spagnolo, and P. L. Mazzeo. A Semi-Automatic System for Ground Truth Generation of Soccer Video Sequences. In *International Conference on Advanced Video and Signal Based Surveillance*, pages 559–564, 2009.

[41] P. Dou, S. K. Shah, and I. A. Kakadiaris. End-to-end 3d face reconstruction with deep neural networks. *Conference on Computer Vision and Pattern Recognition*, 2017.

[42] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[43] J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite Objective Mirror Descent. In *COLT*, pages 14–26, 2010.

[44] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. In *International Conference for Learning Representations*, 2017.

[45] C. N. Duong, K. Luu, K. G. Quach, and T. D. Bui. Deep appearance models: A deep boltzmann machine approach for face modeling. *arXiv Preprint*, 2016.

[46] I. Essa, S. Basu, T. Darrell, and A. Pentland. Modeling, tracking and interactive animation of faces and heads using input from video. In *Computer Animation*, page 68, 1996.

[47] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[48] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results.

[49] L. Fei-fei and P. Perona. A Bayesian Hierarchical Model for Learning Natural Scene Categories. In *Conference on Computer Vision and Pattern Recognition*, 2005.

[50] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[51] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[52] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multi-Camera People Tracking with a Probabilistic Occupancy Map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):267–282, February 2008.

[53] K. Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127, 2010.

[54] R. Frostig, S. Wang, P. Liang, and C. Manning. Simple MAP Inference via Low-Rank Relaxations. In *Advances in Neural Information Processing Systems*, pages 3077–3085, 2014.

[55] M. C. Fu. Gradient estimation. *Handbooks in operations research and management science*, 13:575–616, 2006.

[56] P. Fua. Regularized Bundle-Adjustment to Model Heads from Image Sequences Without Calibration Data. *International Journal of Computer Vision*, 38(2):153–171, July 2000.

[57] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky. Hough Forests for Object Detection, Tracking, and Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2188–2202, 2011.

[58] A. Gelfand and A. Smith. Sampling-Based Approaches to Calculating Marginal Densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

[59] Z. Ghahramani and M. J. Beal. Propagation algorithms for variational bayesian learning. In *Advances in Neural Information Processing Systems*, 2001.

[60] R. Girshick. Fast R-CNN. In *International Conference on Computer Vision*, 2015.

[61] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

[62] L. Gorelick, Y. Boykov, O. Veksler, A. Ben, and A. Delong. Submodularization for Binary Pairwise Energies. In *Conference on Computer Vision and Pattern Recognition*, pages 1154–1161, 2014.

[63] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[64] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[65] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[66] X. He, R. Zemel, and M. Carreira-Perpinan. Multiscale Conditional Random Fields for Image Labeling. In *Conference on Computer Vision and Pattern Recognition*, pages 695–702, 2004.

[67] X. He, R. Zemel, and D. Ray. Learning and Incorporating Top-Down Cues in Image Segmentation. In *European Conference on Computer Vision*, 2006.

[68] M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic Variational Inference. *Journal of Machine Learning Research*, 14(1):1303–1347, May 2013.

[69] M. S. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and G. Mori. A Hierarchical Deep Temporal Model for Group Activity Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[70] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*, 2015.

[71] O. Jafari, D. Mitzel, and B. Leibe. Real-Time RGB-D Based People Detection and Tracking for Mobile Robots and Head-Worn Cameras. In *International Conference on Robotics and Automation*, pages 5636–5643, 2014.

[72] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013.

[73] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[74] N. Jojic and B. Frey. Learning Flexible Sprites in Video Layers. In *Conference on Computer Vision and Pattern Recognition*, 2001.

[75] P. Joshi, W. C. Tien, M. Desbrun, and F. Pighin. Learning controls for blend shape based realistic facial animation. In *ACM Symposium on Computer Animation*, pages 187–192, 2003.

[76] J. Kappes, B. Andres, F. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. Kausler, T. Kröger, J. Lellmann, et al. A Comparative Study of Modern Inference Techniques for Structured Discrete Energy Minimization Problems. *International Journal of Computer Vision*, 115(2):155–184, 2015.

[77] S. Khamis, V. Morariu, and L. Davis. Combining per-frame and per-track cues for multi-person action recognition. In *European Conference on Computer Vision*, 2012.

[78] S. Khamis, V. I. Morariu, and L. S. Davis. A flow model for joint action recognition and identity maintenance. In *Conference on Computer Vision and Pattern Recognition*, 2012.

[79] M. E. Khan, P. Baqué, F. Fleuret, and P. Fua. Kullback-Leibler Proximal Variational Inference. In *Advances in Neural Information Processing Systems*, 2015.

[80] Kinect for Windows SDK 2.0, 2014. http://www.microsoft.com/en-us/kinectforwindows/.

[81] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimisation. In *International Conference for Learning Representations*, 2015.

[82] D. P. Kingma, T. Salimans, and M. Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 2016.

[83] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *International Conference for Learning Representations*, 2014.

[84] P. Kohli, L. Ladicky, and P. Torr. Robust Higher Order Potentials for Enforcing Label Consistency. In *Conference on Computer Vision and Pattern Recognition*, 2008.

[85] P. Kohli and C. Rother. Higher-Order Models in Computer Vision. In O. Lezoray and L. Grady, editors, *Image Processing and Analysis with Graphs*, pages 65–100. CRC Press, 2012.

[86] D. Koller and N. Friedman. *Probabilistc Graphical Models*. The MIT Press, 2009.

[87] V. Kolmogorov. A New Look at Reweighted Message Passing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(5):919–930, 2015.

[88] P. Krähenbühl and V. Koltun. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In *Advances in Neural Information Processing Systems*, 2011.

[89] P. Krähenbühl and V. Koltun. Parameter Learning and Convergent Inference for Dense Random Fields. In *International Conference on Machine Learning*, pages 513–521, 2013.

[90] R. Krishna, I. Chami, M. Bernstein, and L. Fei-Fei. Referring relationships. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[91] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

[92] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, pages 1106–1114, 2012.

[93] L. Ladicky, P. Sturgess, K. Alahari, C. Russell, and P. Torr. What, Where and How Many? Combining Object Detectors and CRFs. In *European Conference on Computer Vision*, pages 424–437, 2010.

[94] J. Lafferty, A. Mccallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *International Conference on Machine Learning*, pages 282–289, 2001.

[95] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Conference on Computer Vision and Pattern Recognition*, 2008.

[96] M. Lau, J. Chai, Y.-Q. Xu, and H.-Y. Shum. Face poser: Interactive modeling of 3d facial expressions using facial priors. *ACM Transactions on Graphics*, 29(1):3:1–3:17, Dec. 2009.

[97] Y. Lecun, S. Chopra, R. Hadsell, M. A. Ranzato, and F. J. Huang. A tutorial on energy-based learning. In *Predicting structured data*. MIT Press, 2006.

[98] J. P. Lewis, K. Anjyo, T. Rhee, M. Zhang, F. Pighin, and Z. Deng. Practice and Theory of Blendshape Facial Models. In S. Lefebvre and M. Spagnuolo, editors, *Eurographics 2014 - State of the Art Reports*. The Eurographics Association, 2014.

[99] H. Li, P. Roivainen, and R. Forcheimer. 3-D motion estimation in model-based facial image coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):545–555, 1993.

[100] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, 2014.

[101] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. Berg. SSD: Single Shot Multibox Detector. In *European Conference on Computer Vision*, 2016.

[102] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[103] M. Luber, L. Spinello, and K. Arras. People Tracking in Rgb-D Data with On-Line Boosted Target Models. In *International Conference on Intelligent Robots and Systems*, pages 3844–3849, 2011.

[104] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[105] A. Miller, N. Foti, A. D'Amour, and R. P. Adams. Reducing reparameterization gradient variance. In *Advances in Neural Information Processing Systems*, 2017.

[106] T. Minka. Expectation Propagation for Approximate Bayesian Inference. In *Conference on Uncertainty in Artificial Intelligence*, pages 362–369, 2001.

[107] A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, 2014.

[108] M. Munaro and E. Menegatti. Fast RGB-D People Tracking for Service Robots. *Autonomous Robots*, 37(3):227–242, 2014.

[109] K. Murphy, Y. Weiss, and M. Jordan. Loopy Belief Propagation for Approximate Inference: An Empirical Study. In *Onference on Uncertainty in Artificial Intelligence*, pages 467–475, 1999.

[110] K.-G. Na and M.-R. Jung. Local shape blending using coherent weighted regions. *The Visual Computer*, 27(6-8):575–584, 2011.

[111] R. M. Neal. Probabilistic inference using markov chain monte carlo methods. *Technical Report No. CRG-TR-93-1*, 1993.

[112] T. Neumann, K. Varanasi, S. Wenger, M. Wacker, M. Magnor, and C. Theobalt. Sparse localized deformation components. *ACM Transactions on Graphics*, 32(6):179:1–179:10, 2013.

[113] S. Nowozin, P. Gehler, and C. Lampert. On Parameter Learning in CRF-Based Approaches to Object Class Image Segmentation. In *European Conference on Computer Vision*, pages 98–111, 2010.

[114] S. Nowozin, C. H. Lampert, et al. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3–4):185–365, 2011.

[115] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kholi. Decision Tree Fields. In *International Conference on Computer Vision*, November 2011.

[116] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *arXiv Preprint*, 2016.

[117] G. Papandreou, L.-C. Chen, K. Murphy, and A. Yuille. Weakly-And Semi-Supervised Learning of a DCNN for Semantic Image Segmentation. *arXiv Preprint*, 2015.

[118] M. A. Pinsk, K. DeSimone, T. Moore, C. G. Gross, and S. Kastner. Representations of faces and body parts in macaque temporal cortex: a functional mri study. *Proceedings of the National Academy of Sciences of the United States of America*, 102(19):6996–7001, 2005.

[119] B. Polyak. Some Methods of Speeding Up the Convergence of Iteration Methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[120] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin. Variational autoencoder for deep learning of images, labels and captions. In *Advances in neural information processing systems*, pages 2352–2360, 2016.

[121] C. Qi, H. Su, K. Mo, and L. Guibas. Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[122] C. Qi, L. Yi, H. Su, and L. Guibas. Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv preprint arXiv:1706.02413*, 2017.

[123] V. Ramanathan, J. Huang, S. Abu-El-Haija, A. Gorban, K. Murphy, and L. Fei-Fei. Detecting events and key actors in multi-person videos. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[124] R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In *International Conference on Artificial Intelligence and Statistics*, volume 33, pages 814–822, 2014.

[125] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[126] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, 2015.

[127] D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. In *Journal of Machine Learning Research*, 2015.

[128] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286, 2014.

[129] E. Richardson, M. Sela, and R. Kimmel. 3d face reconstruction by learning from synthetic data. In *International Conference on 3D Vision*, 2016.

[130] E. Richardson, M. Sela, R. Or-El, and R. Kimmel. Learning detailed face reconstruction from a single image. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[131] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019, 1999.

[132] H. Robbins and S. Monro. A Stochastic Approximation Methods. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[133] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Conference on Medical Image Computing and Computer Assisted Intervention*, 2015.

[134] R. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *International Conference on Robotics and Automation*, Shanghai, China, 2011.

[135] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. A semi-automatic methodology for facial landmark annotation. In *Conference on Computer Vision and Pattern Recognition Workshops*, 2013.

[136] M. Saito, T. Okatani, and K. Deguchi. Application of the Mean Field Methods to MRF Optimization in Computer Vision. In *Conference on Computer Vision and Pattern Recognition*, June 2012.

[137] T. Salimans, D. Kingma, and M. Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*, volume 37, pages 1218–1226, 2015.

[138] J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, 91(2):200–215, 2011.

[139] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv Preprint*, 2013.

[140] A. K. Seth. *The cybernetic Bayesian brain*. Open MIND. Frankfurt am Main: MIND Group, 2014.

[141] J. Shotton, A. Fitzgibbon, M. Cook, and A. Blake. Real-Time Human Pose Recognition in Parts from a Single Depth Image. In *Conference on Computer Vision and Pattern Recognition*, 2011.

[142] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-Time Human Pose Recognition in Parts from Single Depth Images. *Communications of the ACM*, 56(1):116–124, 2013.

[143] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv Preprint*, 2014.

[144] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[145] S. Singh, C. Arora, and C. V. Jawahar. First person action recognition using deep learned descriptors. In *Conference on Computer Vision and Pattern Recognition*, June 2016.

[146] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 109–116, 2007.

[147] L. Spinello and K. Arras. People Detection in RGB-D Data. In *International Conference on Intelligent Robots and Systems*, pages 3838–3843, 2011.

[148] R. Stewart, M. Andriluka, and A. Y. Ng. End-to-end people detection in crowded scenes. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[149] X. Sun, M. Christoudias, V. Lepetit, and P. Fua. Real-Time Landing Place Assessment in Man-Made Environments. *Machine Vision and Applications*, 2013.

[150] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *arXiv Preprint*, 2015.

[151] M. Teboulle. Entropic Proximal Mappings with Applications to Nonlinear Programming. *Mathematics of Operations Research*, 17(3):670–690, 1992.

[152] J. R. Tena, F. De la Torre, and I. Matthews. Interactive region-based linear 3d face models. *ACM Transactions on Graphics*, 30(4):76:1–76:10, 2011.

[153] A. Tewari, M. Zollhöfer, H. Kim, P. Garrido, F. Bernard, P. Pérez, and C. Theobalt. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *International Conference on Computer Vision*, 2017.

[154] J. M. Tomczak and M. Welling. VAE with a VampPrior. In *International Conference on Artificial Intelligence and Statistics*, 2018.

[155] V. Veeriah, N. Zhuang, and G.-J. Qi. Differential recurrent neural networks for action recognition. In *International Conference on Computer Vision*, 2015.

[156] V. Vineet, J. Warrell, and P. Torr. Filter-Based Mean-Field Inference for Random Fields with Higher-Order Terms and Product Label-Spaces. *International Journal of Computer Vision*, 110(3):290–307, 2014.

[157] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM Transactions on Graphics*, 24(3):426–433, 2005.

[158] M. Wainwright and M. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, January 2008.

[159] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103(1):60–79, 2013.

[160] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[161] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[162] D. Weinland, M. Ozuysal, and P. Fua. Making Action Recognition Robust to Occlusions and Viewpoint Changes. In *European Conference on Computer Vision*, pages 635–648, 2010.

[163] F. Weissbach, A. Pelster, and B. Hamprecht. High-order variational perturbation theory for the free energy. *Physical Review E*, 66(3):036129, 2002.

[164] J. Winn and C. Bishop. Variational Message Passing. In *Journal of Machine Learning Research*, pages 661–694, 2005.

[165] C. Wu, D. Bradley, M. Gross, and T. Beeler. An anatomically-constrained local deformation model for monocular face capture. *ACM Transactions on Graphics*, 35(4):115:1–115:12, 2016.

[166] Z. Wu, A. Thangali, S. Sclaroff, and M. Betke. Coupling Detection and Data Association for Multiple Object Tracking. In *Conference on Computer Vision and Pattern Recognition*, 2012.

[167] S. Zhang, R. Benenson, and B. Schiele. Filtered feature channels for pedestrian detection. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[168] S. Zheng, S. Jayasumana, B. Romera-paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional Random Fields as Recurrent Neural Networks. In *International Conference on Computer Vision*, 2015.

Timur BAGAUTDINOV

PERSONAL DATA

DATE OF BIRTH:	31 August 1989
CITIZENSHIP:	Russian
ADDRESS:	Rue de la Paix 7, Renens, Switzerland
PHONE:	+41 789672969
EMAIL:	timur.bagautdinov@epfl.ch

PUBLICATIONS

T. Bagautdinov, C. Wu, J. Saragih, P. Fua, Y. Sheikh. **Modeling Facial Geometry using Compositional VAEs**. *Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, Utah, USA, 2018. (spotlight)*

T. Chavdarova, P. Baqué, S. Bouquet, A. Maksai, C. Jose, T. Bagautdinov, L. Lettry, P. Fua, L. Van Gool, F. Fleuret. **The WILDTRACK Multi-Camera Person Dataset**. *Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, Utah, USA, 2018.*

T. Bagautdinov, A. Alahi, F. Fleuret, P. Fua, S. Savarese, **Social Scene Understanding: End-to-End Multi-Person Action Localization and Collective Activity Recognition**. *Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii, USA, 2017. (oral)*

T. Bagautdinov*, P.Baque*, F. Fleuret and P. Fua. **Principled Parallel Mean-Field Inference for Discrete Random Fields**. *Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, Nevada, USA, 2016.*

T. Bagautdinov, F. Fleuret and P. Fua. **Probability Occupancy Maps for Occluded Depth Images**. *Conference on Computer Vision and Pattern Recognition (CVPR), Boston, Massachusetts, USA, 2015.*

Alewijnse, Bagautdinov, de Berg, Bouts, Brink, Buchin and Westenberg. **Progressive Geometric Algorithms**. *In Proc. 30th Symposium on Computational Geometry (SoCG) 2014.*

EDUCATION

2013 - 2018	PhD, Computer Vision **Ecole Polytechnique Federale de Lausanne**, Switzerland Topic: Variational Methods for Human Modeling Advisors: Pascal Fua, François Fleuret
SUMMER 2016	Visiting Researcher, Computer Vision **Stanford University**, USA Topic: Joint Multi-Person Detection and Activity Recognition Advisors: Silvio Savarese, Alexandre Alahi
2011 - 2013	Master of Science, Computer Science Engineering **Eindhoven University of Technology**, Netherlands Advisors: Bert de Vries, Mykola Pechenizkiy GPA: 9.1 / 10
2006 - 2010	Bachelor of Engineering, Software Engineering **Bauman Moscow State Technical University**, Russia Advisor: dr. Olga Peskova GPA: 4.75 / 5

Work Experience

May - Oct 2017	Research Intern at **Facebook Reality Labs**, Pittsburgh
2010 - 2011	Software engineer at **Yandex**, Moscow
2009 - 2010	Software engineer at **JSC Morinformsystem-Agat**, Moscow

Teaching

Spring 2018	TA, Deep Learning (EE-559), with François Fleuret
Autumn 2015	TA, Algorithms (CS-250), with Ola Svensson
Autumn 2014	TA, Pattern classification and ML (CS-433), with Emtiyaz Khan

Computer Skills

Programming Languages	C++, Python
Frameworks	TensorFlow, CUDA, OpenCV
Other	Emacs, git, LaTeX

Awards and Honors

Sep 2015	**TA Award** Ecole Polytechnique Federale de Lausanne
Aug 2013	**Cum Laude Diploma** Eindhoven University of Technology
2012 - 2013	**Honors Program Participant** Eindhoven University of Technology
Aug 2010	**First Class Honors Diploma** Bauman Moscow State Technical University

Languages

English	Fluent
Russian	Mothertongue
French	Beginner