LEARNING TO LOCALIZE LANDMARKS

BY

SAURABH SINGH

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2016

Urbana, Illinois

Abstract

The world is full of tiny but useful objects such as the door handle of a car or the light switch in a room. Such objects are barely visible in an image and can be well approximated by a single point. We refer to these small objects as *landmarks* in addition to the more common usage of the term to refer to the anatomical or facial landmarks. Landmark localization refers to the detection of one or more such landmarks in an image. In this dissertation, we describe methods for localizing such landmarks in images.

Automatically localizing these landmarks in images is hard as they usually don't have a distinctive appearance of their own. They are largely defined by their context. Absence of local appearance necessitates effective modeling of context to achieve good localization performance. This context can be explicit in the form of other landmarks that are spatially related or implicit in the form of certain recurring and informative patterns that may not have a semantic label. We describe methods that model both explicit and implicit context to improve landmark localization performance.

Localization performance is tied to the underlying learning machinery being used. Deep neural networks have proven to be quite successful for computer vision applications and this dissertation employs them as the underlying learning machine. We describe a method that uses a deep neural network with residual architecture, a recently proposed architecture for classification, and improves its performance with a novel stochastic training method called Swapout.

Acknowledgements

I am greatly thankful for the guidance, support and insights provided by my advisors David Forsyth and Derek Hoiem. They have provided a perfect mix of attention to low level details to high level thinking. Lessons learned from them have not only enabled this thesis but have also made me a better researcher. I would like to thank the members of my dissertation committee for their time and valuable suggestions.

I would like to thank my parents for instilling curiosity and encouraging risk taking. They have taught me to work hard and always improve, a life lesson that has helped me through difficult times. I am thankful to my brother, Apoorv, for his love and support. I am very thankful to my wonderful wife Harshitha for numerous discussions that have played an important role in bringing clarity to my ideas and materialization of some of the work described here. I am also thankful for the unwavering support, love and general happiness that she has brought into my life. These last few years have been specially wonderful with the arrival of our daughter Samhita. She has been a pure joy and has made this journey much more fun.

Finally, I would like to thank my lab mates and collaborators. I have learned quite a lot from them and our discussions have kept me abreast with last few years of rapid advances in the field.

Grants

This thesis is based upon the work supported in part by the National Science Foundation under Grants No. IIS 09-16014, IIS-1421521, and IIS-1029035, ONR MURI Award N00014-10-1-0934 and N00014-16-1-2007, and a Sloan Fellowship. We would also like to thank NVIDIA for donating some of the GPUs used in this work.

Dedicated to my family and my teachers . . . .

Table of Contents

Chapter 1: Introduction **1**
1.1 Localizing Landmarks 2
1.1.1 Localizing Multiple Landmarks 2
1.1.2 Localizing Little Landmarks 2
1.2 Discovering Latent Structure 3
1.3 Latent Landmarks 5
1.3.1 Nature of the Latent Landmarks 5
1.3.2 Structure of the Latent Landmarks 5
1.4 Thesis Statement 6
1.5 Contributions and Goals 6
1.5.1 Learning a Sequential Search for Landmarks 7
1.5.2 Learning to Localize Little Landmarks 7
1.5.3 Shared Latent Landmarks for Multiple Predictions 8
1.5.4 Swapout: A Stochastic Method for Training DNNs 8
1.5.5 Improved Landmark Localization with Swapout 9

Chapter 2: Learning a Sequential Search for Landmarks **10**
2.1 Background 13
2.2 Learning to Find Landmarks 15
2.2.1 Model, Inference and Training 16
2.3 Experiments 20
2.3.1 Features 21
2.3.2 Datasets 23
2.3.3 Comparison with the SOA Parsing 24
2.3.4 Ordering and Relations 26
2.3.5 Uniqueness of Detection Orderings 28
2.3.6 Finding Landmarks on Birds 30
2.4 Conclusion 31

Chapter 3: Learning to Localize Little Landmarks **32**
3.1 Background 35
3.2 Approach 37
3.2.1 Model and Inference 38
3.2.2 Prediction Scheme for a Step 39

3.2.3 The Estimate at a Location 40
3.2.4 Training 41
3.3 Experiments 44
3.3.1 Car Door Handle Dataset 45
3.3.2 Light Switch Dataset 46
3.3.3 Caltech UCSD Birds Dataset - Beaks 47
3.3.4 Leeds Sports Dataset - Left Wrist 48
3.4 Discussion 49
3.5 Conclusion 51

Chapter 4: Localizing Multiple Landmarks 52
4.1 Background 53
4.2 Approach 53
4.2.1 Visibility Prediction 54
4.3 Model and Training 54
4.4 Implementation Details 54
4.5 Experiments 57
4.5.1 Datasets and Evaluation Metric 57
4.5.2 Leeds Sports Dataset (LSP) 59
4.5.3 Caltech UCSD Birds Dataset (CUBS 200) 59
4.6 Conclusion 60

Chapter 5: Swapout - A Stochastic Method for Training DNNs 62
5.1 Background 64
5.2 Swapout 66
5.2.1 Inference in Stochastic Networks 69
5.2.2 Baseline Comparison Methods 70
5.3 Experiments 71
5.4 Discussion and Future Work 76

Chapter 6: Improved Landmark Localization with Swapout 79
6.1 Approach 80
6.1.1 Relation with Residual Architecture 80
6.2 Model 81
6.2.1 Training and Implementation Details 82
6.3 Experiments 82
6.3.1 Swapout 83
6.3.2 Comparison with State of the Art 84
6.3.3 Comparison with Methods in this Thesis 85
6.4 Conclusion 85

Chapter 7: Conclusions 87

Bibliography 88

Chapter 1

Introduction

Landmark localization is the task of localizing points of interest in a given image e.g. human body joints such wrists, elbows etc.. Depending on application, these landmarks are also referred to as keypoints, parts or joints in literature. While it is common for the landmarks to be semantically meaningful, they are not required to be so. Landmark localization has been studied in the context of various applications in the literature. One of the earliest application was localization of the facial landmarks, where the goal is to localize points that correspond to various facial features such as ends of lips, eyes, nostrils etc. Another well studied application is pose estimation, where the points correspond to various part or joint locations such as wrist, ankle, hip etc. Over time various new applications of landmark localization have cropped up, e.g. bird keypoint localization, Pascal keypoints, others.

Majority of the approaches for landmark localization focus on effective modeling of the contextual information. This information arises from spatial relations between the landmarks. These spatial relations are a form of structure that exists among landmarks and various approaches are designed to be effective at the discovery of this structure. In this thesis we formulate landmark localization in several ways that hypothesize and explore other forms of latent structures that may exist and are useful for landmark localization.

1.1 Localizing Landmarks

We consider landmark localization in two different scenarios. First is the more common setting of localizing multiple landmarks together, and the second is the localization of little landmarks in isolation.

1.1.1 Localizing Multiple Landmarks

In a typical landmark localization task, e.g. pose estimation, the goal is to localize a set of multiple target landmarks. These landmarks tend to be spatially related to each other occurring in consistent spatial configurations. Most of the approaches build upon this observations and are tailored to exploit the mutual context provided by these landmarks to each other. Various approaches model these spatial relations in different ways, e.g. tree model, star model etc (see discussion in section 2.1 of chapter 2. In this thesis we explore a variety of approaches for this more common setting of localizing multiple landmarks. We present a sequential search based scheme in chapter 2 that learns to find landmarks one after another in a image dependent order. In chapter 4 we describe a method that discovers a set of patterns which are predictive of the target landmarks. In chapter 6 we employ a more power learning technique, Swapout, to further improve the performance of landmark localization.

1.1.2 Localizing Little Landmarks

We use the term *little landmark* to refer to a landmark that is so small that it doesn't have much local appearance. Such landmarks typically represent very small objects or parts (Figure 1.1). To successfully localize such landmarks an approach has to rely on the effective use of contextual information that may be further away in the image. This contextual information can be supervised, in the form of other landmarks that are not

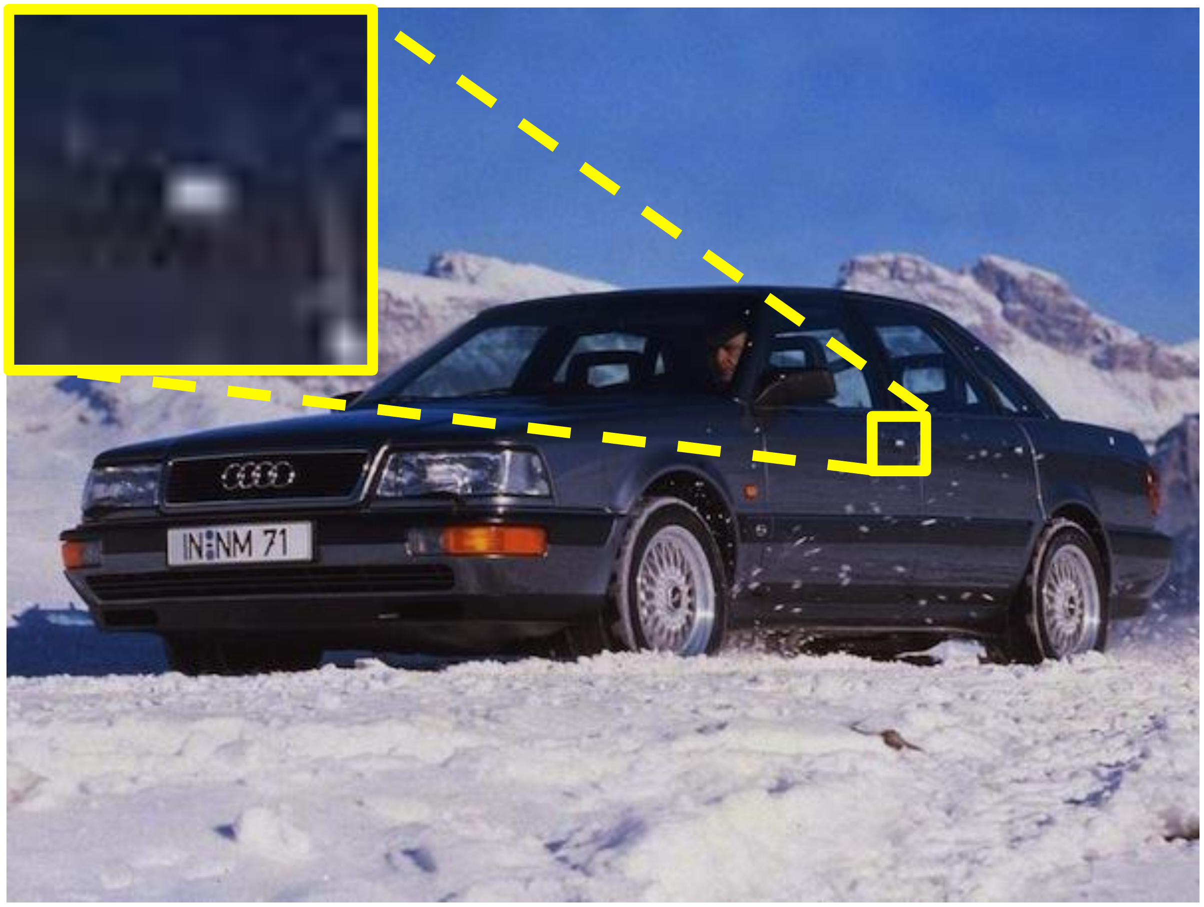


FIGURE 1.1: Several objects of interest are so tiny that they barely occupy few pixels (door handle above), yet we can easily localize them in images.

little landmarks, or unsupervised, in the form of spatially correlated patterns that do not have any semantics. In the most difficult version, there is only a single *little landmark* to be localized without any supervised contextual information. In this thesis we present an approach for this most difficult variant of the problem.

1.2 Discovering Latent Structure

Given a landmark localization task, such as localizing the door handle of a car in an image, how does one approach the problem in a way such that all the contextual information is appropriate accounted for? One way is to have a direct regressor that takes in the image and yields the regressed location of the target. Currently, a convolutional Deep Neural Network (DNN) based regressor would be the method of choice. However, the learned regressor is effectively a blackbox. It is difficult to interpret how the regression was done and in case of a failure, why the system failed. Further, a naive approach that looks at the whole image may lead to models that require a lot of parameters, easily overfit and need a lot of labeled data.

A successful approach has to effectively model the context and its relation to the target little landmark. Consider the front door handle of the car in Figure 1.1. It is barely visible but we can still find it. We are familiar with the structure of the car around it and use that as context to figure out its location. We can even explain our inference process as well if needed. Our search is mostly guided by the familiar elements visible nearby such as the wheels and the windows.

In this thesis, we present several approaches that make this contextual modeling more explicit. These approaches make assumptions about the form of underlying structure and are designed to discover and utilize that. In chapter 2 we describe an approach that casts landmark localization as a sequential search problem where the latent structure is the image specific order in which various landmarks should be detected. In chapter 3 we describe ***latent landmarks*** which model consistent patterns that are predictive in nature. They may be useful for predicting other latent landmarks or the target itself. Latent landmarks are inspired by the intuitive inference process discussed in previous paragraph. They are defined to be predictive and can potentially capture such informative patterns that can help solve the task.

1.3 Latent Landmarks

A latent landmark is a set of locations in an image, often nearby, that are predictive in nature. In the simplest case it is predictive of the target. However, it may be not be directly predictive of the target itself but of a set of latent landmarks. Those latent landmarks then might be predictive of the target or yet another set of latent landmarks and so on. Thus localizing the target requires finding the first set of latent landmarks and then using those to find the next set and so on until the target is found. Such an inference process requires the latent landmarks to posses several interesting properties as described below.

1.3.1 Nature of the Latent Landmarks

Latent landmarks are required to be predictive. They are either predictive of final task or of another set of latent landmarks. This requirement is on their function rather than on their appearance. Thus they may or may not be consistent in appearance. However, it does not matter as long as they are useful for the final task.

Note that the property of being predictive is very different from the property of being distinctive or discriminative. Distinctiveness is mostly appearance based and may have little to do with the usefulness for a task. Similarly, discriminativeness is often associated with appearance where it helps differentiate one set from from another (Singh, Gupta, and Efros, 2012; Doersch et al., 2012).

1.3.2 Structure of the Latent Landmarks

The sequential inference process causes some interesting properties to emerge. The first set of latent landmarks that are detected in such an inference not only need to be predictive but also distinctive. The next set of latent landmarks while being predictive need only be distinctive given the context of the previously detected landmarks and so on. Thus the first set of landmarks may be easy to find. The next set becomes easy to find once the first set has been localized. Thus, if the target is particularly hard to localize then these latent landmarks provide a way to automatically break the task into a sequence of subtasks where each subsequent one becomes easier once the earlier one has been completed.

1.4 Thesis Statement

We show that latent structures can be discovered and exploited to perform well on landmark localization tasks and describe a framework of ***latent landmarks*** *that can be used to design solutions that achieve this.* In this framework, a set of latent landmarks are first localized using a sequential inference scheme and these are then used to make the final prediction. These latent landmarks are learned in a task driven manner, i.e. they are tuned to the particular landmark localization task. Thus, although the learning framework is the same, they can be task specific. The main feature of our approach is that it takes the original prediction task and breaks it into a sequence of intermediate tasks. Thus, for a very hard prediction task, our approach can break it into a sequence of tasks of increasing difficulty where the solutions to the easier ones help solve the next in sequence. Each task in this sequence is formulated in terms of the latent landmarks. We demonstrate their effectiveness by applying them to several different tasks.

1.5 Contributions and Goals

This thesis describes several novel approaches for localizing landmarks in the two settings of: 1) Multiple landmarks, and 2) Little landmarks. Below we provide a brief summary of various chapters. Note that the work is presented in the chronological order in which it was performed. While this order corresponds to an increasing performance on some of the datasets, this trend can largely be attributed to the use of better and more powerful learning techniques as Deep Learning has advanced over time. It would be premature to assume that approaches described later are more powerful than those described earlier without a level comparison.

1.5.1 Learning a Sequential Search for Landmarks

We explore the problem of localizing a set of landmarks, such as the joints on the human body using a sequential inference process. Many landmarks, such as wrist and ankles, are difficult to find on their own while others, such as shoulders and head, are easier to find. However, the easier ones can help find the harder ones by providing useful contextual information. Thus, we formulate it as a sequential inference problem where we expect to find easier landmarks first and then use those to find the harder ones. Here the latent structure to be discovered is the order in which landmarks should be detected. We further allow this order to vary from image to image. We treat this as a problem where the locations of latent landmarks are known in training but their detection order is unknown. We show that our method can learn to detect these landmarks in an image dependent order and provides performance gains (Chapter 2).

1.5.2 Learning to Localize Little Landmarks

We explore the problem of localizing *little landmarks*, representing tiny objects or parts, that do not have a distinct local appearance. Automatically localizing little landmarks in images is hard, as they don't have a distinctive appearance of their own. These landmarks are largely defined by their context. They may occur in a specific configuration with respect to other patterns which are easier to find. We refer to such patterns as *latent landmarks*. We cast the localization as a sequential search problem. In each step of this sequence a landmark is detected. In intermediate steps latent landmarks are detected, providing context for the subsequent steps, while the final step detects the target. Note that the locations of latent landmarks are not known except the last one, which is the target landmark. Thus, our approach uses a sequential inference process and is supervised solely by the location of the target landmark. Assuming a fixed detection order we show that these latent landmarks can be discovered and make several interesting observations that guide the proposed work (Chapter 3).

1.5.3 Shared Latent Landmarks for Multiple Predictions

To address the more common scenario, we extend the framework of latent landmarks from localizing a single little landmark to localizing multiple landmarks. As before our method is sequential, however unlike before our method detects multiple latent landmarks in each step. These latent landmarks together provide the context for the next step. This scheme (Chapter 4) improves upon the performance of individual landmark prediction as above.

1.5.4 Swapout: A Stochastic Method for Training DNNs

Deep learning provides the underlying learning machinery for various computer vision applications. The performance of deep learning based models has significantly improved for the task of image classification. This is a result of various advances in the training algorithms as well as improved network architectures. The improvement in classification performance has also transferred to improved performance in various other computer vision applications. Notably, recently proposed residual network architecture has achieved impressive performance in image classification and has been successfully used in various other applications. We describe a stochastic training method that can be used to train an implicit ensemble of networks with a wide range of architectures including residual architectures. Our method improves upon the residual networks and achieves accuracies of a much deeper network using a shallower but wider architecture (Chapter 5).

1.5.5 Improved Landmark Localization with Swapout

With improved performance of the underlying learning machinery as discussed above, we expect the performance of landmark localization to improve as well. We apply the techniques described in chapter 5 to train a powerful model for localizing multiple landmarks. This model is trained with swapout and achieves state of the art accuracies for localizing landmarks on the Leeds Sports Dataset (Chapter 6).

Chapter 2

Learning a Sequential Search for Landmarks

Identifying sets of landmarks, such as joints on the human body or parts of a car, is a major problem in computer vision. Many parts, such as a wrist or car door handle, are difficult to find on their own, so the key research problem is modeling the relations among the appearances and positions of landmarks. Relations among landmarks may be very complicated; for example, human bodies are posed and appareled in structured but complex ways, so that the position and appearance of a wrist depends on the positions of shoulders, elbows, hips, presence of long sleeves, and so on.

To make learning and inference tractable, existing approaches are forced to use at least some of a menu of assumptions: that each landmark can be identified relatively easily; that appearance and spatial terms factorize; that spatial relations fit a convenient model; that discriminative methods can satisfactorily handle relational information without expressing it explicitly; or that intractable inference problems can be dealt with approximately in a satisfactory way. The result is a surprisingly small list of strategies for finding landmarks (section 2.1).

This chapter describes a novel, alternative strategy for finding landmarks. Instead of fixing a model structure and then dealing with an intractable inference, we treat the

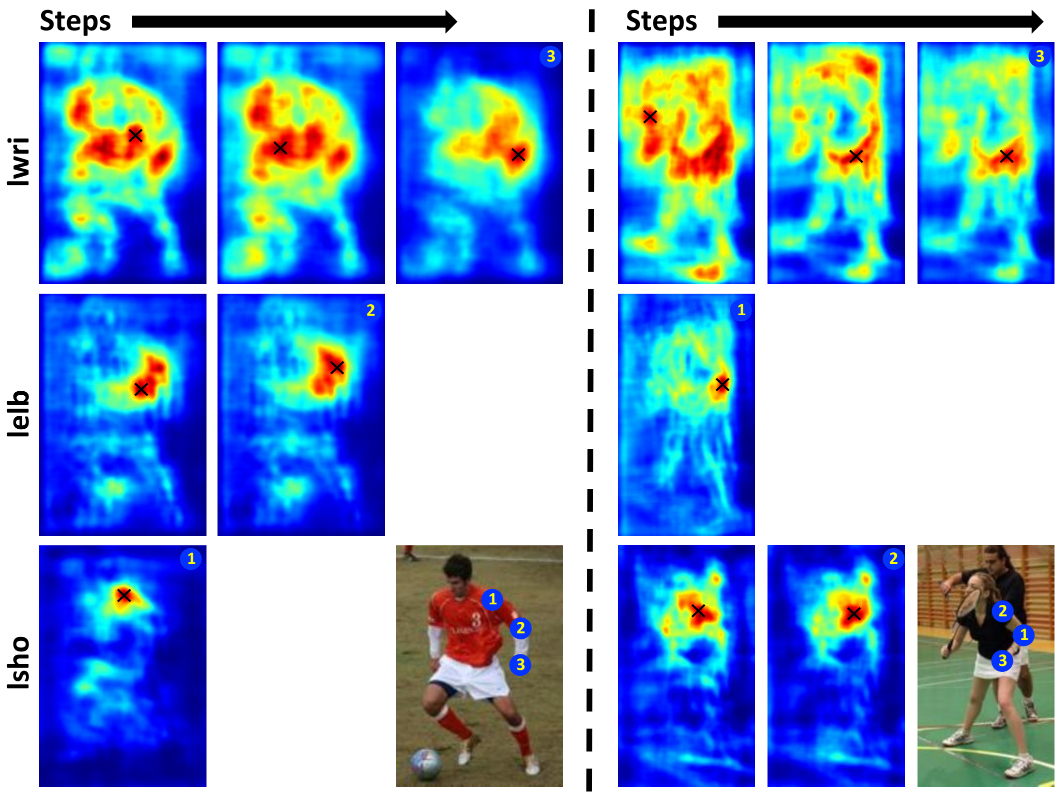


FIGURE 2.1: A visualization of the implicit spatial model learned by the method for the case of three landmarks in two different images. Each column corresponds to a *step* of the method and displays the scores for every location in the image, for each remaining landmark, as a heat map. In bottom- right we show the inferred locations numbered by the step in which they were detected. Note that the landmarks are detected in a different order in the two images. The peaks, marked with a black cross, shift to the correct locations as steps progress; e.g., peak for *lelb* in left image shifts to the correct location in step 2 after lsho is detected in step 1. Similarly, peak for *lwri* shifts in step 3 once *lelb* is detected.

inference as a sequential search procedure and learn parameters such that the search remains tractable. In every step of the search a landmark is detected and our model uses the detected landmarks to capture increasingly complex appearance and spatial relations jointly (Figure 2.1).

We assume that in each image at least one landmark can be detected with high accuracy without additional context (though which one is easy to detect might change from image to image). Our method automatically learns to find the easy landmark, and uses that

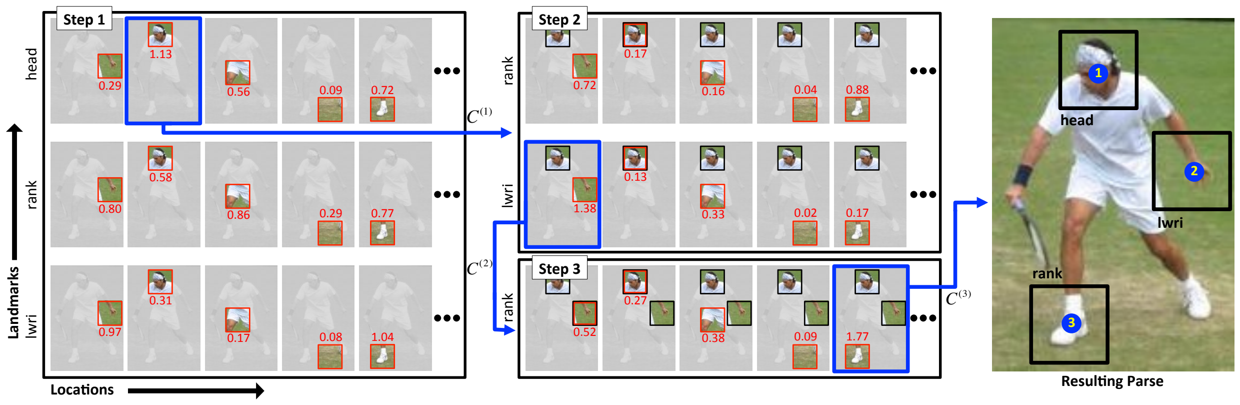


FIGURE 2.2: Visualization of the inference procedure using our learned function. In the first step, all locations in an image are evaluated as candidates for all landmarks. Highest scoring of these yields a landmark detection (blue box in left column). In the next step, all the locations are evaluated jointly with the first landmark to yield the next detection and so on until all landmarks are detected.

landmark to provide context to support an image description and so find the next landmark; it then uses those two landmarks to find the third, and so on. In particular, the sequence of landmarks found may differ from image to image. Our system *learns* how each landmark depends on what has already been found, and *learns* to identify which landmark to find next. Because our method does not require any expert guidance for spatial dependencies or which landmark to detect first, it can be applied easily to any landmark detection problem. We demonstrate this by detecting landmarks on people and birds.

Our approach learns a sequential search for finding landmarks in an image- dependent order. It is: 1) simple, easy to implement and reproduce, and computationally efficient; 2) general, as it makes no assumptions about how landmarks relate to each other making it applicable to any landmark localization task (we apply it *as is* on humans as well as birds); 3) able to model appearance and locations of very high order cliques of landmarks jointly.

2.1 Background

Landmark detection is a well-studied problem, usually in the domain of finding human body joints or parts Yang and Ramanan, 2013; Wang, Tran, and Liao, 2011; Tran and Forsyth, 2010; Eichner and Ferrari, 2009; Sapp, Toshev, and Taskar, 2010; Andriluka, Roth, and Schiele, 2009; Dantone et al., 2013; Toshev and Szegedy, 2013. Our method applies to humans but is designed to apply equally well to other object landmark-finding problems.

Modeling Landmark Dependencies: The modeling options currently available are:

- *Bag*: where one ignores relations (e.g. Csurka et al., 2004).
- *Full Relational*: model all pairwise relations leading to intractable inference (e.g. Leung, Burl, and Perona, 1995; Fergus, Perona, and Zisserman, 2003; Fei-Fei, Fergus, and Perona, 2006; Wang and Mori, 2008; Tran and Forsyth, 2010). One solution is to reduce the search space, i.e., by segmentation Mori et al., 2004, local searches Tran and Forsyth, 2010; Ferrari, Marín-Jiménez, and Zisserman, 2008 or cascades Felzenszwalb, Girshick, and McAllester, 2010; Sapp, Toshev, and Taskar, 2010.
- *Star*: model positions relative to a root (e.g. Leibe, Seemann, and Schiele, 2005; Crandall, Felzenswalb, and Huttenlocher, 2005; Fergus, Perona, and Zisserman, 2005; Bourdev and Malik, 2009.
- *Tree*: model a subtree of the graph of relations, possibly conditioned on the image (most active for the case of landmark location on humans, e.g. Yang and Ramanan, 2013; Ioffe and Forsyth, 2001; Ramanan, 2006; Felzenszwalb and Huttenlocher, 2005; Andriluka, Roth, and Schiele, 2009; the best current human parser has this form Chen and Yuille, 2014).

- *K-fan*: a variant of tree models, where the relations preserved form a junction tree of tolerable size (e.g. Crandall, Felzenswalb, and Huttenlocher, 2005; a variety of comparable approaches are reviewed in "Part-Based Category Models").
- *Implicit*: model relations implicitly, for example with auto-context Ramakrishna et al., 2014; Tu, 2008.

We offer an alternative: model relations as an ordered search procedure. Our model assumes that $k-1$ landmarks and a local detector can be used together to determine which of the remaining landmarks should be located on a per-image basis. After locating the head, shoulder, and elbow, the wrist may be the best landmark to locate next in one image, while the hip is best in another. Thus, we substitute sequential inference for joint inference in order to benefit from more expressive dependency models.

Modeling Landmark Appearance: There is a rich set of options for modeling image appearance; space does not allow a comprehensive review. We use sums of rectangles offset from landmark centers, which performs reasonably well, but recent advances using feature learning Chen and Yuille, 2014; Tompson et al., 2014; Toshev and Szegedy, 2013 might improve results.

Learning to Search: Our approach to learn which landmark to find first in an image has similarities with Q learning Barto, 1998, though, we do not learn an explicit value function. The view of inference as a search procedure has been taken in several existing works Daumé III and Marcu, 2005; Ratliff, Silver, and Bagnell, 2009. Ratliff *et al.* Ratliff, Silver, and Bagnell, 2009 study this in the context of imitation learning to learn non-linear cost functions. Daume *et al.* Daumé III and Marcu, 2005 treat learning as a parameterized search optimization. Our method uses a computationally efficient greedy search but learns complex intermediate representations and a non-linear scoring function.

2.2 Learning to Find Landmarks

We want our method to be general and reasonably efficient in inference. Therefore, we avoid any prior knowledge about the relations between the landmarks such as upper/lower body, arms/legs and treat all landmarks similarly. For efficiency, we choose a greedy inference procedure but optimize our model specifically for such inference.

We make the following assumptions about the problem structure: (1) In an image, some landmarks may need more contextual information than others to be accurately detected yielding a loose ordering based on required context, e.g. an occluded wrist. Typically, this information is the location and appearance of other landmarks, (2) A subset of landmarks can be detected with fairly high accuracy without additional context. These can then be used to provide context for the next set of landmarks. Note that these assumptions are quite general and true for most practical applications.

Above assumptions lead naturally to our approach. It learns to detect landmarks one by one in the increasing order of contextual information required. It first automatically learns to find landmarks that do not need much contextual information relying on (2). Then, it uses the set of detected landmarks to provide the context for the next landmark based on (1). As more landmarks are detected the available context becomes richer facilitating the detection of harder landmarks. Our approach doesn't assume that some fixed set of landmarks is always easy; it allows these to be different from image to image. It uses no additional supervision about their easiness or detection order. Further, it doesn't impose an explicit spatial model nor does it treat one landmark differently from other. This information is coded in the features (section 2.3.1) of the landmarks and it learns to use them as needed. Thus, *our approach learns not just to score correct landmark locations but also the image dependent order in which to find them*.

2.2.1 Model, Inference and Training

Let c be the image location of a landmark which can be either a point in the image or $\emptyset$, indicating *unknown*. The current state of our inference is an ordered set of P locations, $C = (c_1, \ldots, c_P)$, one for each target landmark. At each step our algorithm can either replace a $\emptyset$ with a location or stop.

Let $x^T = \Phi(C)^T = [\phi(c_1)^T \ldots \phi(c_P)^T]$ be a feature vector corresponding to C formed by concatenating individual feature vectors for c_i. We define $\phi(\emptyset) = \mathbf{0}$ and use the same ϕ for each landmark. Our algorithm chooses whether to replace $\emptyset$ or stop at current state by evaluating a learned scoring function **F** for each possible next state as follows:

$$\mathbf{F}(x) = \mathbf{F}(\Phi(C)) \tag{2.1}$$

Inference: We use a greedy search strategy (Figure 2.2). Let $C^{(s)}$ be the state at some step $s \in \{0, \ldots, P\}$. Let $C^{(s)} \oplus c_{ij}$ be a candidate next state obtained by replacing the i-th unknown landmark with the j-th image location. Then we have:

$$C^{(s+1)} = \arg\max_{i,j} \quad \mathbf{F}(C^{(s)} \oplus c_{ij}) \tag{2.2}$$

Note that at step s, exactly $P - s$ elements of C are $\emptyset$. If the image contains N locations then **F** would evaluate $(P - s) \times N$ possible next states.

This clearly places significant demands on **F**, which much take a large value for good groups of locations and a small value for the bad groups. For example, we require that **F** be large for good elbow groups, and good shoulder-elbow groups, and good shoulder-elbow-wrist groups. We use a 5 layer fully connected neural network as the learner to model **F** (Figure 2.3). All the activation functions are rectified linearities except the output which is linear.

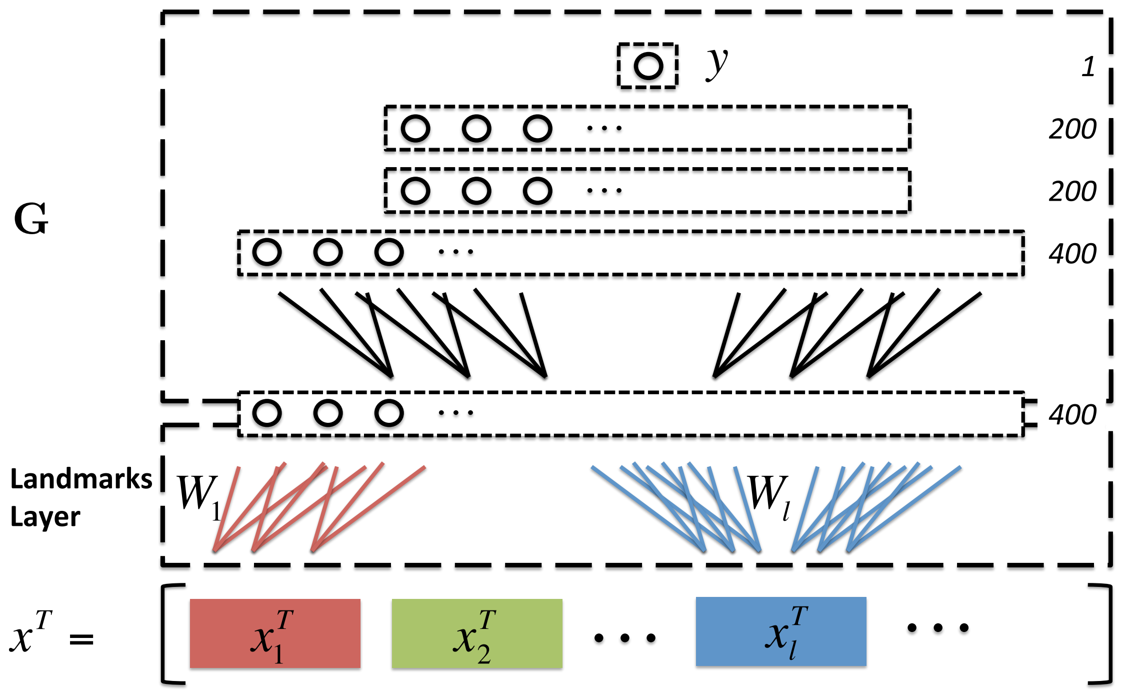


FIGURE 2.3: Structure of the neural network learner used to model the scoring function. We refer to the first hidden layer as *landmarks layer* (see text) with colored blocks corresponding to different target landmarks. The rest of the model acts as a scoring function that learns to score landmark groups of various sizes. Our model is fully connected; numbers on the right show the layer sizes.

There are several interesting properties of this inference scheme. Firstly, it doesn't impose an ordering over landmarks. Thus, they can be detected in one order for one image and in a different order for another image. Figure 2.7 visualizes the frequencies of detections of a subset of landmarks in each step. While elbows and wrists have a preference to be detected later, shoulders and ankles tend to be detected in earlier steps. Figure 2.9 shows several images in which the inference followed different order. Secondly, the next landmark is scored in conjunction with already detected landmarks exploiting the context provided by them.

Training: We train our model in an online fashion, so we consider the loss due to a single image **I**. Let $C^{(s)} = C^{(s-1)} \oplus c_{ij}$ be the state in step s reached by selecting the image location j for landmark i. Similarly, let $C_*^{(s)} = C^{(s-1)} \oplus c_{kl}^*$ be the target ground truth state reachable by selecting the image location l for landmark k. We explain ground truth selection in the next section. Our training loss $\mathbf{J}$ for predicting P landmarks is an average of individual losses J_s for each landmark as follows:

$$\mathbf{J}(\mathbf{I}; W) = \frac{1}{P} \sum_{s=1}^{P} J_s(C^{(s)}, C_*^{(s)}) \tag{2.3}$$

We would like to train our model $\mathbf{F}$ such that it scores $C_*^{(s)}$ higher than any other $C^{(s)}$. Additionally, it should penalize predictions which are further away from ground truth more than those which are closer. We use a margin based structured loss J_s which uses a candidate dependent margin function $\Delta_{ij,kl}$ to achieve this. Let $x = \Phi(C^{(s)})$ and $x_* = \Phi(C_*^{(s)})$ be the features for $C^{(s)}$ and $C_*^{(s)}$ respectively. Further, let d_{ij}^* be the distance of image location j from the ground truth location of landmark i in image space. Then the loss J_s is defined as follows:

$$J_s(C^{(s)}, C_*^{(s)}) = \max(0, \Delta_{ij,kl} + \mathbf{F}(x) - \mathbf{F}(x_*)) \tag{2.4}$$

$$\Delta_{ij,kl} = \min(\alpha \min(d_{ij}^*, d_{kj}^*), 1) \tag{2.5}$$

We use a scaling constant, $\alpha < 1$, to control the steepness of margin function. Note how the margin depends on both the target ground truth landmark for the current step as well as the ground truth landmark of the candidate that was selected. Consider a candidate for a landmark i which lies exactly on its ground truth and the ground truth landmark for the current step k where $i \neq k$. $\Delta_{ij,kl}$ as defined above ensures that such a candidate has a margin of zero w.r.t. k.

Ground Truth Selection: For each training image we have the ground truth landmark locations but we do not have the ground truth ordering $C_*^{(s)}$ in which they should be detected. Our training algorithm dynamically selects $C_*^{(s)}$ by considering the highest scoring candidate for each of the remaining landmarks in step s and picking the one which is closest to its ground truth. This scheme favors the detection of those landmarks in early steps which can be learned easily, i.e., whose predictions tend to fall closer to their ground truths. Further, it enables the learning of an image dependent ordering. We also tried using the ground truth landmark whose candidate was scored highest in the current step. This strategy is very sensitive to initialization and prone to getting stuck in bad local minima.

Practical Considerations: We train our model through back-propagation using stochastic gradient descent with momentum. Although the updates are straightforward for all the layers, we observed an optimization instability. Consider the *landmarks layer* (Figure 2.3) and let $\{W_L, b_L\}$ with $W_L = [W_1 \dots W_P]$ be the parameters of this layer. We have $x^T = [x_1^T \dots x_P^T]^T = \Phi(C^{(s)})$ with $x_l = \phi(c_l)$ as the features of the individual elements in $C^{(s)}$. We re-write our scoring function making the parameters of this layer explicit.

$$\mathbf{F}(x) = \mathbf{G}(W_L x + b_L) = \mathbf{G}(\sum_{l=1}^{P} W_l x_l + b_L) \tag{2.6}$$

Thus, W_l is the block of W_L corresponding to the landmark l. If s was the step in which landmark l was detected then the gradient of the objective w.r.t. W_l is

$$\frac{\partial \mathbf{J}}{\partial W_l} = \frac{1}{P}\left(\frac{\partial J_s}{\partial W_l} + \lambda \sum_{i=s+1}^{P} \frac{\partial J_i}{\partial W_l}\right) \tag{2.7}$$

Earlier terms for $s \in \{1, \dots, s-1\}$ are zeros since x_l is zero in those terms. We introduce a multiplier λ for the later terms. Setting $\lambda = 1$ yields the original gradient. In its original form, different W_l receive gradients of different magnitudes depending on the step in which landmark l was detected. This introduces instability during optimization and hurts the performance of landmarks that are detected earlier. We use λ as a normalizer to counter this effect and set its value to $\frac{0.5}{(P-s)}$ in experiments. We experimented with setting $\lambda = 0$

Method	Torso	Up. Leg	Lo. Leg	Up. Arm	Forearm	Head	Total
Yang and Ramanan, 2013	84.1	69.5	65.6	52.5	35.9	77.1	60.8
Eichner and Ferrari, 2012	86.2	74.3	69.3	56.5	37.4	80.1	64.3
Kiefel and Gehler, 2014	84.3	74.5	67.6	54.1	28.3	78.3	61.2
Pishchulin et al., 2013a	87.4	75.7	68.0	54.4	33.7	77.4	62.8
Ramakrishna et al., 2014	88.1	79.0	73.6	62.8	39.5	80.4	67.8
Ouyang, Chu, and Wang, 2014	88.6	77.8	71.9	61.9	45.4	84.3	68.7
Pishchulin et al., 2013b	88.7	78.9	73.2	61.8	45.0	85.1	69.2
Chen and Yuille, 2014	92.7	82.9	77.0	69.2	55.4	87.8	75.0
Ours	88.0	77.2	72.7	58.2	34.0	79.9	65.2

TABLE 2.1: Comparison of our approach with state of the art on PCP@0.5. We perform similar to several recent approaches as is evident in Figure 2.6 where our method is close to state of the art in the high precision area.

and found that it is better behaved when compared to using the original gradients but yields slightly inferior final performance when compared to the suggested setting. We set α (eq. 2.5) in a dataset dependent way such that on an average the margin for a landmark is close to one near other landmarks.

Features are centered and scaled using the mean and range computed over the training set for each dataset. For each image, we consider locations in a grid with a stride of 5 pixels during training and 2 pixels during testing for computational efficiency. We augment the data by adding left-right flips, random crops and small scalings of the images. Models are trained on an NVIDIA K40 to further speed up the training.

2.3 Experiments

We evaluate our approach on two different landmark prediction tasks: 1) Humans and, 2) Birds. We apply our model *as is* for both the tasks and use the same set of generic features for both. We first describe the features used and then evaluate our approach on several established datasets.

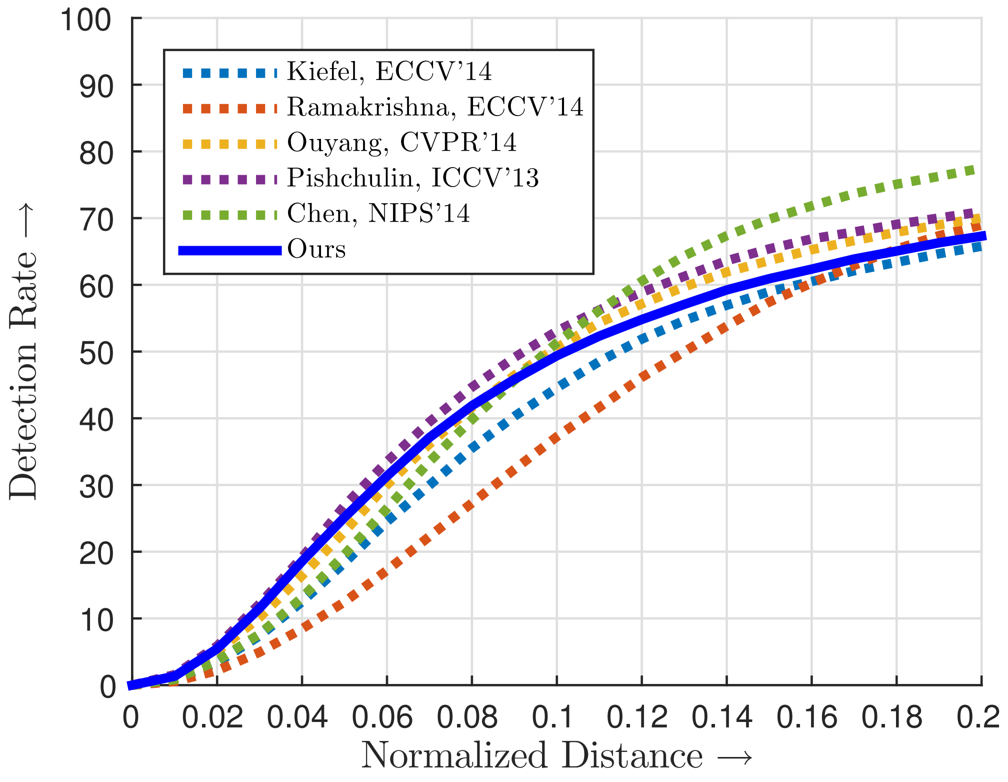


FIGURE 2.4: Comparison of our method with state of the art for human landmark detection on the Leeds Sports dataset. We are comparable with the leading approaches in the high precision area of small distances despite using a very simple set of features and no explicit imposed structure.

2.3.1 Features

We use a simple set of features which allow the model to capture appearance and spatial relations between landmarks.

Appearance Features: Each appearance feature for a given location is the average of values in a box of random size, at a random offset, from a randomly chosen image channel. This allows the model to capture relations in appearance between nearby landmarks. We convert the image into 10 channels consisting of 3 Luv channels, 6 gradient orientation channels and 1 gradient magnitude channel Dollár et al., 2009. Boxes are constructed by sampling the square root of their areas from the range of $[\sqrt{5}, \sqrt{1000}]$ pixels, their log aspect ratio from the range of $[\log(1/5), \log 5]$ and then solving for an integer width and

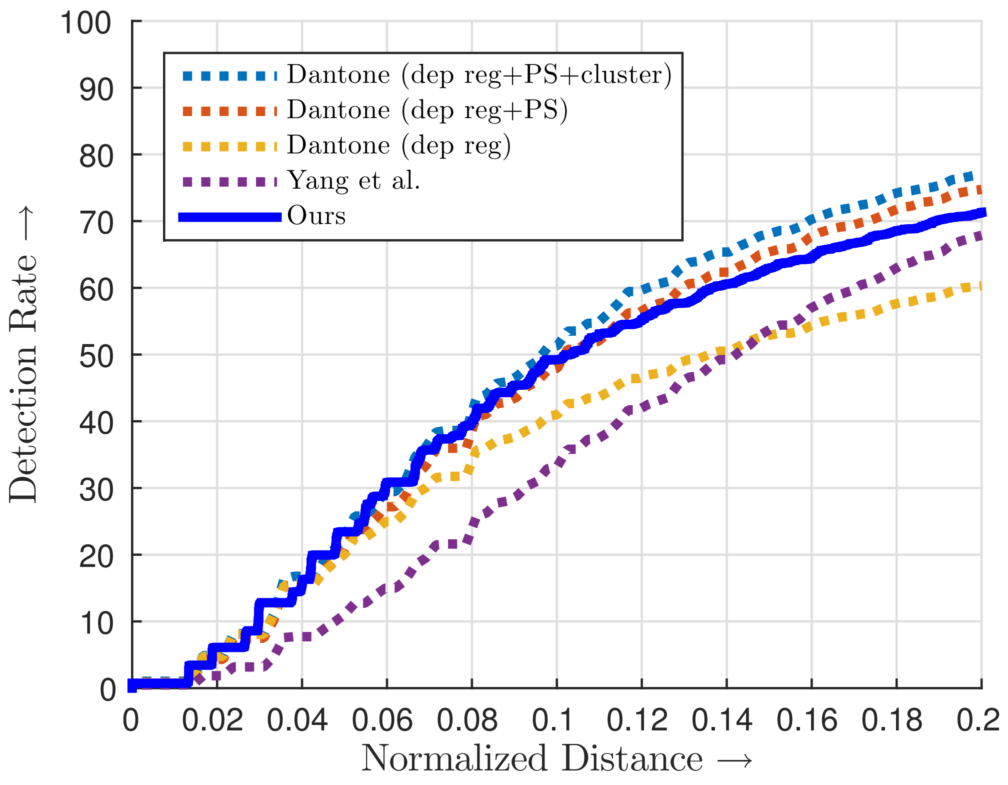


FIGURE 2.5: On the Fashion Pose dataset our performance is similar to the approach of Dantone et al., 2013 and better than Yang and Ramanan, 2013

height. This sampling of areas is biased towards generating smaller boxes which we found to perform better. Offsets are sampled randomly within a circle of radius 50. Biasing the sampling towards smaller offsets performed better than uniform sampling.

Location Features: Location is encoded in two ways. First, directly normalized (x, y) yield two features. Normalization is done by assuming the origin as the center of the image and dividing by the maximum of width and height of images, m, across the whole dataset. Next, location is encoded in terms of some fixed points. For any image, we set down 20×20 fixed points at equal intervals in range $[-m/2, m/2]$ from the center along both axis. Position is then encoded as a 400 dimensional vector capturing proximity to these points. The proximity is computed by $1 - \frac{min(d,r)}{r}$, where d is the distance to the point and r is a dataset dependent constant computed as $0.15 \times m$ to ensure some points

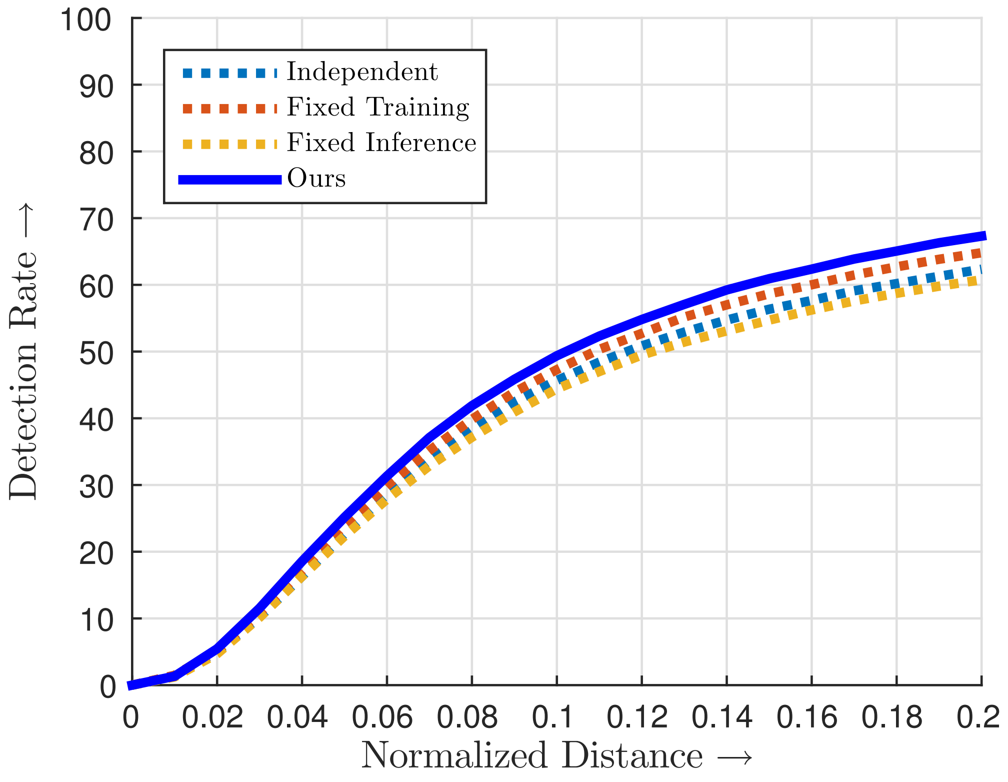


FIGURE 2.6: Our method perform better than several variants. Independent trains individual landmark detectors with no interaction. Fixed Training follows a predetermined landmark detection order and thus uses more context than Independent, performing better. Fixed Inference uses our model but follows a fixed order of detection. Our approach utilizes the flexibility of image dependent ordering to outperform all.

are always in proximity. We found that these features improve the recall for larger error thresholds.

2.3.2 Datasets

We demonstrate the performance of our approach for human landmarks on the Leeds Sports Dataset (LSP) Johnson and Everingham, 2010 and the Fashion Pose dataset Dantone et al., 2013. LSP contains 1000 training and 1000 testing images of humans in difficult articulated poses with 14 landmarks. We use the Observer Centric (OC) annotations Eichner and Ferrari, 2012. Fashion Pose Dataset contains 6530 training and 775 testing images of human

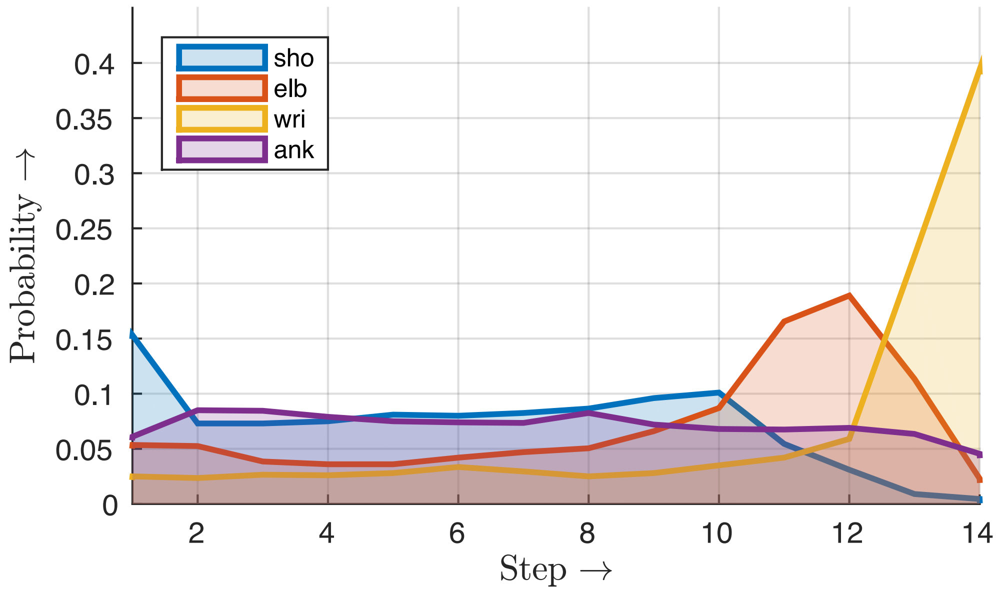


FIGURE 2.7: We visualize the frequency of detection of a few landmarks in all the steps. Elbows tend to be detected later while shoulders tend to be detected earlier. However, all have a non-zero frequency at each step.

models with 13 landmarks. This dataset has less pose variation but significant appearance variation due to clothing. Further, to demonstrate the general applicability of our approach we apply it to the Caltech-UCSD Birds 200 (2011) dataset (CUBS 200) Welinder et al., 2010. It contains 5994 training and 5794 testing images with 15 landmarks for birds. For all the datasets we work with provided train and test splits.

Evaluation Metric: We adopt the generally accepted metric of plotting detection rate against normalized distance from ground truth. Normalization is done using the torso height. PCP hides improvements in high precision region, but we report it on LSP to compare with existing work.

2.3.3 Comparison with the SOA Parsing

Figure 2.4 and 2.5 show that our method compares favorably with the state of the art on both the LSP and the Fashion Pose datasets. Note that our performance is close to

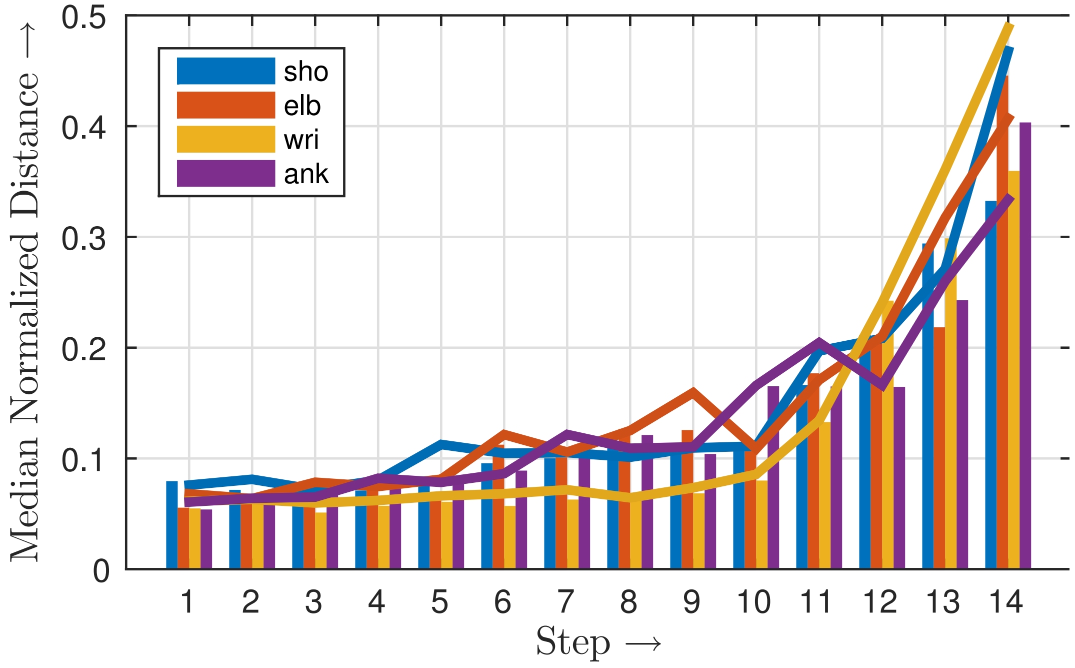


FIGURE 2.8: Median errors at various steps. The bars correspond to errors from our model while the lines correspond to errors from the *Independent*baseline. Every landmark had a minimum of 33 detections at any step. From the bars it is clear that, independent of landmarks, earlier steps tend to have lower error indicating that our model has learned to detect them in a meaningful image dependent order. For each step, error for *Independent* for a certain landmark is computed by first selecting those images in which our method detected that landmark in that step and reporting the median error of *Independent* over those. There is high correlation between cases that our method finds hard (detects in later steps) and cases that the independent detector finds hard. Thus, our method is able to pick out landmarks which are more likely to be correct and is detecting them early yielding reliable context for the later steps. Also note that there is larger gain in later steps where the context would play a larger role.

the best performing approaches in the high precision area. Figure 2.9 and 2.10 show qualitative results on LSP and Fashion Pose for various error quartiles. Table 2.1 compares our method with a few more approaches on LSP using PCP@0.5. Our performance is close to several recent approaches, though, other methods such as Chen *et al.* Chen and Yuille, 2014 outperform with a looser criterion (Figure 2.4). However, note that in contrast to other state of the art body parsers, our method can be applied without modification to parse bird landmarks too (section 2.3.6).

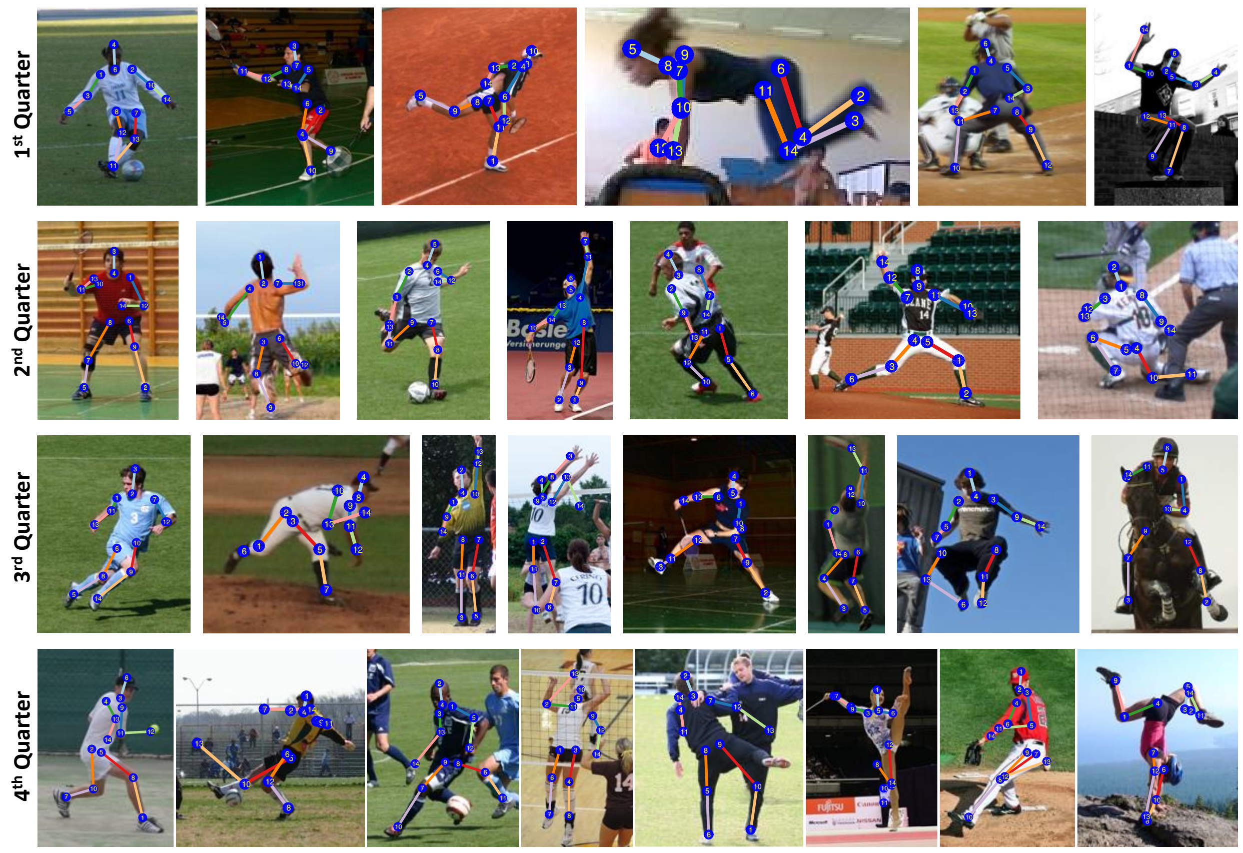


FIGURE 2.9: Qualitative Results for different error percentiles of Leeds Sports dataset: We sort the test images by sum of squared errors of our predictions from the ground truth. Each rows shows results on atypical poses from different quarters of this ordering with first row being most accurate. Numbers in blue circles show the step in which each landmark was detected. We also display the limbs as a visual aid. Different images exhibit different detection order. From top to bottom, as poses get harder, wrists quickly lose precision. Other failure modes include confusion with clutter, confusion with other people and under-represented poses such as inverted people.

2.3.4 Ordering and Relations

Figure 2.1 visualizes the implicit spatial model learned by our method in two images. It is clear that the landmarks are detected in different orders and the ones detected earlier help localize the harder ones as shown by the shift in peaks. Our opportunistic ordering strategy is able to use appearance and spatial relations to improve performance.

In Figure 2.6, we compare our method with several variants and show that it improves upon them. *Independent* trains all the landmark detectors independently. *Fixed Training*

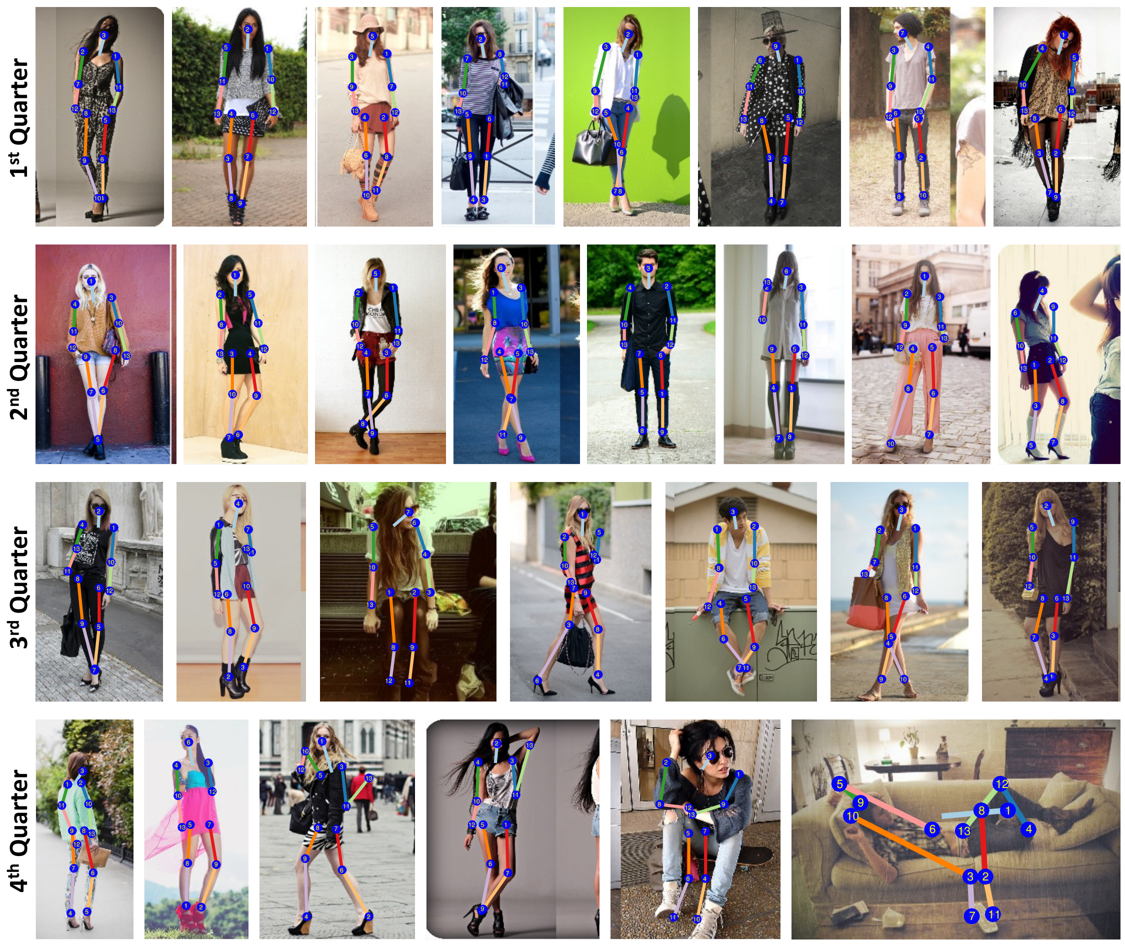


FIGURE 2.10: Qualitative Results for different error percentiles of Fashion Pose dataset: As in figure 2.9, test images are sorted by the sum of squared errors of the predictions from the ground truth. Each rows shows results on different quarters of this ordering with first row being most accurate. Primary failure mode is atypical poses (bottom right).

trains the model using a fixed ordering of $head \rightarrow shoulders \rightarrow elbows \rightarrow wrists \rightarrow hips \rightarrow knees \rightarrow ankles$. Note that this ordering deliberately puts harder landmarks like $wrists$ and $ankles$ after $shoulders$ and $hips$ as they may be needed as context. *Fixed Inference* uses our trained model but follows the aforementioned fixed ordering during inference. *Fixed Training* does better than *Independent* by exploiting the context provided by earlier landmarks. Our method improves upon it by allowing a flexible choice of ordering and learning a richer dependency model. *Fixed Inference* does the worst as a fixed ordering

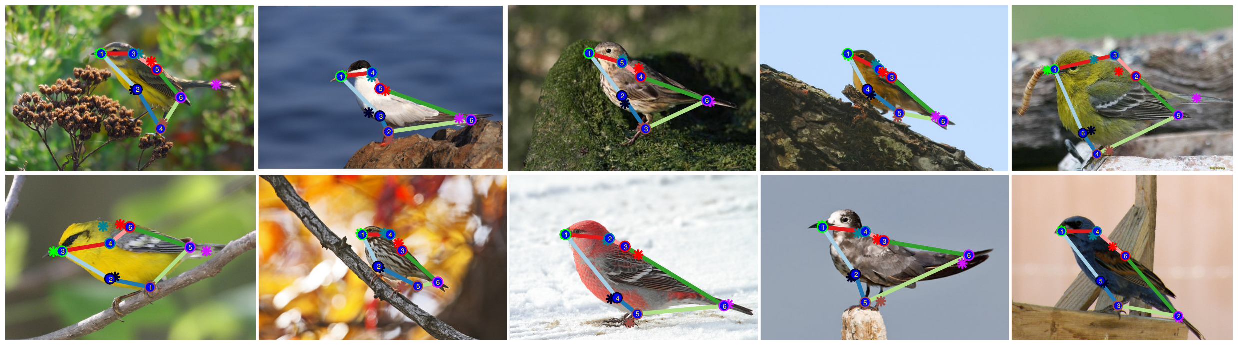


FIGURE 2.11: Qualitative examples of landmark detections on CUBS 200 birds dataset. Although we detect all the landmarks, we visualize a subset, {*beak, belly, leftleg, tail, back, nape*}, to avoid clutter. Asterisks show their ground truth locations, blue circles with colored borders show the corresponding detections and the numbers indicate the step in which they were detected. Lines connecting detections are shown as a visual aid and are color coded to denote same pair of landmarks. Our approach does quite well in comparison to a baseline of independent detectors (See Figure 2.12). Note the difference in order in which the landmarks are detected from image to image.

forces it into a part of search space for which it has not learned.

Figure 2.7 shows that the learned model is indeed using an image dependent ordering. It plots the frequency of a few landmarks being detected in various steps. Although harder landmarks such as wrists and elbows tend to be detected later, there is a significant spread over the steps. Other landmarks (not visualized) show a trend similar to that of ankle.

Figure 2.8 shows that landmarks which are detected in earlier steps tend to have less error. We plot median normalized distance of the predictions for various landmarks at each of the steps. Irrespective of the landmark type, landmarks which are detected earlier tend to have lower error. This is expected since landmarks that are hard for an image will be detected in later steps.

2.3.5 Uniqueness of Detection Orderings

We counted the number of unique orderings of landmark detections in test images and found that our method follows a unique ordering for each image in both the Leeds

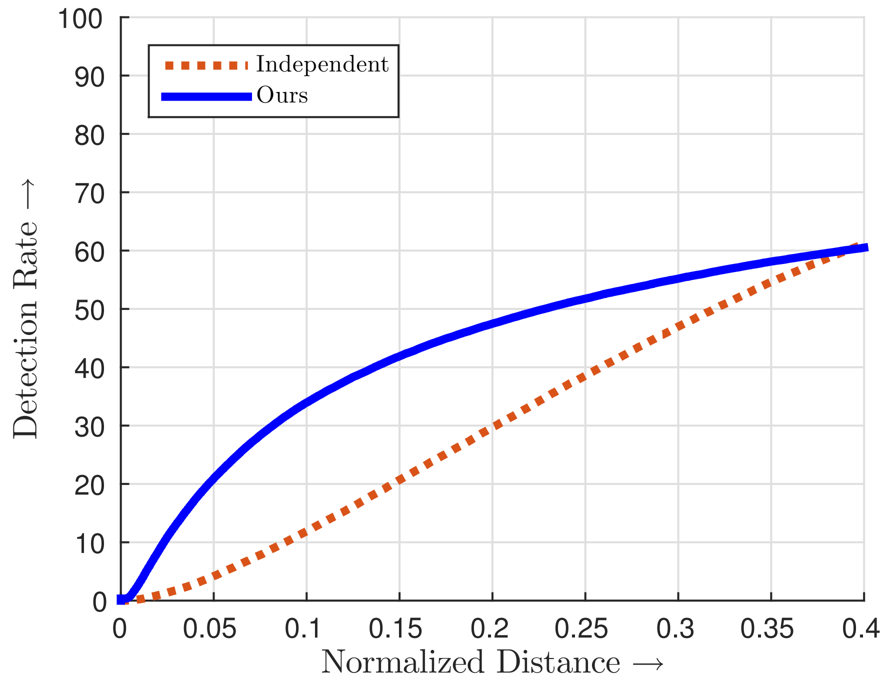


FIGURE 2.12: Comparison of our approach on CUBS 200 dataset with a baseline of independent detectors. Our method is able to exploit the context yielding significant improvement. Normalized distance is computed by dividing the pixel distance by the height of the bird.

Ordering	% Occurrences
{left-shoulder, left-elbow, left-wrist}	62.3
{left-shoulder, left-wrist, left-elbow}	11.6
{left-elbow, left-shoulder, left-wrist}	9.6
{left-wrist, left-shoulder, left-elbow}	7.9
{left-wrist, left-elbow, left-shoulder}	5.3
{left-elbow, left-wrist, left-shoulder}	3.3

TABLE 2.2: Percent occurrences of detection orderings in LSP for a restricted set of landmarks, namely *{left-shoulder, left-elbow, left-wrist}*. Clearly first order is preferred but other orderings are common as well.

Sports Dataset (1000 test images) and the Fashion Pose Dataset (775 test images). This, in conjunction with Figure 2.8, indicates that an image dependent ordering is indeed useful.

However, a random model may also yield unique orderings. To study this, we consider only the *{left-shoulder, left-elbow, left-wrist}* landmarks and show the percentage occurrences of various orderings for LSP in Table 2.2. Clearly, the intuitive ordering of *{left-shoulder, left-elbow, left-wrist}* is preferred, but other orderings are common as well. A χ^2 test rejects the null hypothesis of a uniform distribution with a significance value of zero, clearly showing that the orderings are not uniformly random.

2.3.6 Finding Landmarks on Birds

Landmark finding is a general problem, and our approach is especially well suited for a general application as it doesn't make any strong assumptions about the landmarks and their relations. We verify this by applying our method *as is* to the demanding problem of finding landmarks in birds on CUBS 200 dataset. Unlike human pose datasets, there is no clear intuitive ordering of landmarks in this dataset making our approach all the more appealing. Figure 2.11 shows qualitative results while Figure 2.12 compares our method with an *Independent* baseline which learns a set of detectors independently. Our method shows significant gains over this baseline by using the learned ordering to better propagate context.

2.4 Conclusion

We described a general method to find landmarks in images by greedily expanding groups, exploiting appearance and contextual information within the group to identify the next best landmark using a learned scoring function. Our method learns to detect landmarks in an image dependent order. Using a very simple set of features, our method is able to achieve good performance. Further, we have shown that our method performs very well on two distinct landmark finding problems underlining its general applicability. Learning the low level features and stronger encoding of context are some of the promising future directions.

Chapter 3

Learning to Localize Little Landmarks

The world is full of tiny but useful objects such as the door handle of a car or the light switch in a room. We call these *little landmarks*. We interact with many little landmarks everyday, often not actively thinking about them or even looking at them. Consider the door handle of a car. It is often the first thing we manipulate on a car, but in an image it is barely visible. Yet we know where it is (Figure 3.1). Automatically localizing little landmarks in images is hard, as they don't have a distinctive appearance of their own. These landmarks are largely defined by their context. We describe a method to localize little landmarks by discovering informative context supervised solely by the location of the little landmark. We demonstrate the effectiveness of our approach on several different datasets, including both new and established problems.

Many locations in any given image may look like the target landmark locally. However, these landmarks may occur in a specific configuration with respect to other things. Thus, some consistent pattern nearby may help find the little landmark. We refer to such a pattern as a *latent landmark*. The latent landmark may itself be hard to localize, although easier than the target. A second latent landmark may then help to localize the first one, which in turn localizes the target. Our method discovers a sequence of such landmarks, where every latent landmark resolves some ambiguity about the target landmark and ultimately leads to its localization.

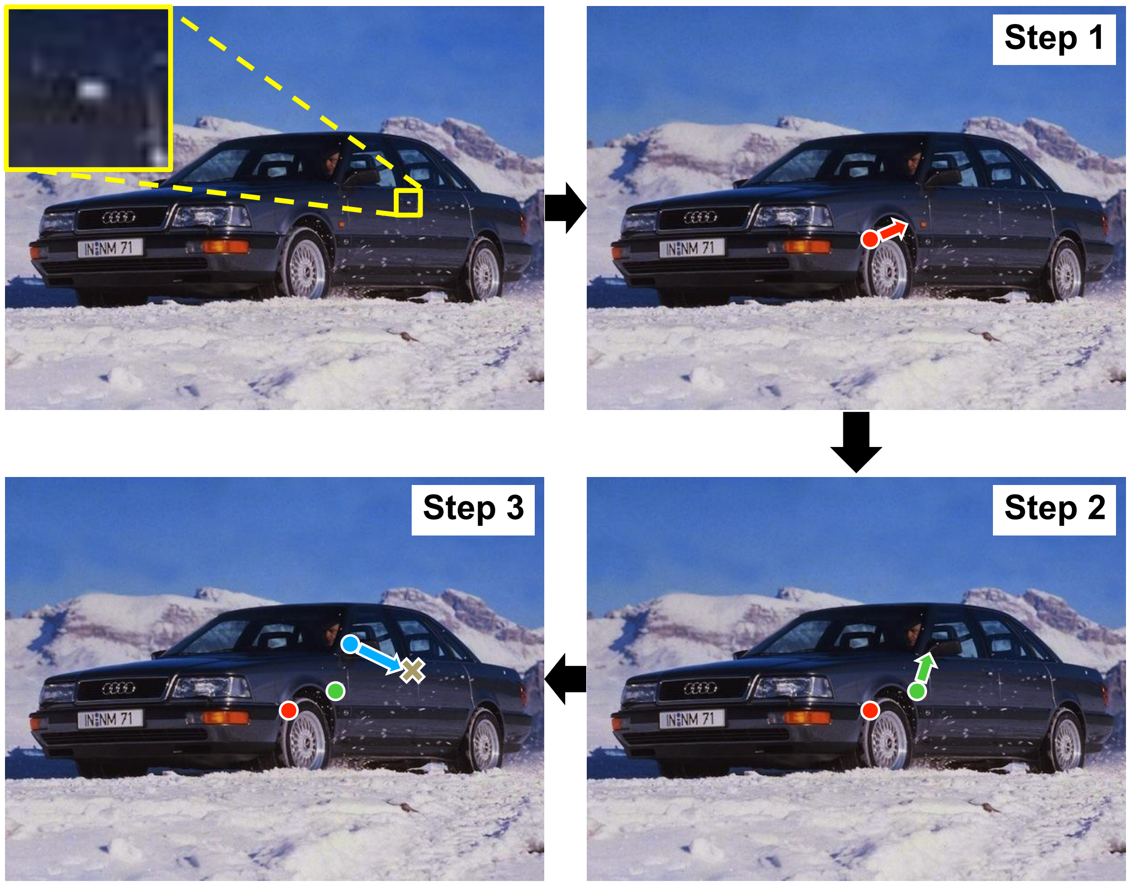


FIGURE 3.1: Several objects of interest are so tiny that they barely occupy few pixels (top-left), yet we interact with them daily. Localizing such objects in images is difficult has they do not have a distinctive local appearance. We propose a method that learns to localize such landmarks by learning a sequence of latent landmarks. Each landmark in this sequence predicts where the next landmark could be found. This information is then used to predict the next landmark and so on, until the target is found in the last step.

We expect that a latent landmark is predictive, i.e. local features around it can predict the location of the next latent landmark. For a sequence of such landmarks to be useful, every landmark should be predictive about the next one, though not necessarily the final target. Thus the last latent landmark can be a distinctive feature that can be localized on its own. It should also be able to predict the location where the next landmark can be found and so on. Our method captures this intuition and operates in steps. In each step, prediction from previous step guides where the next latent landmark can be found.

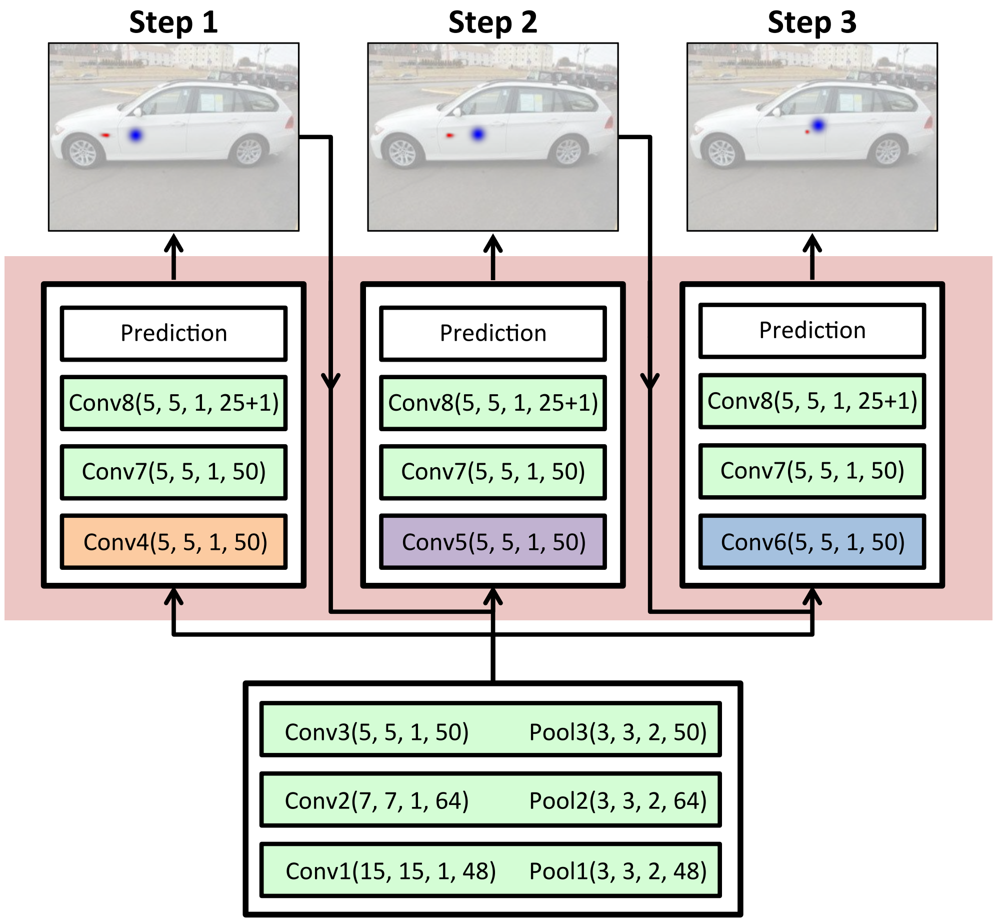


FIGURE 3.2: Model and Inference Overview. Our approach operates in steps where in each step a latent landmark (red blobs at the top, best viewed with zoom) predicts the location of the latent landmark for the next step. This is encoded as a feature map with radial basis kernel (blue blob) and passed as a feature to the next step. This process repeats until the last step when the target is localized. Green boxes show layers and parameters that are shared across steps, while orange, purple and blue show step specific layers. Format for a layer is layer_name(height, width, stride, num_output_channels).

It is then used to make the prediction for the next latent landmark. Finally, the target is predicted in the last step.

3.1 Background

Landmark localization has been well-studied in the domain of human pose estimation (Yang and Ramanan, 2013; Wang, Tran, and Liao, 2011; Tran and Forsyth, 2010; Eichner and Ferrari, 2009; Sapp, Toshev, and Taskar, 2010; Andriluka, Roth, and Schiele, 2009; Dantone et al., 2013; Toshev and Szegedy, 2013) as well as bird part localization (Liu and Belhumeur, 2013; Liu, Li, and Belhumeur, 2014; Wah et al., 2011; Shih et al., 2015). Localization of larger objects has similarly been well- studied (Felzenszwalb et al., 2010; Girshick et al., 2014). However, practically no work exists for localizing little landmarks.

Little landmarks are largely defined by their context. Any successful method for localizing them will have to use this context. Use of context to improve performance has been studied in past. In many problems, explicit contextual supervision is available. Felzenszwalb et al., 2010 use contextual re-scoring to improve object detection performance. Singh, Hoiem, and Forsyth, 2015 use context of easy landmarks to find the harder ones, Su and Jurie, 2012 use context from attributes in image classification . In contrast, our method has no access to explicit contextual supervision. Some methods do incorporate context implicitly e.g. Auto-Context (Tu and Bai, 2010), which iteratively includes information from an increasing spatial support to localize body parts. In contrast, our method learns to find a sequence of latent landmarks that are useful for finding the target little landmark without explicit contextual supervision.

Methods that discover mid-level visual elements (Singh, Gupta, and Efros, 2012; Doersch et al., 2012; Juneja et al., 2013) and use those for a task are related to our approach. Such methods first find a set of discriminative elements and then use them as a representation for the task at hand. The criterion for discovery of these elements is often not related to the final task they are used for. Some approaches have tried to address this by alternating between updating the representation and learning for the task (Parizi et al., 2014). In contrast, our method learns to find latent landmarks that are directly useful for localizing the target and is trainable end to end.

Reinforcement learning based methods are closely related to our approach. Caicedo and Lazebnik, 2015 cast object detection in a reinforcement learning framework and learn a policy to iteratively refine a bounding box for object detection, Zhang et al., 2015 learn to predict a better bounding box given an initial estimate. Our method in comparison does not have explicitly defined actions or value function. It is supervised to predict the target in the final step while an intermediate step is encouraged to be informative for the next step.

The work of Karlinsky et al., 2010 is conceptually most closely related to our method. They evaluate keypoint proposals to choose an intermediate set of locations that can be used to form chains from a known landmark to a target. The target is predicted by marginalizing over evidence from all chains. In contrast, our approach does not use keypoint proposals and learns to find the first point in the chain as well.

Another closely related approach is that of Carreira et al., 2015. They use iterative error feedback to improve accuracy with each step. Each step is supervised to predict the target and constrained to get closer to it in comparison to previous steps prediction. In contrast, our method does not supervise the intermediate steps with the target. Only the final step is directly supervised. The latent landmark can be anywhere in the image as long as they are predictive of the next one in the sequence.

Methods that learn to attend to specific parts of an image to solve a task (Ba, Mnih, and Kavukcuoglu, 2014; Xu et al., 2015; Larochelle and Hinton, 2010; Bahdanau, Cho, and Bengio, 2014; Mnih, Heess, Graves, et al., 2014) are relevant to our method. Although sharing the theme, our method learns to localize little landmarks by using a sequence of locations in the image.

3.2 Approach

The simplest scheme for finding a landmark looks at every location and decides whether it is the target landmark or not. We refer to this scheme as *Detection*. Training such a system is easy: we provide direct supervision for which location is the target. However this doesn't work well for little landmarks because they are not strongly distinctive. Now imagine we wish to use a single latent landmark to predict the location of the target, which could be far way. We refer to this scheme as *Prediction*. This is hard, because we don't have direct supervision for the latent landmark. Instead, the system must infer where these landmarks are. Furthermore, it must learn to find these landmarks *and* to use them to predict where the target is. While this is clearly harder to learn than detection, we describe a method that is successful and outperforms *Detection* (§ 3.3.1).

Prediction is hard when the best latent landmark for a target is itself hard to find. Here, we can generalize to a sequential prediction scheme (referred to as *SeqPrediction*). The system uses a latent landmark to predict the location of another latent landmark; uses that to predict the location of yet another latent landmark; and so on, until the final step predicts the location of the target. Our method successfully achieves this and outperforms *Prediction* (§ 3.3.1).

Note that another generalization that is natural, but not useful is an alternating scheme. One might estimate some distinctive latent landmarks, learn a prediction model and then re-estimate the latent landmarks conditioned on current target estimates, etc. This scheme is unhelpful when the landmark is itself hard to find. First, re-estimates tend to be poor. Second, it is tricky to learn a sequential prediction as one would have to find conditionally distinctive patterns. The objective that matters the most is the localization of target landmark. Our approach is supervised by this sole objective and can be trained end-to-end given a target landmark.

Our method thus *learns to find a sequence of latent landmarks each with a prediction model*

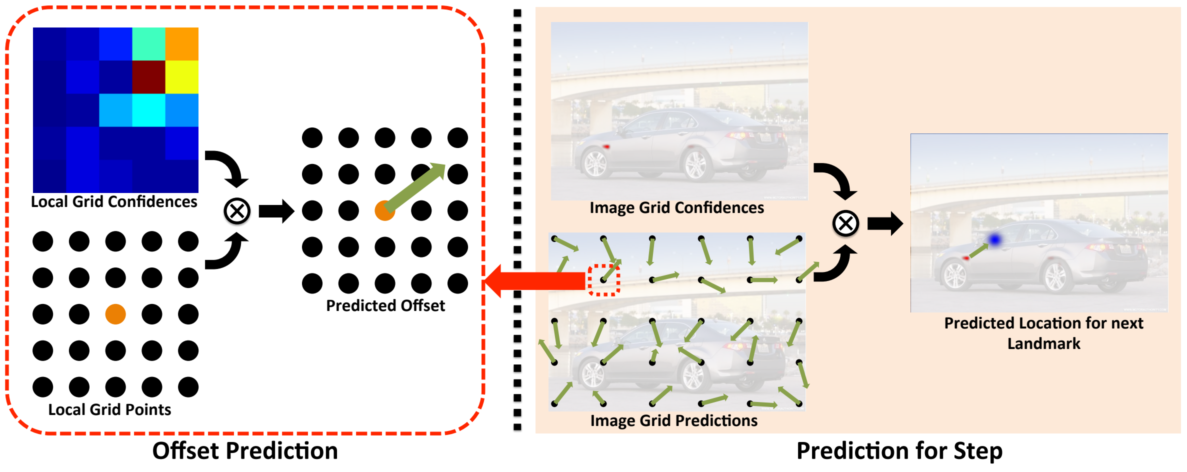


FIGURE 3.3: Prediction Scheme for a Step: On the left, we visualize how the offset prediction is made at each location. The model predicts confidences for the points on a local grid around a location of interest. The offset is then computed as a weighted average of the local grid points using the confidences. On the right, we visualize the prediction scheme for the whole image given the individual predictions. Model predicts a confidence in each offset prediction (red blob in the top image, best viewed with zoom). Individual offset predictions are then averaged using the confidences as weights to produce the final prediction. The prediction is then encoded as a radial basis kernel centered at the prediction (blue blob).

to find the next in sequence. In the following, we first provide an overview of the model, followed by the prediction scheme, and finally the training details.

3.2.1 Model and Inference

Figure 3.2 provides an overview of the model and how it is used for inference. Our method operates in steps where each step $s \in \{1, \dots, S\}$ corresponds to a *Prediction*. Each step predicts the location of the next latent landmark using the image features and the prediction from the previous step. The final step predicts the location of the target little landmark. To make the prediction, each step finds a latent landmark (Figure 3.2, red blob) and makes an offset prediction to the location of the next latent landmark relative to it. This prediction is then summarized to pass on to the next step (blue blob). Note that the spatial prediction scheme is of key importance for the system to work. We describe it in section 3.2.2.

Our system uses a fully convolutional neural network architecture. It can be viewed as a network sliding over the image to make the predictions at each location. In figure 3.2, the green boxes indicate the layers with parameters shared across various steps. Other colored boxes (orange, purple and blue) show layers that have step specific parameters. The step specific parameters allow the features of a step to quickly adapt as estimates of underlying landmarks improve. Our model is trained using stochastic gradient descent on a robust loss function. Our loss function encourages earlier steps to be informative for the later steps by penalizing disagreement between the predicted latent landmark locations and the actual ones used.

3.2.2 Prediction Scheme for a Step

Our model is fully convolutional and thus images of different sizes produce features maps of different sizes. This makes a single prediction for the whole image difficult. We instead view the image as a grid of locations $l_i, i \in \{1, \dots, L\}$, where each location can make a prediction using the sliding neural network.

Each step s produces a summary estimate of the position of the next latent landmark $P^{(s)}$. Each location l_i separately estimates this position as $p_i^{(s)}$, with a confidence $c_i^{(s)}$. Each $p_i^{(s)}$ is estimated using a novel representation scheme with several nice properties (§ 3.2.3). The individual predictions are then combined as

$$P^{(s)} = \sum_{i=1}^{L} c_i^{(s)} p_i^{(s)} \tag{3.1}$$

The local scheme for producing each $p_i^{(s)}$ looks at both the image features and $P^{(s-1)}$. The confidence $c_i^{(s)}$ is a softmax over all locations, computed as $e^{z_i^{(s)}} / \sum_i e^{z_i^{(s)}}$, where $z_i^{(s)} \in \mathbb{R}$

is the output from the network for confidence at l_i in step s. The right half of Figure 3.3 visualizes the prediction scheme. Locations with high confidences can be seen as a red blob for each of the steps in figure 3.2 and 3.3 (best viewed with zoom).

$P^{(s)}$ is then encoded as a feature map that is passed on to the subsequent step together with the image feature maps. The encoding is done by placing a radial basis kernel of fixed bandwidth β centered at the predicted location (blue blob, Figure 3.2). Note that passing the prediction as a feature instead of as a rigid constraint to the next step allows it to choose to ignore the prediction from the previous step if necessary. This is specially helpful in early stages of the training where we don't have reliable estimates of the latent landmarks or their prediction models.

Furthermore, the scheme of $P^{(s)}$ as a weighted average of several individual predictions has the advantage that it averages over redundant information from several locations and thus is robust to individual variances. With proper initialization at the beginning of the training, we can ensure that all the locations have non-zero weights and thus are explored as potential latent landmarks.

3.2.3 The Estimate at a Location

We need a prediction $p_i^{(s)}$ from location l_i at step s. Obtaining these predictions takes care. Pure regression works poorly, because it is sensitive to learning rate and weight scales and it is difficult to confine predictions to a range.

Instead, we place a local grid of G points over each l_i. Each grid point has coordinates g_j relative to l_i. We train the network to produce G confidence values $o_{j,i}^{(s)}$ for $j \in \{1, \ldots, G\}$ and at each location. These $o_{j,i}^{(s)}$ are a softmax of network outputs which themselves depend on the image as well as the feature map representing $P^{(s-1)}$. Each l_i then produces the estimate

$$p_i^{(s)} = l_i + \sum_{j=1}^{G} o_{j,i}^{(s)} g_j \tag{3.2}$$

Our scheme has several strengths. The network is required to predict confidences, rather than locations, and so deals with well-scaled values. By construction, each prediction is within the range specified by the local grid. Finally, redundancy helps control the variance of the prediction.

In our experiments we use a local 5×5 grid with $g_j(x), g_j(y) \in \{-50, -25, 0, 25, 50\}$ pixels.

3.2.4 Training

For regression tasks it is usual to use L_2 loss. However, use of L_2 loss with neural networks requires careful tuning of learning rate. Setting it too high results in an explosion of gradients at the beginning and too low slows down the learning in later epochs. Instead, we use Huber loss (eq. 3.3) for robustness.

$$\mathcal{H}(x) = \begin{cases} \frac{x^2}{2\delta}, & \text{if } |x| < \delta. \\ |x| - \frac{\delta}{2}, & \text{otherwise.} \end{cases} \tag{3.3}$$

For a vector $\mathbf{x} \in \mathbb{R}^D$ we define Huber loss as $\mathcal{H}(\mathbf{x}) = \sum_{i=1}^{D} \mathcal{H}(x_i)$. Note that the robustness arises from the fact that the gradients are exactly one for large loss values ($|x| > \delta$), and less than one for smaller values. This ensures that the gradient magnitudes doesn't change much for a large part of the training. We use $\delta = 1$.

Assume that we know the regression target $y_*^{(s)}$ for step s. Then, given the prediction $p^{(s)}$, we define the loss for step s as following

$$\mathcal{L}^{(s)} = \mathcal{H}(p^{(s)} - y_*^{(s)}) + \gamma \sum_{i=1}^{L} c_i^{(s)} \mathcal{H}(p_i^{(s)} - y_*^{(s)}) \tag{3.4}$$

The first term enforces that the prediction $p^{(s)}$ coincides with the target $y_*^{(s)}$. The second term enforces that the individual predictions for each location also fall on the target, but the individual losses are weighted by their contribution to the final prediction. We found that the use of this term with a small value of $\gamma = 0.1$ consistently leads to solutions that generalize better.

Now, the regression target for the final step S is known and is the ground truth location of the target y_*. But we do not have supervision for the intermediate steps. We would like our step s to predict the location of the latent landmark of the next step $s+1$. Note that the latent landmark for the next step is considered to be the set of locations in the image that the model considers to be predictive and therefore assigns high confidences $c_i^{(s+1)}$. Thus, a reasonable scheme is to set $y_*^{(s)} = \sum_{i=1}^{L} c_i^{(s+1)} l_i$, i.e. as the centroid of locations with confidence weights $c_i^{(s+1)}$. This encourages the prediction from step s to coincide with the locations that are predictive in next step. Figure 3.4 visualizes the ground truth assignment scheme.

We define the full loss for a given sample as a weighted sum of the losses from individual steps as following

$$\mathcal{L} = \sum_{s=1}^{S} \lambda_s \mathcal{L}^{(s)} + \mathcal{R}(\theta) \tag{3.5}$$

We use $\lambda_s = 0.1$, except for the final step S where $\lambda_S = 1$. $\mathcal{R}(\theta)$ is a regularizer for the parameters of the network. We use L_2 regularization of network weights with a multiplier of 0.005.

Training Details: We train our model through back-propagation using stochastic gradient

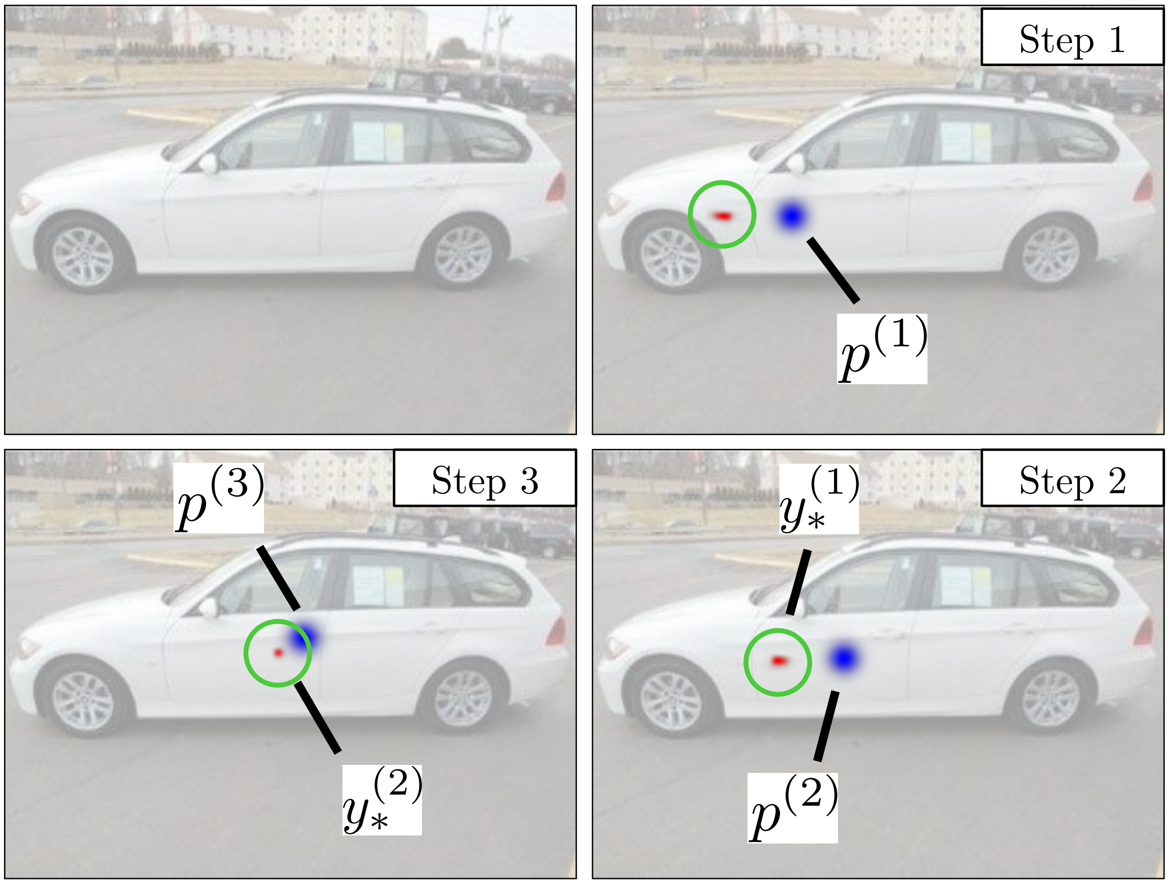


FIGURE 3.4: Supervision for intermediate steps. For any step s, the latent landmark of the next step (red blob) is used as a target. Intuitively, this encourages the current step to predict the locations that will be assigned a high confidence in the next step. For example, in the figure above, $y_*^{(1)}$ computed from the second step serves as the target for the prediction from the first step $p^{(1)}$ (blue blob). Note that ground truth location of the final target will be used as $y_*^{(3)}$ in the above figure.

descent with momentum. The errors are back-propagated across the steps through the radial basis kernel based feature encoding of the latent landmark prediction in each step. Since our model is recurrent, we found that the use of gradient scaling makes the optimization better behaved Pascanu, Mikolov, and Bengio, 2012. We do this by scaling the gradient for filters with combined L_2 norm more than 1000 back down to 1000. We initialize the weights using the method suggested by Glorot *et al.* Glorot and Bengio, 2010.

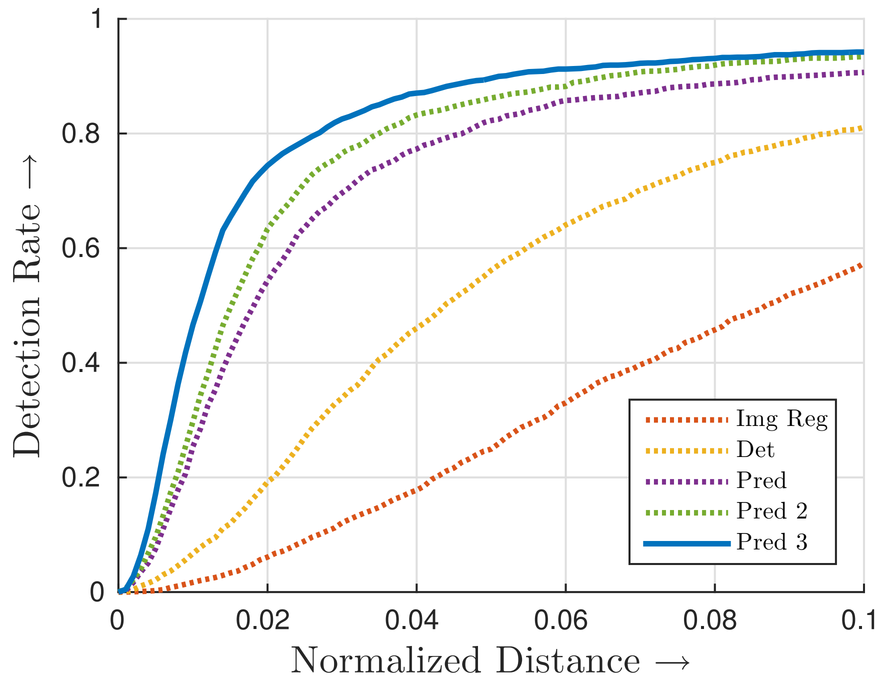


FIGURE 3.5: Our method localizes car door handles very accurately. Pred 3, the three step *SeqPrediction* scheme, outperforms the other schemes. Det, Pred and Pred 2 are *Detection*, *Prediction* and two step *SeqPrediction* schemes respectively. While, Img Reg is a baseline that replaces classification layer by a regression layer in the VGG-M model. Table 3.1 reports detection rates at a fixed normalized distance of 0.02.

We augment the datasets by including left-right flips as well as random crops near the border. The images are scaled to make the longest side 500 pixels.

3.3 Experiments

We evaluate our approach both on more difficult variants of established tasks and on several new tasks. We choose tasks which require localizing a little landmark. We present two new datasets: The Light Switch Dataset (LSD) and the Car Door Handle Dataset (CDHD) that emphasize practically important and hard to find little landmarks. Further, we evaluate our method for the tasks of beak localization on the Caltech UCSD Birds

				Seq. Prediction	
Method	Img Reg	Det	Pred	Pred 2	Pred 3
Det. Rate	6.1	19.2	54.3	63.3	**74.4**

TABLE 3.1: Detection rates for the Car Door Handle Dataset at the fixed normalized distance of 0.02. The three step scheme (Pred 3) performs significantly better than the alternatives. Refer to Figure 3.5 for more details.

Dataset (CUBS) and wrist localization on the Leeds Sports Dataset (LSP). Note that we refer to the three step *SeqPrediction* as *Ours* in the following unless specified otherwise .

Evaluation Metric: We adopt the generally accepted metric of plotting detection rate against normalized distance from ground truth for all datasets except CUBS. For LSP, normalization uses torso height, for CDHD it uses car bounding box height and for LSD it uses switch board height. For CUBS we report PCP computed as detection rate for an error radius define as $1.5 \times \sigma_{human}$, where σ_{human} is the standard deviation of the human annotations Liu and Belhumeur, 2013.

3.3.1 Car Door Handle Dataset

Our method finds car door handles very accurately (Figure 3.5 and 3.9 and Table 3.1). It shows superior performance in comparison to various baselines. The evaluation provides clear evidence that the sequential prediction scheme with three steps is superior to the alternatives evaluated. Det is the *Detection* method, Reg is the *Prediction* method and Reg 2 and Reg 3 are two and three step *SeqPrediction* methods respectively. Use of *Prediction* instead of *Detection* gives considerable performance improvement, while *SeqPrediction* provides additional improvement. Img Reg is a baseline implemented by taking the VGG-M model Chatfield et al., 2014, removing the top classification layer and replacing it by a 2D regression layer. The learning rate for all the layers was set to 0.1 times the learning

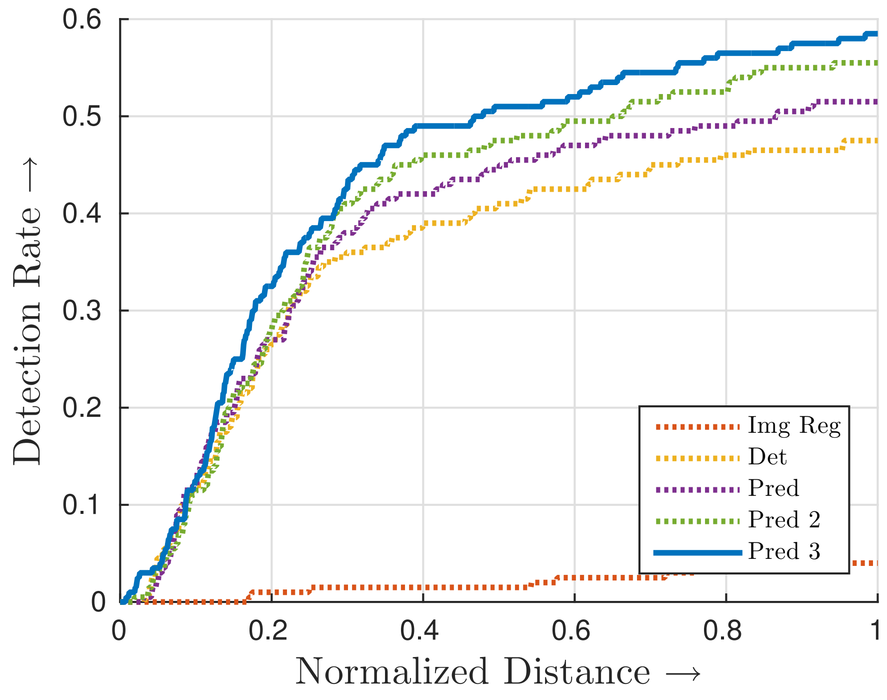


FIGURE 3.6: Our method localizes light switches relatively well in comparison to the baselines. The baselines are the same as the ones for Car Door Handle dataset. Table 3.2 reports detection rates at a fixed normalized distance of 0.5.

rate λ_r for the regression layer. The model performed best with a learning rate $\lambda_r = 0.01$, chosen by trying values in $\{0.1, 0.01, 0.001\}$.

Dataset details: To collect a dataset with car door handles, 4150 images of the Stanford Cars dataset for fine grained categorization Krause et al., 2013 were annotated. Annotators were asked to annotate the front door handle of the visible side. The handle was marked as hidden for frontal views of the car when it was not visible. We use the training and test split of the original dataset.

3.3.2 Light Switch Dataset

Our method finds light switches with reasonable accuracy (Figure 3.6 and 3.9 and Table 3.2). The baselines are the same as the ones for the Car Door Handle dataset. Again, the three

				Seq. Prediction	
Method	Img Reg	Det	Pred	Pred 2	Pred 3
Det. Rate	1.5	41.0	44.5	47.5	**51.0**

TABLE 3.2: Detection rates for the Light Switch Dataset at the fixed normalized distance of 0.5. Again, the three step scheme performs better than alternatives.

Method	PCP
Liu and Belhumeur, 2013	49.0
Liu, Li, and Belhumeur, 2014	61.2
Shih et al., 2015	51.8
Ours	**64.1**

TABLE 3.3: Our method outperforms several state of the art methods for localizing beaks on the Caltech UCSD Birds Dataset. Note that this comparison is biased against our method; others are trained with *all* landmarks while ours is supervised only by beak.

step scheme performs better than the alternatives.

Dataset details: With the aim of building a challenging lone landmark localization problem, we collected the Light Switch Dataset (LSD). It has 822 annotated images (622 train, 200 test). Annotators were asked to mark the middle points of the top and bottom edge of the switch board. The location of the light switch is approximated as the mean of these. These two points also provide approximate scale information used for normalization in evaluation. This dataset is significantly harder than the Car Door Handles dataset as context around light switches exhibits significant variation in appearance and scale.

3.3.3 Caltech UCSD Birds Dataset - Beaks

Caltech-UCSD Birds 200 (2011) dataset (CUBS 200) (Wah et al., 2011) contains 5994 training and 5794 testing images with 15 landmarks for birds. We evaluate our approach on the task of localizing *beak* as the target landmark. We chose beak because it is one of the hardest landmarks and several state of the art approaches do not perform well on this. We used the provided train and test splits for the dataset.

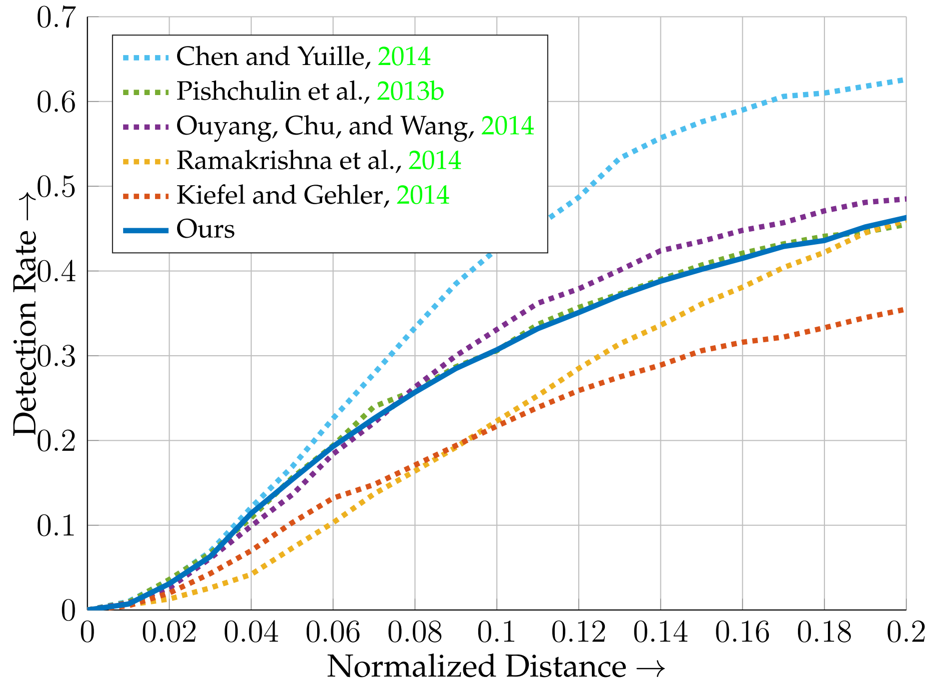


FIGURE 3.7: Our method, while supervised only by the location of left wrist, performs competitively against several state of the art methods for localizing the left wrist landmark on the Leeds Sports Dataset.

Our method, *while only having access to the beak landmark's location during training* (all other methods are trained using other landmarks as well), outperforms several state of the art methods (Table 3.3).

3.3.4 Leeds Sports Dataset - Left Wrist

Leeds Sports Dataset (LSP) (Johnson and Everingham, 2010) contains 1000 training and 1000 testing images of humans in difficult articulated poses with 14 landmarks. We choose left wrist as the target landmark as wrists are known to be difficult to localize. We use the Observer Centric (OC) annotations (Eichner and Ferrari, 2012) and work with provided training/test splits.

Our method performs competitively with several recent works all of which train their

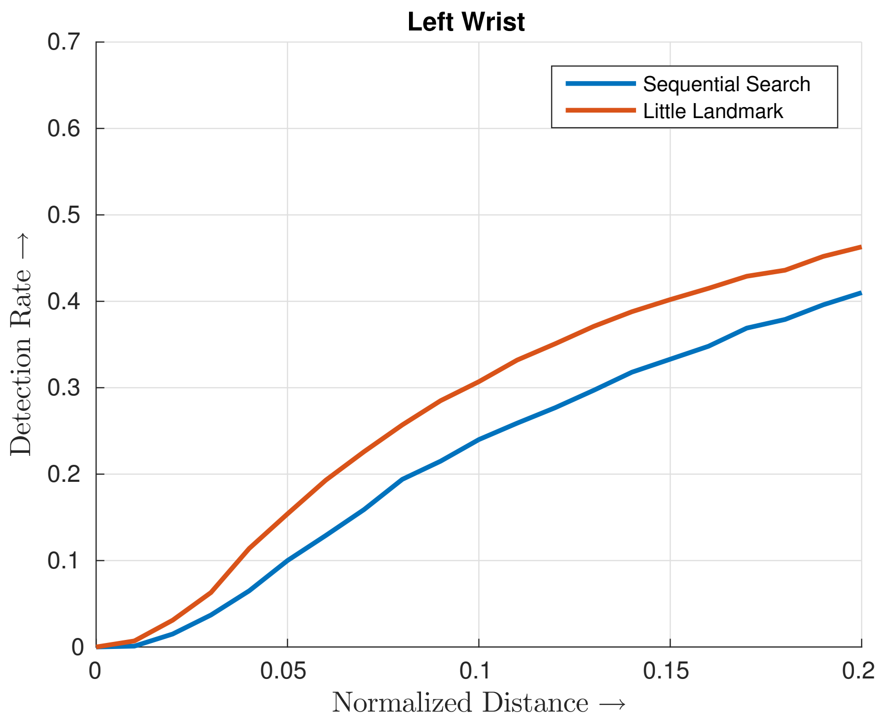


FIGURE 3.8: In comparison to the method described in chapter 2 (Sequential Search) our method is better at localizing left wrist despite being supervised only by the location of the wrist. The superior performance is likely due to the use of better features as well as better modeling of the context.

method using other landmarks as well (Figure 3.7). Figure 3.8 compares the performance with the method described in 2. The superior performance is likely due to the use of better features as well as better modeling of the context.

3.4 Discussion

Figure 3.9 shows some qualitative results for various datasets. First thing to note is the pattern in the locations of the latent landmarks for each of the datasets shown. For cars, it tends to find the wheel as the first latent landmark and then moves towards the door handle in subsequent steps. For light switches it relies on finding the edge of the door first. For birds, the first landmark tends to be on the neck, followed by one near the eye

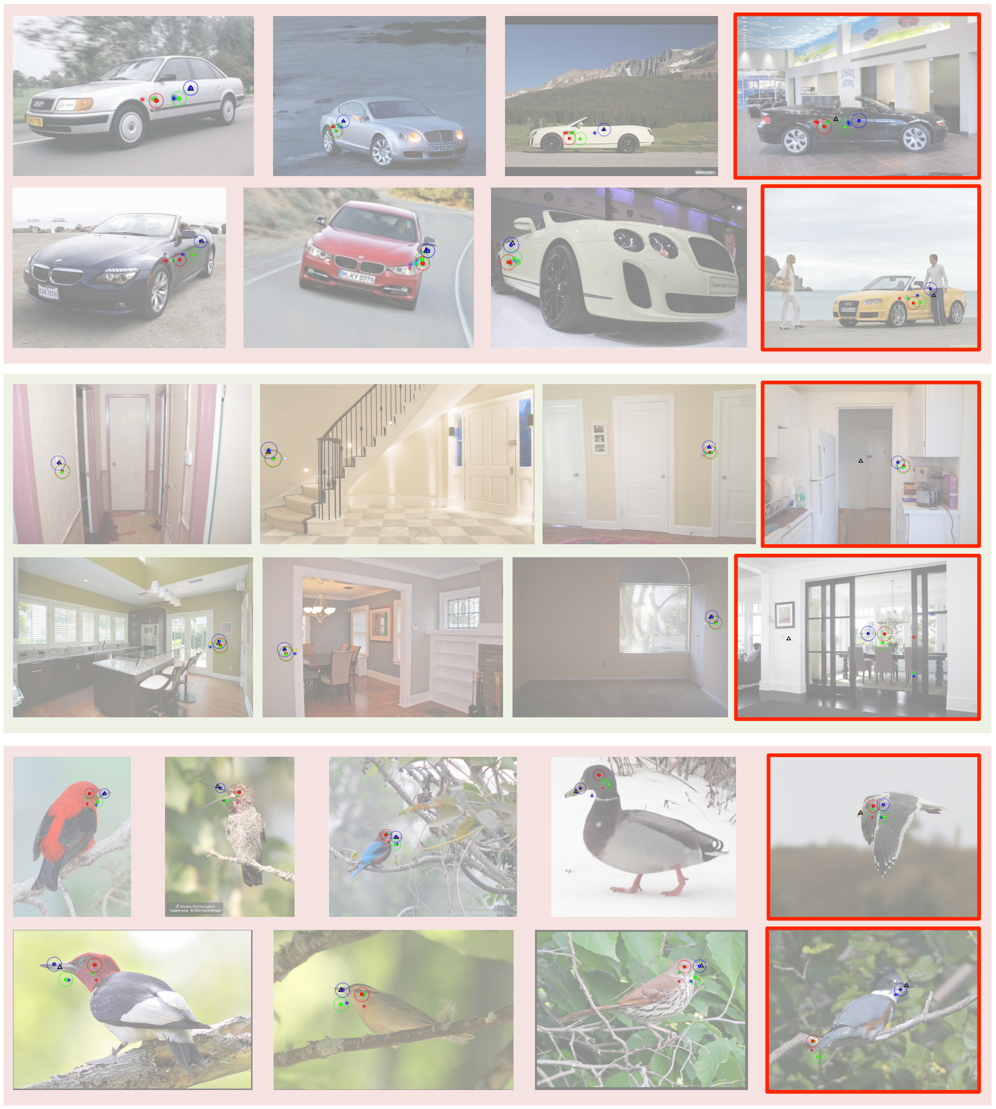


FIGURE 3.9: Qualitative results for the CDHD (top two rows), LSD (middle two rows) and the CUBS200 Dataset (last two rows). Ground truth locations are shown as black triangles. Step 1, 2 and 3 are color coded as Red, Green and Blue respectively. Colored blobs show the locations of the latent landmarks for each step. Numbered solid circles show the predicted location of the next latent landmark by each step. Dotted circles show the bandwidth of the kernel used to encode the predictions. Note the patterns in the locations of the latent landmarks. For cars, the first latent landmark tends to be on the wheel and later ones get closer to the door handle. For light switches it relies on finding the edge of the door first. For birds, the first landmark tends to be on the neck, followed by one near the eye and the last tends to be outside neck. Rightmost column shows failure/interesting cases. The latent landmarks tend to be closer to the prediction from the previous step. Although, they are not constrained to do so (bottom right bird image). Typical failures include clutter, circumstantial context (door frame) and rarer examples (e.g. flying bird).

and the last tends to be outside at the curve neck and beak. It is remarkable that these patterns emerge solely from the supervision of the target landmark. Also, note that these patterns are not rigid; they adapt to the image evidence. This is primarily due to the fact that our method does not impose any hard constraints. Later steps can choose to ignore the evidence from the earlier steps. This property allows our model to be trained effectively, specially in the beginning when the latent landmarks as well as their prediction models are not known.

We explored two other architectures for propagating information from one step to the next and found that the current scheme performs the best in terms of the final performance. In the first scheme, step specific weights were at the top instead of at the bottom of the recurrent portion of our model (Figure 3.2, middle block). In the second scheme, instead of passing the encoded prediction as a feature, it was used as a prior to modulate the location confidences of the next step.

3.5 Conclusion

We described a method to localize little landmarks by finding a sequence of latent landmarks and their prediction models. We demonstrated strong performance of our method on harder variants of several existing and new tasks. The success of our approach arises from the spatial prediction scheme and the encoding of information from one step to be used by the next. A novel and well behaved local estimation model coupled with a robust loss ensures that our model can be trained. Some promising future directions include generalizing sequence of latent landmarks to Directed Acyclic Graphs of latent landmarks and accumulating all the information from previous steps to be used as features for the next step.

Chapter 4

Localizing Multiple Landmarks

In many practical problems, we are interested in localizing several landmarks. Human pose estimation is one of the most studied example of such a problem. Here the goal is to localize anatomical landmarks, such as *shoulder*, *wrist*, *ankle* etc., of a person. Chapter 2 hypothesized that some of these are easier to find while others are harder and described a method that learns to sequentially find landmarks. The method uses context from easier ones (detected earlier) to improve the detection of harder ones.

In chapter 3 we discussed a method for localizing little landmarks, namely landmarks that do not have much appearance of their own. A key feature of this method is that it learns to discover and use context for localizing a single landmark without any explicit supervision of other contextually useful landmarks. It does so by detecting latent landmarks which are predictive and contextually useful for the landmark of interest. It attempts the most general problem of localizing a single landmark of interest.

In this chapter, we revisit the problem of localizing multiple landmarks by extending the framework of latent landmarks described in Chapter 3. This framework allows us to not only pool context from explicit supervision, as used in Chapter 2, but also discover and use unlabeled context.

4.1 Background

In addition to the background discussed in Sections 2.1 and 3.1, the recent proliferation of deep convolutional neural network based architectures for pose estimation (Carreira et al., 2015; Chen and Yuille, 2014; Ouyang, Chu, and Wang, 2014; Tompson et al., 2014; Tompson et al., 2015; Pishchulin et al., 2015; Insafutdinov et al., 2016; Pfister, Charles, and Zisserman, 2015; Wei et al., 2016) is relevant to the work described in this chapter. Most relevant are the work of Toshev and Szegedy, 2013 and Carreira et al., 2015. While Toshev and Szegedy, 2013 regress to the 2D coordinates similar to us, our method learns latent landmarks with additional structure. Carreira et al., 2015 use a iterative method with a feature coding that is similar to us, however each step is directly optimized for the final task and is not trainable end-to-end.

4.2 Approach

The approach described in chapter 3 is extended to predict multiple keypoints while keeping the model and inference quite similar. As before, inference proceeds in steps. The primary difference is that each step makes K predictions, one for each target landmark. Thus, each step receives all the prediction from the previous step as a set of additional feature channels and predicts the location of the next set of K latent landmarks. Finally, the last step predicts the locations of the K target landmarks. Conceptually, the methods learns K sequences for K target landmarks. However, note that each step has access to all the predictions from the previous step. Thus, the method can condition its prediction for a latent or a target landmarks on all the predictions from the previous steps.

4.2.1 Visibility Prediction

In general, some of the landmarks may not be visible due to occlusion or cropping. We extended the approach to handle such scenarios by adding an additional visibility prediction in the last step. This is achieved by adding an additional branch for visibility prediction in the network. This branch receives the same input as to the last step and has the same structure and number of parameters as the branch for location prediction. However, instead of predicting the offset to the next landmark, each location $i \in \{1, \dots, L\}$ predicts the visibility $v_{i,k}$ of each target landmark $k \in \{1, \dots, K\}$ along with a confidence $c_{i,k}$. Final visibility prediction is done by

$$v_k = \sum_{i=1}^{L} c_{i,k} v_{i,k} \tag{4.1}$$

4.3 Model and Training

Model is largely the same as described in section 3.2.1 with the addition of a branch for visibility prediction. Individual visibilities $v_{i,k}$ are predicted by feeding the output of the last convolution layer of the branch through a sigmoid. Figure 4.1 shows the full model.

For datasets with visibility annotations of landmarks, a separate cross entropy loss for each visibility prediction is used. For others, the visibility branch is dropped.

4.4 Implementation Details

Model is trained through back-propagation using stochastic gradient descent with momentum. The errors are back-propagated across the steps through the radial basis kernel based feature encoding of the latent landmark prediction in each step. Since our model is recurrent, we found that the use of gradient scaling makes the optimization better behaved

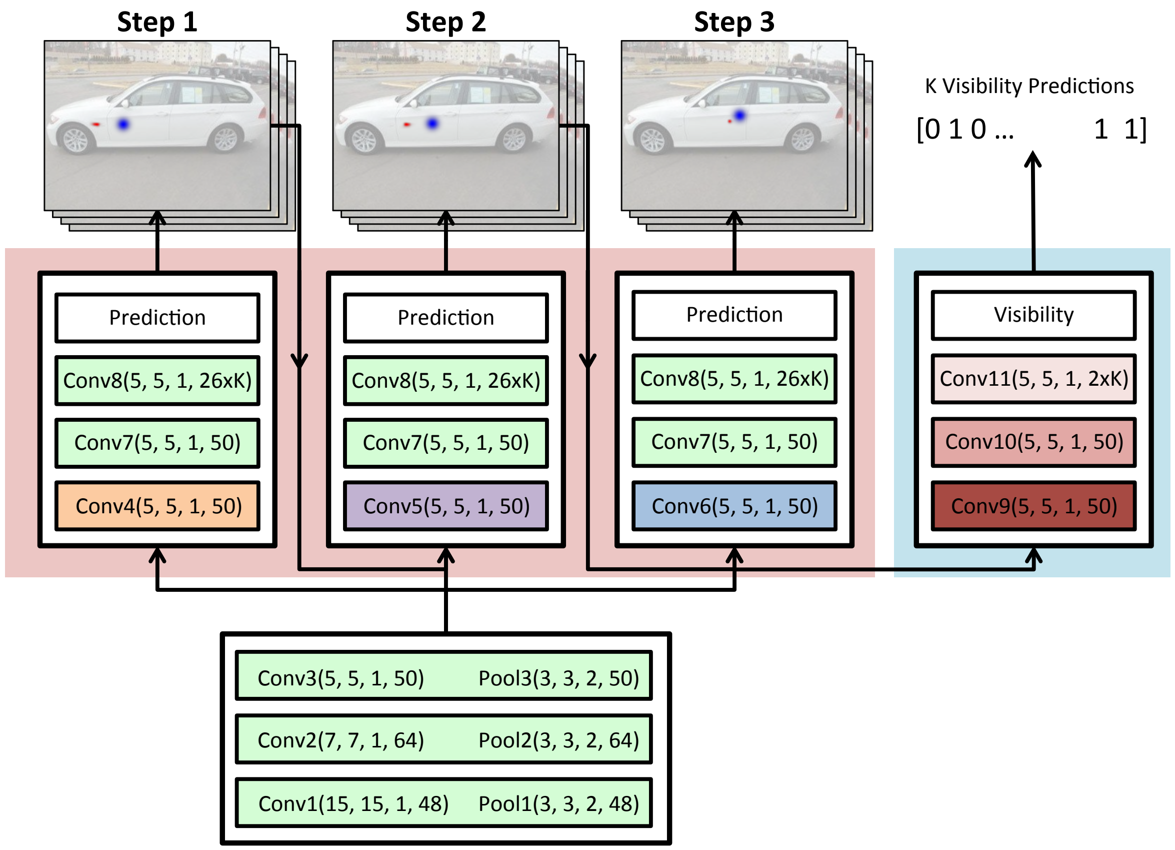
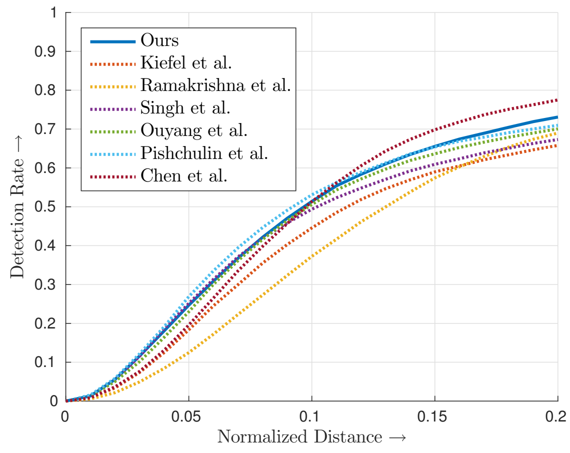


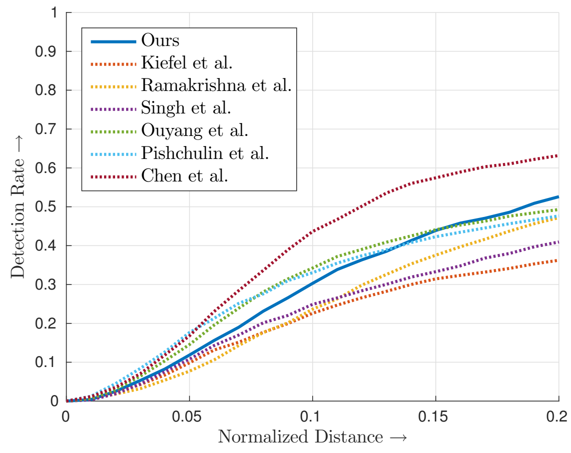
FIGURE 4.1: We extend the model described in chapter 3. As before, our approach operates in steps where in each step K latent landmarks (red blobs at the top for one of those, best viewed with zoom) predict the locations of the latent landmarks for the next step. This is encoded as K feature maps with radial basis kernel (blue blob) and passed as a feature to the next step. This process repeats until the last step when the K targets are localized. Green boxes show layers with parameters shared across steps, while other colors show step specific layers. Format for a layer is layer_name(height, width, stride, num_output_channels). An additional branch in the last step predicts the visibility of each target landmark (light blue box).

(Pascanu, Mikolov, and Bengio, 2012). We do this by scaling the gradient for filters with combined L_2 norm more than 1000 back down to 1000. We initialize the weights using the method suggested by Glorot and Bengio, 2010.

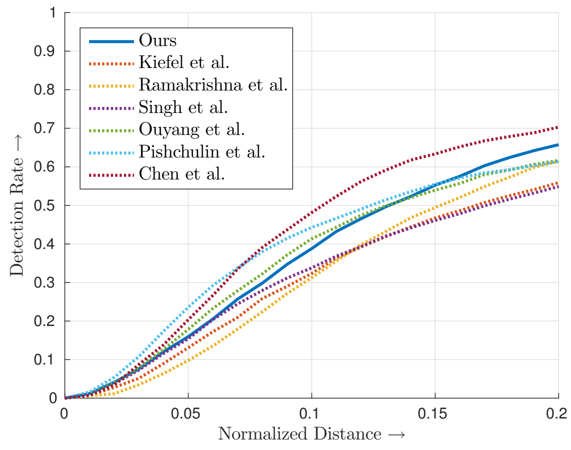
We augment the datasets by including left-right flips, random color augmentation (Deng et al., 2009), random scaling and cropping. Image are scaled such that longest side is smaller than 500 pixels. Random cropping is done by cropping the border randomly such



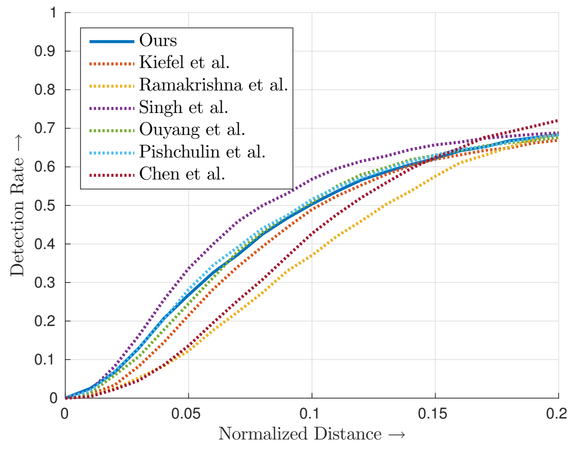
(A) all



(B) wrist



(C) elbow



(D) ankle

FIGURE 4.2: Our method compares favorably with other leading approaches for human landmark detection on LSP. The figure compares various approaches using the PCK evaluation metric for (A) all, (B) wrist, (C) elbow and, (D) ankle

that all visible landmarks are always included. A mini-batch of 20 images is constructed by randomly placing the images in a 500×500 pixel grid. A momentum of 0.9 is used along with a learning rate schedule of 0.1 for first 400 epochs, 0.01 for next 150 and 0.001 for next 50.

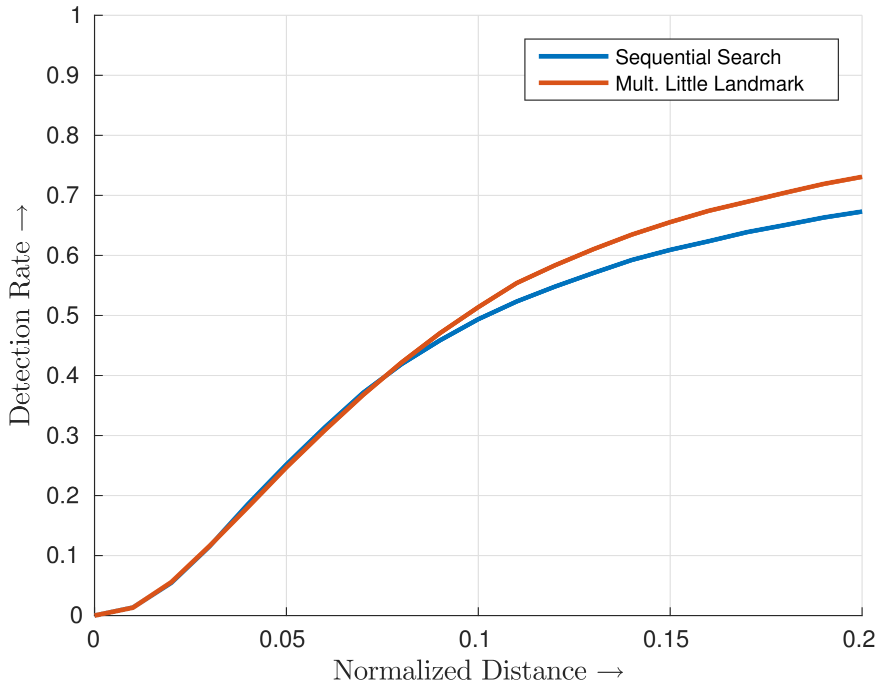


FIGURE 4.3: Comparison with method described in chapter 2. Performance gain for larger distances is indicative of better contextual modeling.

4.5 Experiments

We evaluate the described method on two different tasks of predicting landmarks for: 1) Humans and, 2) Birds. We first describe the datasets and then present our results.

4.5.1 Datasets and Evaluation Metric

Humans: We evaluate our method on Leeds Sports Dataset (LSP) Johnson and Everingham, 2010. LSP contains 1000 training and 1000 testing images of humans in difficult articulated poses with 14 landmarks marking different joint locations. Note that for occluded landmarks, annotations mark the expected location. The original annotations for the dataset are person centric, i.e. joints are labeled with respect to the person in the image. Thus, the left arm of a person looking at the camera will be on a different side in

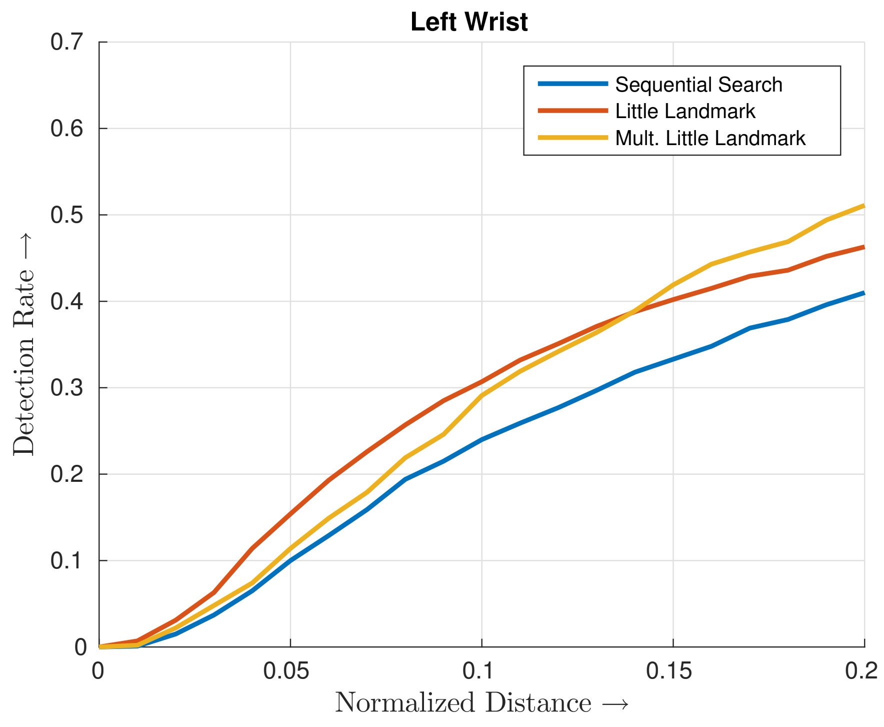


FIGURE 4.4: Performance comparison for left wrist with approaches described in Chapter 2 and 3. Wrist accuracy is comparatively lower for smaller distances while higher for larger distances. Higher accuracy for larger distances is due to better contextual information resulting from supervision of multiple landmarks. Lower accuracy for smaller distances is due to interference in shared features from supervision of multiple keypoints and may require better tuning of hyper-parameters (learning rate, training steps).

comparison to a person looking away from it. However, for ease of comparison with prior art we use the observer centric (OC) annotations provided by Eichner and Ferrari, 2012. In observer centric annotations the joint are labeled with respect to the observer. Thus, whichever arm is on the left in the image would be labeled left irrespective of the direction the person is facing.

We compare various methods using the Percentage Correct Keypoints (PCK) evaluation metric Yang and Ramanan, 2013. It plots the detection rate against normalized distance from ground truth. Normalization is done using the torso height.

Birds: We evaluate our method on the Caltech-UCSD Birds 200 (2011) dataset (CUBS 200) Welinder et al., 2010. It contains 5994 training and 5794 testing images with 15 landmarks for birds. Due to the anatomy of the birds, some of the landmarks such as *left-wing, right-wing, left-eye, right-eye* are visible only half the time. Dataset does not provide annotations for the invisible landmarks. This requires a modification to the model as described in section 4.3 for visibility prediction.

Methods are evaluated using the code provided by Liu and Belhumeur, 2013. Evaluation metric is PCP (Percent Correct Parts), which is defined as the percentage of keypoints localized within 1.5 times the annotator standard deviation. We received the pre-computed standard deviations and evaluation code from Liu and Belhumeur, 2013 to avoid any discrepancies during evaluation.

For all the datasets we work with the provided train and test splits.

4.5.2 Leeds Sports Dataset (LSP)

Our method performs competitively with several recent works. Figure 4.2 shows the combined evaluation for all the landmarks using the PCK measure. Figure 4.3 compares the performance with that of the method described in chapter 2. Performance gain for larger distances is indicative of better contextual information, due to the supervision provided for multiple landmark locations. Figure 4.4 compares the performance for wrist localization with the approaches described in chapter 2 and 3. Our method outperforms both for larger errors while under-performs the approach in chapter 2 for lower errors. While we expect our method to perform better at lower errors as well, above result may be due to slow convergence of model as both models were trained to same number of epochs.

4.5.3 Caltech UCSD Birds Dataset (CUBS 200)

Table 4.1 shows that our methods outperforms several recent approaches for the task of localizing bird landmarks. Further, table 4.2 shows that our extended method for multiple

Part	Liu *et al.*(2013)	Liu *et al.*(2014)	Shih *et al.*(2015)	Ours
back	62.1	64.5	**74.9**	73.9
beak	49.0	61.2	51.8	**66.2**
belly	69.0	71.7	81.8	**81.9**
breast	67.0	70.5	77.8	**78.9**
crown	72.9	76.8	77.7	**82.4**
forehead	58.5	72.0	67.5	**74.0**
eye	55.7	**70.0**	61.3	68.7
leg	40.7	45.0	52.9	**55.5**
wing	71.6	74.4	**81.3**	80.9
nape	70.8	79.3	76.1	**82.4**
tail	40.2	46.2	59.2	**63.1**
throat	70.8	80.0	78.7	**82.9**
Total	59.7	66.7	69.1	**73.1**

TABLE 4.1: Comparison of per-part PCP with Liu and Belhumeur, 2013, Liu, Li, and Belhumeur, 2014 and Shih et al., 2015. Our method outperforms these recent methods that explicitly train on the CUBS dataset.

Part	Liu *et al.* (2013)	Liu *et al.* (2014)	Shih *et al.* (2015)	Singh *et al.* (2016)	Ours
beak	49.0	61.2	51.8	64.1	**66.2**

TABLE 4.2: Comparison of PCP for localization of the *beak* landmark. Our extended method for multiple landmarks further improve the localization accuracy by incorporating contextual supervision from other landmarks.

landmarks further improve the localization accuracy of *beak* landmark by incorporating contextual supervision from other landmarks.

4.6 Conclusion

Our extended method for localizing multiple landmarks performs competitively on the Leeds Sports dataset and outperforms several recent approaches on the Caltech UCSD birds dataset. Our method effectively incorporates any available contextual supervision and is able to use it to further improve the performance of the individual landmark localization.

Chapter 5

Swapout - A Stochastic Method for Training DNNs

Swapout is a stochastic training method for general deep networks. It is a generalization of dropout Srivastava et al., 2014 and stochastic depth Huang et al., 2016 methods. Dropout zeros the output of individual units at random during training, while stochastic depth skips entire layers at random during training. In comparison, the most general swapout network produces the value of each output unit independently by reporting the sum of a randomly selected subset of current and all previous layer outputs for that unit. As a result, while some units in a layer may act like normal feedforward units, others may produce skip connections and yet others may produce a sum of several earlier outputs. In effect, swapout averages over a very large set of architectures that includes all architectures used by dropout and all used by stochastic depth.

Experiments described here focus on a version of swapout which is a natural generalization of the residual network He et al., 2015; He et al., 2016. It is shown that this results in improvements in accuracy over residual networks with the same number of layers.

Improvements in accuracy are often sought by increasing the depth, leading to serious practical difficulties. The number of parameters rises sharply, although recent works such as Simonyan and Zisserman, 2014; Szegedy et al., 2014 have addressed this by reducing the filter size Simonyan and Zisserman, 2014; Szegedy et al., 2014. Another issue resulting from increased depth is the difficulty of training longer chains of dependent variables. Such difficulties have been addressed by architectural innovations that introduce shorter paths from input to loss either directly Szegedy et al., 2014; Srivastava, Greff, and Schmidhuber, 2015; He et al., 2015 or with additional losses applied to intermediate layers Szegedy et al., 2014; Lee et al., 2015. Currently, the deepest networks that have been successfully trained are residual networks (1001 layers He et al., 2016). This work shows that increasing the depth of swapout networks increases their accuracy.

There is compelling experimental evidence that these very large depths are helpful, though this may be because architectural innovations introduced to make networks trainable reduce the capacity of the layers. The theoretical evidence that a depth of 1000 is *required* for practical problems is thin. Bengio and Dellaleau argue that circuit efficiency constraints suggest increasing depth is important, because there are functions that require exponentially large shallow networks to compute Bengio and Delalleau, 2011. Less experimental interest has been displayed in the width of the networks (the number of filters in a convolutional layer). This work shows that increasing the width of swapout networks leads to significant improvements in their accuracy; an appropriately wide swapout network is competitive with a deep residual network that is 1.5 orders of magnitude deeper and has more parameters.

Contributions: Swapout is a novel stochastic training scheme that can sample from a rich set of architectures including dropout, stochastic depth and residual architectures as special cases. Swapout improves the performance of the residual networks for a model of the same depth. Wider but much shallower swapout networks are competitive with very deep residual networks.

5.1 Background

Convolutional neural networks have a long history (see the introduction of LeCun et al., 1998). They are now intensively studied as a result of recent successes (e.g. Krizhevsky, Sutskever, and Hinton, 2012). Increasing the number of layers in a network improves performance Simonyan and Zisserman, 2014; Szegedy et al., 2014 if the network can be trained. A variety of significant architectural innovations improve trainability, including: the ReLU Nair and Hinton, 2010; Glorot, Bordes, and Bengio, 2011; batch normalization Ioffe and Szegedy, 2015; and allowing signals to skip layers.

Our method exploits this skipping process. Highway networks use gated skip connections to allow information and gradients to pass unimpeded across several layers Srivastava, Greff, and Schmidhuber, 2015. Residual networks use identity skip connections to further improve training He et al., 2015; extremely deep residual networks can be trained, and perform well He et al., 2016. In contrast to these architectures, our method skips at the unit level (below), and does so randomly.

Our method employs randomness at training time. For a review of the history of random methods, see the introduction of Rahimi and Recht, 2007, which shows that entirely randomly chosen features can produce an SVM that generalizes well. Randomly dropping out unit values (dropout Srivastava et al., 2014) discourages co- adaptation between units. Randomly skipping layers (stochastic depth) Huang et al., 2016 during training reliably leads to improvements at test time, likely because doing so regularizes the network. The precise details of the regularization remain uncertain, but it appears that stochastic depth represents a form of tying between layers; when a layer is dropped, other layers are encouraged to be able to replace it. Each method can be seen as training a network that averages over a family of architectures during inference. Dropout averages over architectures with "missing" units and stochastic depth averages over architectures with "missing" layers. Other successful recent randomized methods include dropconnect Wan et al., 2013 which generalizes dropout by dropping individual connections instead of units (so dropping several connections together), and stochastic pooling Zeiler and Fergus, 2013 (which regularizes by replacing the deterministic pooling by randomized pooling). In contrast, our method skips layers randomly at a unit level enjoying the benefits of each method.

Recent results show that (a) stochastic gradient descent with sufficiently few steps is stable (in the sense that changes to training data do not unreasonably disrupt predictions) and (b) dropout enhances that property, by reducing the value of a Lipschitz constant (Hardt, Recht, and Singer, 2015, Lemma 4.4). Swapout enjoys the same behavior as dropout in this framework.

Like dropout, the network trained with swapout depends on random variables. A reasonable strategy at test time with such a network is to evaluate multiple instances (with different samples used for the random variables) and average. Reliable improvements in accuracy are achievable by training distinct models (which have *distinct* sets of parameters), then averaging predictions Szegedy et al., 2014, thereby forming an explicit ensemble. In contrast, each of the instances of our network in an average would draw from the *same* set of parameters (referred to as an implicit ensemble). Srivastava et al. argue that, at test time, random values in a dropout network should be replaced with expectations, rather than taking an average over multiple instances Srivastava et al., 2014 (though they use explicit ensembles, increasing the computational cost). Considerations include runtime at test; the number of samples required; variance; and experimental accuracy results. For our model, accurate values of these expectations are not available. Section 5.3 shows that (a) swapout networks that use estimates of these expectations outperform strong comparable baselines and (b) in turn, these are outperformed by swapout networks that use an implicit ensemble.

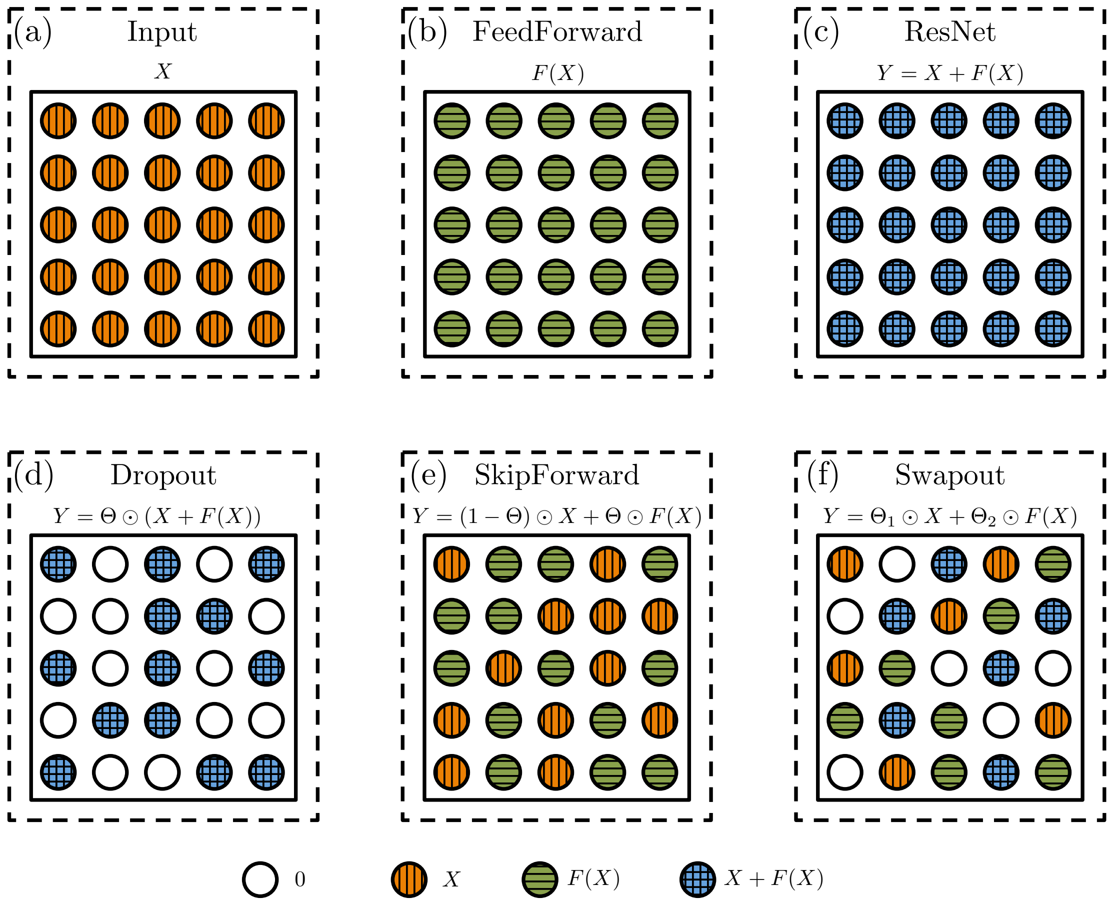

(a) Input
X
(b) FeedForward
$F(X)$
(c) ResNet
$Y = X + F(X)$
(d) Dropout
$Y = \Theta \odot (X + F(X))$
(e) SkipForward
$Y = (1 - \Theta) \odot X + \Theta \odot F(X)$
(f) Swapout
$Y = \Theta_1 \odot X + \Theta_2 \odot F(X)$
0
X
$F(X)$
$X + F(X)$

FIGURE 5.1: Visualization of architectural differences, showing computations for a block using various architectures. Each circle is a unit in a grid corresponding to spatial layout, and circles are colored to indicate what they report. Given input X (**a**), all units in a feed forward block emit $F(X)$ (**b**). All units in a residual network block emit $X + F(X)$ (**c**). Dropout randomly chooses between 0 and $X + F(X)$ *per unit* (**d**). A skipforward network randomly chooses between reporting X and $F(X)$ *per unit* (**e**). Finally, swapout randomly chooses between reporting 0 (and so dropping out the unit), X (skipping the unit), $F(X)$ (imitating a feedforward network at the unit) and $X + F(X)$ (imitating a residual network unit)(**f**).

5.2 Swapout

Notation and terminology: Capital letters are used to represent tensors and $\odot$ to represent element- wise product (broadcasted for scalars). Boldface **0** and **1** represent tensors of 0 and 1 respectively. A network block is a set of simple layers in some specific configuration e.g. a convolution followed by a ReLU or a residual network block He et al., 2015. Several such potentially different blocks can be connected in the form of a directed acyclic graph to form the full network model.

Dropout kills individual units randomly; stochastic depth skips entire blocks of units randomly. Swapout allows individual units to be dropped, or to skip blocks randomly. Implementing swapout is a straightforward generalization of dropout. Let X be the input to some network block that computes $F(X)$. The u'th unit produces $F^{(u)}(X)$ as output. Let Θ be a tensor of i.i.d. Bernoulli random variables. Dropout computes the output Y of that block as

$$Y = \Theta \odot F(X). \tag{5.1}$$

It is natural to think of dropout as randomly selecting an output from the set $\mathcal{F}^{(u)} = \{0, F^{(u)}(X)\}$ for the u'th unit.

Swapout generalizes dropout by expanding the choice of $\mathcal{F}^{(u)}$. Now write $\{\Theta_i\}$ for N distinct tensors of iid Bernoulli random variables indexed by i and with corresponding parameters $\{\theta_i\}$. Let $\{F_i\}$ be corresponding tensors consisting of values already computed somewhere in the network. Note that one of these F_i can be X itself (identity). However, F_i are not restricted to being a function of X and the X are dropped to indicate this. Most natural choices for F_i are the outputs of earlier layers. Swapout computes the output of the layer in question by computing

$$Y = \sum_{i=1}^{N} \Theta_i \odot F_i \tag{5.2}$$

and so, for unit u, $\mathcal{F}^{(u)} = \{F_1^{(u)}, F_2^{(u)}, \ldots, F_1^{(u)} + F_2^{(u)}, \ldots, \sum_i F_i^{(u)}\}$. This work studies the simplest case where

$$Y = \Theta_1 \odot X + \Theta_2 \odot F(X) \tag{5.3}$$

so that, for unit u, $\mathcal{F}^{(u)} = \{0, X^{(u)}, F^{(u)}(X), X^{(u)} + F^{(u)}(X)\}$. Thus, each unit in the layer could be:

1) dropped (choose 0);

2) a feedforward unit (choose $F^{(u)}(X)$);

3) skipped (choose $X^{(u)}$);

4) or a residual network unit (choose $X^{(u)} + F^{(u)}(X)$).

Since a swapout network can clearly imitate a residual network, and since residual networks are currently the best-performing networks on various standard benchmarks, this work performs exhaustive experimental comparisons with them.

If one accepts the view of dropout and stochastic depth as averaging over a set of architectures, then swapout extends the set of architectures used. Appropriate random choices of Θ_1 and Θ_2 yield: all architectures covered by dropout; all architectures covered by stochastic depth; and block level skip connections. But other choices yield unit level skip and residual connections.

Swapout retains important properties of dropout. Swapout discourages co-adaptation by dropping units, but also by on occasion presenting units with inputs that have come from earlier layers. Dropout has been shown to enhance the stability of stochastic gradient descent (Hardt, Recht, and Singer, 2015, lemma 4.4). This applies to swapout in its most general form, too. We extend the notation of that paper, and write L for a Lipschitz constant that applies to the network, $\nabla f(v)$ for the gradient of the network f with parameters v, and $D\nabla f(v)$ for the gradient of the dropped out version of the network.

The crucial point in the relevant lemma is that $\mathbb{E}[\| Df(v) \|] < \mathbb{E}[\| \nabla f(v) \|] \leq L$ (the inequality implies improvements). Now write $\nabla S\,[f]\,(v)$ for the gradient of a swapout network, and $\nabla G\,[f]\,(v)$ for the gradient of the swapout network which achieves the largest Lipschitz constant by choice of Θ_i (this exists, because Θ_i is discrete). First, a Lipschitz constant applies to this network; second, $\mathbb{E}[\| \nabla S\,[f]\,(v) \|] \leq \mathbb{E}[\| \nabla G\,[f]\,(v) \|] \leq L$, so swapout makes stability no worse; third, light conditions on f can be expected to provide $\mathbb{E}[\| \nabla S\,[f]\,(v) \|] < \mathbb{E}[\| \nabla G\,[f]\,(v) \|] \leq L$, improving stability (Hardt, Recht, and Singer, 2015, Section 4).

5.2.1 Inference in Stochastic Networks

A model trained with swapout represents an entire family of networks with tied parameters, where members of the family were sampled randomly during training. There are two options for inference. Either replace random variables with their expected values, as recommended by Srivastava et al. Srivastava et al., 2014 (deterministic inference). Alternatively, sample several members of the family at random, and average their predictions (stochastic inference). Note that such stochastic inference with dropout has been studied in Gal and Ghahramani, 2015

There is an important difference between swapout and dropout. In a dropout network, one can estimate expectations exactly (as long as the network isn't trained with batch normalization, below). This is because

$$\mathbb{E}[\text{ReLU}[\Theta \odot F(X)]] = \text{ReLU}[\mathbb{E}[\Theta \odot F(X)]]. \tag{5.4}$$

Recall Θ is a tensor of Bernoulli random variables, and thus non- negative.

In a swapout network, one usually can not estimate expectations exactly. The problem is that $\mathbb{E}[\text{ReLU}[(\Theta_1 X + \Theta_2 Y)]]$ is not the same as $\text{ReLU}[\mathbb{E}[(\Theta_1 X + \Theta_2 Y)]]$ in general. Estimates of expectations that ignore this are successful, as the experiments show, but stochastic inference gives significantly better results.

Srivastava et al. argue that deterministic inference is significantly less expensive in computation. It is likely that Srivastava et al. may have overestimated how many samples are required for an accurate average, because they use *distinct* dropout networks in the average (Figure 11 in Srivastava et al., 2014). Our experience of stochastic inference with swapout has been positive, with the number of samples needed for good behavior small (Figure 5.2). Furthermore, computational costs of inference are smaller when each instance of the network uses the same parameters

A technically more delicate point is that both dropout and swapout networks interact poorly with batch normalization *if* one uses deterministic inference. The problem is that the estimates collected by batch normalization during training may not reflect test time statistics. To see this consider two random variables X and Y and let $\Theta_1, \Theta_2 \sim Bernoulli(\theta)$. While $\mathbb{E}[\Theta_1 X + \Theta_2 Y] = \mathbb{E}[\theta X + \theta Y] = \theta X + \theta Y$, it can be shown that $\text{Var}[\Theta_1 X + \Theta_2 Y] \geq \text{Var}[\theta X + \theta Y]$ with equality holding only for $\theta = 0$ and $\theta = 1$. Thus, the variance estimates collected by Batch Normalization during training do not represent the statistics observed during testing if the expected values of Θ_1 and Θ_2 are used in a deterministic inference scheme. These errors in scale estimation accumulate as more and more layers are stacked. This may explain why Huang et al., 2016 reports that dropout doesn't lead to any improvement when used in residual networks with batch normalization.

5.2.2 Baseline Comparison Methods

Following is a concise description of main baselines.

ResNets: ResNet architectures as described in He et al., 2015(referred to as v1) and in He et al., 2016(referred to as v2).

Dropout: Standard dropout on the output of residual block, as below.

$$Y = \Theta \odot (X + F(X)) \tag{5.5}$$

Layer Dropout: We replace equation 5.3 by $Y = X + \Theta^{(1\times1)}F(X)$. Here $\Theta^{(1\times1)}$ is a single Bernoulli random variable shared across all units.

SkipForward: Equation 5.3 introduces two stochastic parameters Θ_1 and Θ_2. We also explore and compare with a simpler architecture, SkipForward, that introduces only one parameter but samples from a smaller set $\mathcal{F}^{(u)} = \{X^{(u)}, F^{(u)}(X)\}$ as below.

$$Y = \Theta \odot X + (1 - \Theta) \odot F(X) \tag{5.6}$$

5.3 Experiments

We experiment extensively on the CIFAR-10 dataset and demonstrate that a model trained with swapout outperforms a comparable ResNet model. Further, a 32 layer wider model matches the performance of a 1001 layer ResNet on both CIFAR-10 and CIFAR-100 datasets.

Model: We experiment with ResNet architectures as described in He et al., 2015(referred to as v1) and in He et al., 2016(referred to as v2). However, our implementation (referred to as ResNet Ours) has the following modifications which improve the performance of the original model (Table 5.1). Between blocks of different feature sizes we subsample using average pooling instead of strided convolutions and use projection shortcuts with learned parameters. For final prediction we follow a scheme similar to Network in Network Lin, Chen, and Yan, 2013. We replace average pooling and fully connected layer by a 1x1 convolution layer followed by global average pooling to predict the logits that are fed into the softmax.

Layers in ResNets are arranged in three groups with all convolutional layers in a group containing equal number of filters. We represent the number of filters in each group as a tuple with the smallest size as (16, 32, 64) (as used in He et al., 2015for CIFAR-10). We refer to this as *width* and experiment with various multiples of this base size represented as $W \times 1$, $W \times 2$ etc.

Training: We train using SGD with a batch size of 128, momentum of 0.9 and weight decay of 0.0001. Unless otherwise specified, we train all the models for a total 256 epochs. Starting from an initial learning rate of 0.1, we drop it by a factor of 10 after 196 epochs and then again after 224 epochs. We do the standard augmentation of left-right flips and random translations of up to four pixels. For translation, we pad the images by 4 pixels on all the sides and sample a random 32x32 crop. All the images in a mini-batch use the same crop. Note that dropout slows convergence (Srivastava et al., 2014, A.4), and swapout should do so too for similar reasons. Thus using the same training schedule for all the methods should *disadvantage* swapout.

Models trained with swapout consistently outperform baselines: Table 5.1 compares Swapout with various 20 layer baselines. Models trained with Swapout consistently outperform all other models of similar architecture.

The stochastic training schedule matters: Different layers in a swapout network could be trained with different parameters of their Bernoulli distributions (the stochastic training

Method	Width	#Params	Error(%)
ResNet v1 He et al., 2015	$W \times 1$	0.27M	8.75
ResNet v1 Ours	$W \times 1$	0.27M	8.54
Swapout v1	$W \times 1$	0.27M	**8.27**
ResNet v2 Ours	$W \times 1$	0.27M	8.27
Swapout v2	$W \times 1$	0.27M	**7.97**
Swapout v1	$W \times 2$	1.09M	6.58
ResNet v2 Ours	$W \times 2$	1.09M	6.54
Stochastic Depth v2 Ours	$W \times 2$	1.09M	5.99
Dropout v2	$W \times 2$	1.09M	5.87
SkipForward v2	$W \times 2$	1.09M	6.11
Swapout v2	$W \times 2$	1.09M	**5.68**

TABLE 5.1: In comparison with fair baselines on CIFAR-10, swapout is always more accurate. We refer to the base width of (16, 32, 64) as $W \times 1$ and others are multiples of it (See Table 5.3 for details on width). We report the width along with the number of parameters in each model. Models trained with swapout consistently outperform all other models of comparable architecture. All stochastic methods were trained using the Linear(1, 0.5) schedule (Table 5.2). v1 and v2 represent residual block architectures in He et al., 2015 and He et al., 2016 respectively.

schedule). Table 5.2 shows that different stochastic training schedules have a significant affect on the performance. We report the performance with deterministic as well as stochastic inference. These schedules differ in how the values of parameters θ_1 and θ_2 of the Bernoulli random variables in equation 5.3 are set for the different layers. Note that $\theta_1 = \theta_2 = 0.5$ corresponds to the maximum stochasticity. A schedule with less randomness in the early layers (bottom row) performs the best. This is expected because Swapout adds per unit noise and early layers have the largest number of units. Thus, low stochasticity in early layers significantly reduces the randomness in the system. We use this schedule for all the experiments unless otherwise stated.

Swapout improves over ResNet architecture: From Table 5.3 it is evident that networks trained with Swapout consistently show better performance than corresponding ResNets,

Method	Determin. Error(%)	Stoch. Error(%)
Swapout ($\theta_1 = \theta_2 = 0.5$)	10.36	6.69
Swapout ($\theta_1 = 0.2, \theta_2 = 0.8$)	10.14	7.63
Swapout ($\theta_1 = 0.8, \theta_2 = 0.2$)	7.58	6.56
Swapout ($\theta_1 = \theta_2 =$ Linear(0.5, 1))	7.34	6.52
Swapout ($\theta_1 = \theta_2 =$ Linear(1, 0.5))	**6.43**	**5.68**

TABLE 5.2: The choice of stochastic training schedule matters. We evaluate the performance of a 20 layer swapout model ($W \times 2$) trained with different stochasticity schedules on CIFAR-10. These schedules differ in how the parameters θ_1 and θ_2 of the Bernoulli random variables in equation 5.3 are set for the different layers. Linear(a, b) refers to linear interpolation from a to b from the first block to the last (see Huang et al., 2016). Others use the same value for all the blocks. We report the performance for both the deterministic and stochastic inference (with 30 samples). Schedule with less randomness in the early layers (bottom row) performs the best.

for most choices of width investigated, using just the deterministic inference. This difference indicates that the performance improvement is not just an ensemble effect.

Stochastic inference outperforms deterministic inference: Table 5.3 shows that the stochastic inference scheme outperforms the deterministic scheme in all the experiments. Prediction for each image is done by averaging the results of 30 stochastic forward passes. This difference is not just due to the widely reported effect that an ensemble of networks is better as networks in our ensemble share parameters. Instead, stochastic inference produces more accurate expectations and interacts better with batch normalization.

Stochastic inference needs few samples for a good estimate: Figure 5.2 shows the estimated accuracies as a function of the number of forward passes per image. It is evident that relatively few samples are enough for a good estimate of the mean. Compare Figure-11 of Srivastava et al., 2014, which implies ~ 50 samples are required.

Increase in width leads to considerable performance improvements: The number of filters in a convolutional layer is its width. Table 5.3 shows that the performance of a 20 layer model improves considerably as the width is increased both for the baseline ResNet

Model	Width	#Param	ResNet v2	Swapout	
				Deterministic	Stochastic
v2 (20) $W \times 1$	(16, 32, 64)	0.27M	8.27	8.58	7.92
v2 (20) $W \times 2$	(32, 64, 128)	1.09M	6.54	6.40	5.68
v2 (20) $W \times 4$	(64, 128, 256)	4.33M	5.62	5.43	5.09
v2 (32) $W \times 4$	(64, 128, 256)	7.43M	**5.23**	**4.97**	**4.76**

TABLE 5.3: Wider swapout models work better. We evaluate the effect of increasing the number of filters on CIFAR-10. ResNets (He et al., 2015) contain three groups of layers with all convolutional layers in a group containing equal number of filters. We indicate the number of filters in each group as a tuple and report the performance with deterministic as well as stochastic inference with 30 samples. For each size, model trained with Swapout outperforms the corresponding ResNet model.

v2 architecture as well as the models trained with Swapout. Swapout is better able to use the available capacity than the corresponding ResNet with similar architecture and number of parameters. Table 5.4 compares models trained with Swapout with other approaches on CIFAR-10 while Table 5.5 compares on CIFAR-100. On both datasets our shallower but wider model compares well with 1001 layer ResNet model.

Swapout uses parameters efficiently: Persistently over tables 5.1, 5.3, and 5.4, swapout models with fewer parameters outperform other comparable models. For example, Swapout v2(32) $W \times 4$ gets 4.76% error with 7.43M parameters in comparison to the ResNet version at 4.91% with 10.2M parameters.

Experiments on CIFAR-100 confirm our results: Table 5.5 shows that Swapout is very effective as it improves the performance of a 20 layer model (ResNet Ours) by more than 2%. Widening the network and reducing the stochasticity leads to further improvements. Further, a wider but relatively shallow model trained with Swapout (22.72%; 7.46M params) is competitive with the best performing, very deep (1001 layer) latest ResNet model (22.71%;10.2M params).

Method	#Params	Error(%)
DropConnect Wan et al., 2013	-	9.32
NIN Lin, Chen, and Yan, 2013	-	8.81
FitNet(19) Romero et al., 2015	-	8.39
DSN Lee et al., 2015	-	7.97
HighwaySrivastava, Greff, and Schmidhuber, 2015	-	7.60
ResNet v1(110) He et al., 2015	1.7M	6.41
Stochastic Depth v1(1202) Huang et al., 2016	19.4M	4.91
SwapOut v1(20) $W \times 2$	1.09M	6.58
ResNet v2 (1001) He et al., 2016	10.2M	4.92
SwapOut v2(32) $W \times 4$	7.43M	**4.76**

TABLE 5.4: Swapout outperforms comparable methods on CIFAR-10. Note that a 32 layer wider model performs competitively in comparison to a 1001 layer ResNet model.

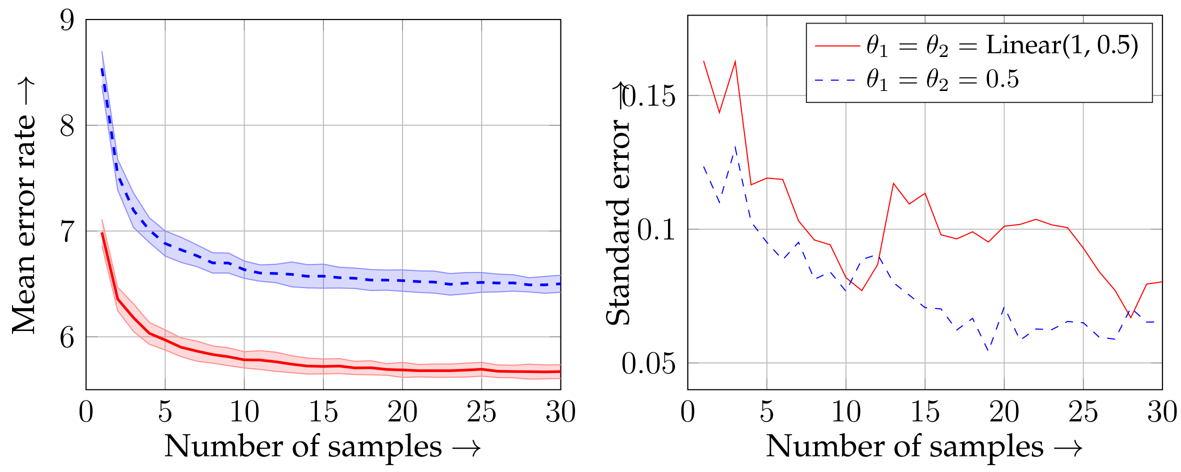


FIGURE 5.2: Stochastic inference needs few samples for a good estimate. We plot the mean error rate on the left as a function of the number of samples for two stochastic training schedules. Standard error of the mean is shown as the shaded interval on the left and magnified in the right plot. It is evident that relatively few samples are needed for a reliable estimate of the mean error. The mean and standard error was computed using 30 repetitions for each sample count. Note that stochastic inference quickly overtakes accuracies for deterministic inference in very few samples (2-3)(Table 5.2).

5.4 Discussion and Future Work

Swapout is a stochastic training method that shows reliable improvements in performance and leads to networks that use parameters efficiently. Relatively shallow swapout networks

Method	#Params	Error(%)
NIN Lin, Chen, and Yan, 2013	-	35.68
DSN Lee et al., 2015	-	34.57
FitNet Romero et al., 2015	-	35.04
Highway Srivastava, Greff, and Schmidhuber, 2015	-	32.39
ResNet v1 (110) He et al., 2015	1.7M	27.22
Stochastic Depth v1 (110) Huang et al., 2016	1.7M	24.58
ResNet v2 (164) He et al., 2016	1.7M	24.33
ResNet v2 (1001) He et al., 2016	10.2M	**22.71**
ResNet v2 Ours (20) $W \times 2$	1.09M	28.08
SwapOut v2 (20)(Linear(1,0.5)) $W \times 2$	1.10M	25.86
SwapOut v2 (56)(Linear(1,0.5)) $W \times 2$	3.43M	24.86
SwapOut v2 (56)(Linear(1,0.8)) $W \times 2$	3.43M	23.46
SwapOut v2 (32)(Linear(1,0.8)) $W \times 4$	7.46M	**22.72**

TABLE 5.5: Swapout is strongly competitive with the best methods on CIFAR-100, and uses parameters efficiently in comparison. A 20 layer model (Swapout v2 (20)) trained with Swapout improves upon the corresponding 20 layer ResNet model (ResNet v2 Ours (20)). Further, a 32 layer wider but much shallower model performs competitively in comparison to a 1001 layer ResNet model (last row).

give comparable performance to extremely deep residual networks.

We have shown that different stochastic training schedules produce different behaviors, but have not searched for the best schedule in any systematic way. It may be possible to obtain improvements by doing so. We have described an extremely general swapout mechanism. It is straightforward using equation 5.2 to apply swapout to inception networks Szegedy et al., 2014 (by using several different functions of the input and a sufficiently general form of convolution); to recurrent convolutional networks Pinheiro and Collobert, 2013 (by choosing F_i to have the form $F \circ F \circ F \ldots$); and to gated networks. All our experiments focus on comparisons to residual networks because these are the current top performers on CIFAR-10 and CIFAR-100. It would be interesting to experiment with other versions of the method.

As with dropout and batch normalization, it is difficult to give a crisp explanation of why swapout works. We believe that our results support the idea that swapout causes some form of improvement in the optimization process. This is because relatively shallow networks with swapout reliably work as well as or better than quite deep alternatives; and because swapout is notably and reliably more efficient in its use of parameters than comparable deeper networks. Unlike dropout, swapout will often propagate gradients while still forcing units not to co-adapt. Furthermore, our swapout networks involve some form of tying between layers. When a unit sometimes sees layer i and sometimes layer $i - j$, the gradient signal will be exploited to encourage the two layers to behave similarly. The reason swapout is successful likely involves both of these points.

Chapter 6

Improved Landmark Localization with Swapout

With the advent of high performing new architectures such as Residual Networks (He et al., 2015; He et al., 2016) and powerful training methods such as Swapout (Singh, Hoiem, and Forsyth, 2016), performance on existing tasks can be improved by simply moving to the new architectures and training methods. We posit that these new architectures make discovery of rare but important features in images easier. Further, stochastic training methods are able to better regularize the models leading to impressive gains.

In this chapter we draw parallels between the residual network architecture and the one described in chapter 3. With the insights from this connection we simplify the network architecture proposed in 3, bringing it closer to a residual network architecture. With the hypothesis that residual architectures can better discover rare features, we remove the intermediate losses as well. In addition to exploiting the strength of residual network architecture, it allows us to use Swapout to train a higher performing model.

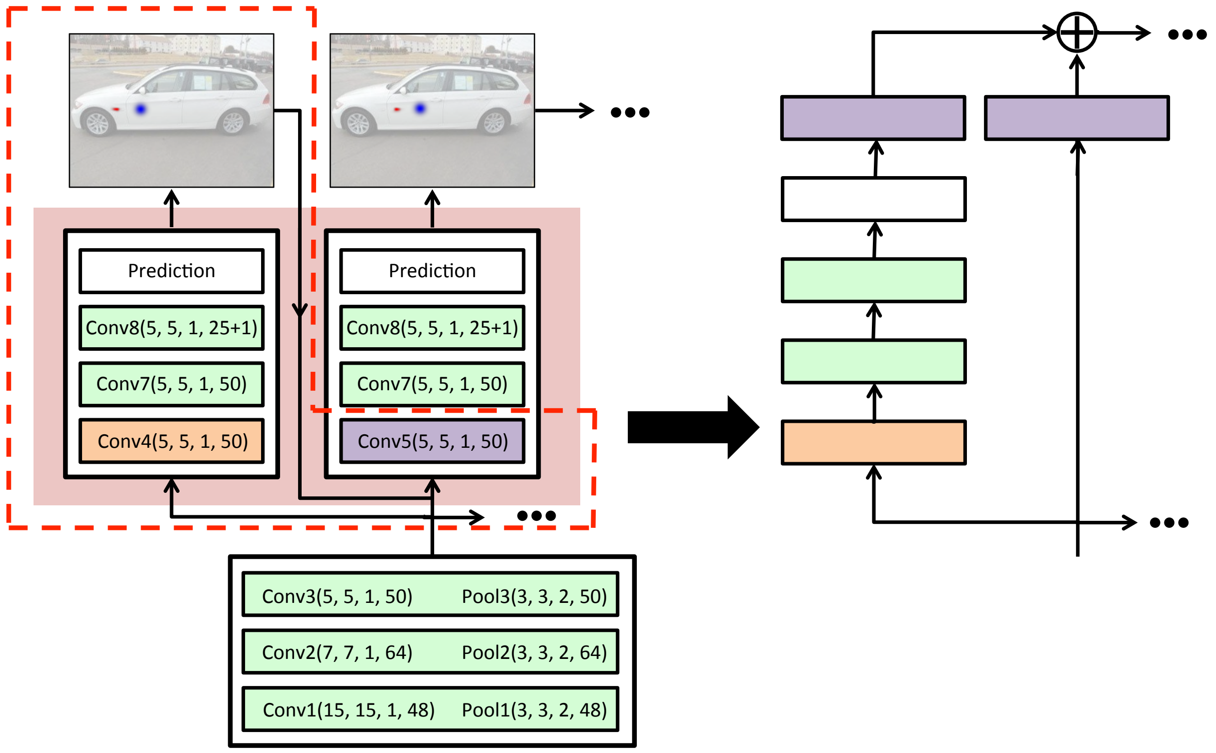


FIGURE 6.1: Relation of the model described in chapter 3 with residual network architecture. Rearrangement of blocks makes shortcut connections more apparent. These shortcut connections are akin to parameterized shortcuts described in He et al., 2015.

6.1 Approach

We revisit the architecture described in chapter 3 and view it in the light of residual architectures. We draw connections with the residual architectures and motivate the design of model described in this chapter.

6.1.1 Relation with Residual Architecture

First note that for a given matrix $W = [W_1 W_2]$ and vector $x^T = [x_1^T x_2^T]$, where W_1, W_2, x_1, x_2 are of appropriate sizes

$$Wx = W_1 x_1 + W_2 x_2 \tag{6.1}$$

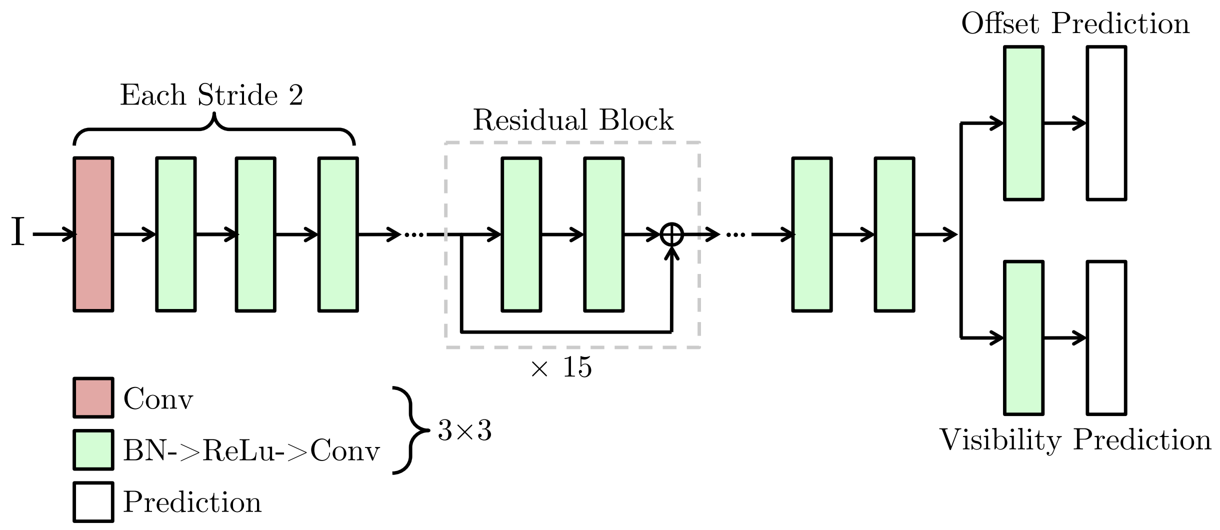


FIGURE 6.2: Residual architecture based model. Red box denotes a convolutional layer while green boxes denote batch normalization followed by ReLU followed by convolution. While box is the offset prediction layer. All convolutional layers use 3 × 3 filters. Model uses 15 residual blocks. These residual blocks are replaced by corresponding swapout blocks when train with swapout.

Similar argument is valid for convolution as well. Thus, for any two given feature maps, a concatenation followed by a convolution can be expressed as individual convolutions followed by an addition. With this, the architecture expressed in figure 3.2 of Chapter 3 can be expressed as an architecture that has a resemblance to a residual architecture, but with bigger residual blocks, shared weights and parameterized shortcuts. Figure 6.1 visualizes a part of the model described in chapter 3 to make this analogy apparent. This is not entirely surprising as model in chapter 3 is quite similar to a recurrent model, but with some step specific weights, while some recent works, e.g. Liao and Poggio, 2016, have discussed the connections between ResNets and RNNs.

6.2 Model

With the parallel described in previous section, we redefine the architecture as a stack of residual blocks with two convolutional layers each. A prediction block, same as the one used in section 3.2.2 is used to make per location predictions at the top. These individual predictions are then averaged using predicted confidences. Figure 6.2 shows a visualization of the model. For swapout training, each residual block is replaced by a swapout block.

6.2.1 Training and Implementation Details

We use 15 residual blocks and train the model using swapout through back- propagation using stochastic gradient descent with momentum. We initialize the weights using the method suggested by Glorot and Bengio, 2010. Dataset is augmented by including left-right flips, random color augmentation (Deng et al., 2009), random scaling, cropping and rotation. Images are scaled such that longest side is smaller than 500 pixels. Random cropping is done by cropping the border randomly such that all visible landmarks are always included. Random rotation is done in the range of $[-40, 40]$ degrees. A mini-batch of 20 images is constructed by randomly placing the images in a 500×500 pixel grid. A momentum of 0.9 is used along with a learning rate schedule of 0.1 for first 400 epochs, 0.01 for next 150 and 0.001 for next 50. For swapout, a linear stochastic schedule, Linear(1.0,0.8), is used.

6.3 Experiments

We evaluate our method on the task of predicting landmarks for humans on the Leeds Sports dataset. Provided train and test split are used. We first present ablation experiments that study the effect of swapout training. We then present a comparison with state of the art. Finally, we compare all the methods presented in this thesis.

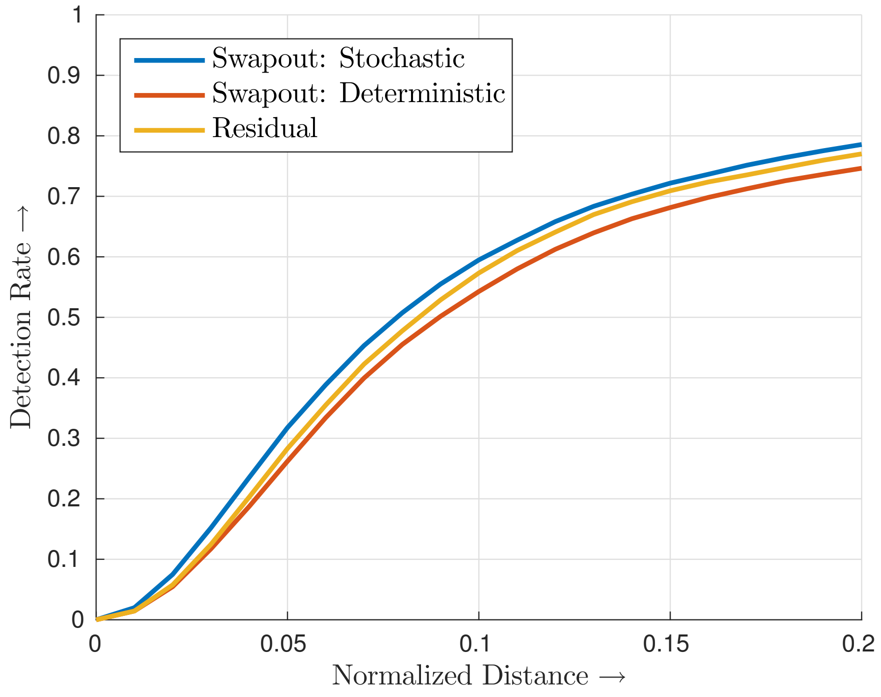
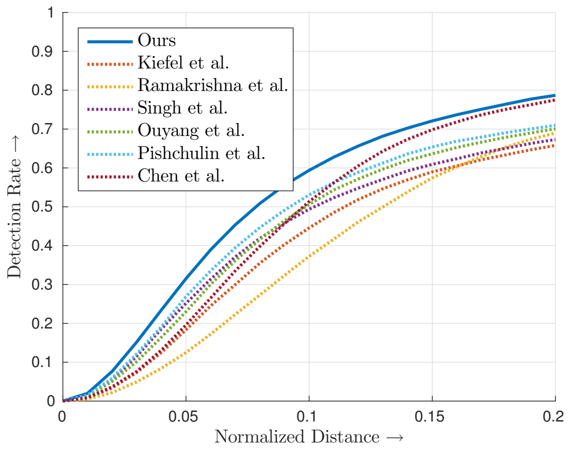


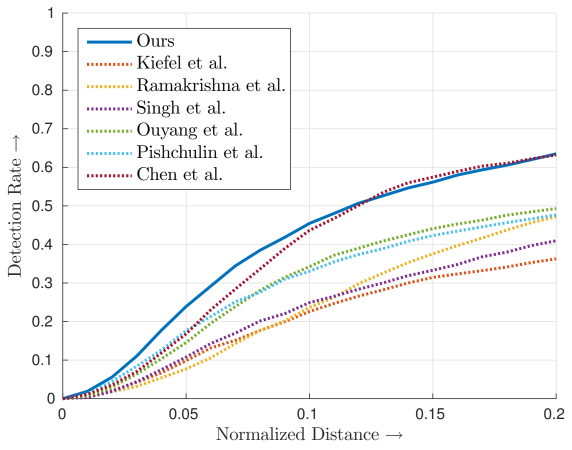
FIGURE 6.3: A model trained with swapout outperforms the corresponding residual model using stochastic inference. However, deterministic inference under-performs residual model. This may be because regression requires precise prediction and poor interaction between batch normalization and stochastic training yields imprecise predictions.

6.3.1 Swapout

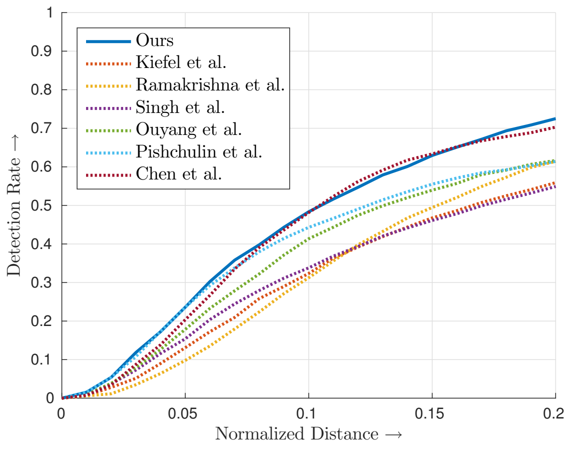
Figure 6.3 shows that a model trained with swapout outperforms the corresponding residual model using stochastic inference. Figure plots the evaluation using PCK measure. Interestingly, deterministic inference in the same model under-performs the residual model. This may be due to the fact the regression requires precise prediction and poor interaction between batch normalization with stochastic training results in higher errors for deterministic inference.



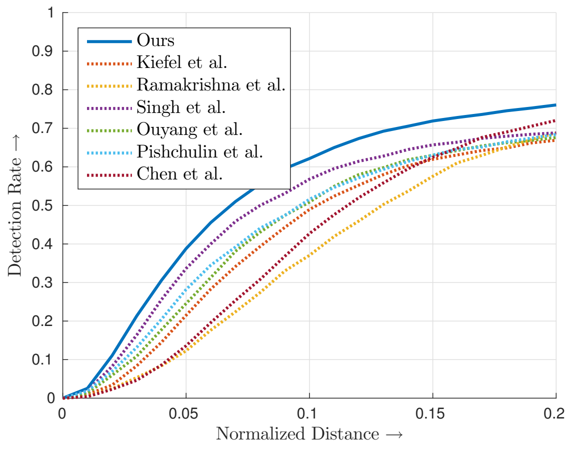
(A) all



(B) wrist



(C) elbow



(D) ankle

FIGURE 6.4: Our method outperforms several leading approaches for human landmark detection on LSP. The figure compares various approaches using the PCK evaluation metric for (A) all, (B) wrist, (C) elbow and, (D) ankle

6.3.2 Comparison with State of the Art

Figure 6.4 shows the evaluation for all the landmarks using the PCK measure. Our swapout based method outperforms several recent works. Note that a model trained with intermediate losses performed similar to the purely residual network based model and was thus excluded from the results.

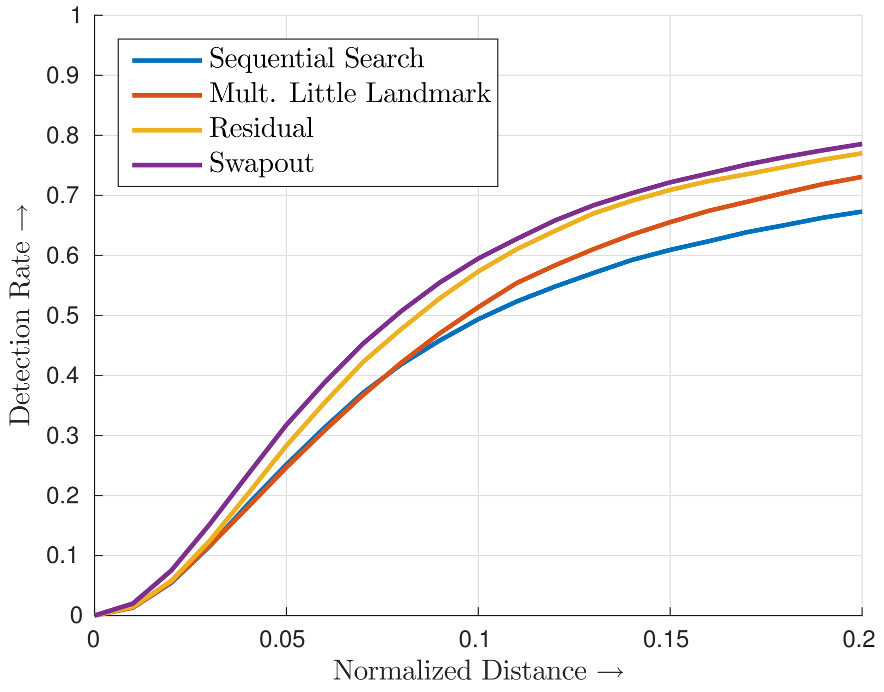


FIGURE 6.5: For all landmarks (on Leeds Sports Dataset), model trained with swapout (using stochastic inference) performs the best, outperforming other methods described in this thesis.

6.3.3 Comparison with Methods in this Thesis

Figure 6.5 compares the performance of various methods for all landmarks, while figure 6.6 compares the performance for the *left wrist* localization. The model trained with swapout performs the best and displays significant improvement in wrist localization over other methods.

6.4 Conclusion

Adoption of residual architecture further improves the performance of the landmark localization while simplifying the model. It seems to be the case that newer architectures are better at capturing rare features obviating the need for intermediate losses that encourage their discovery.

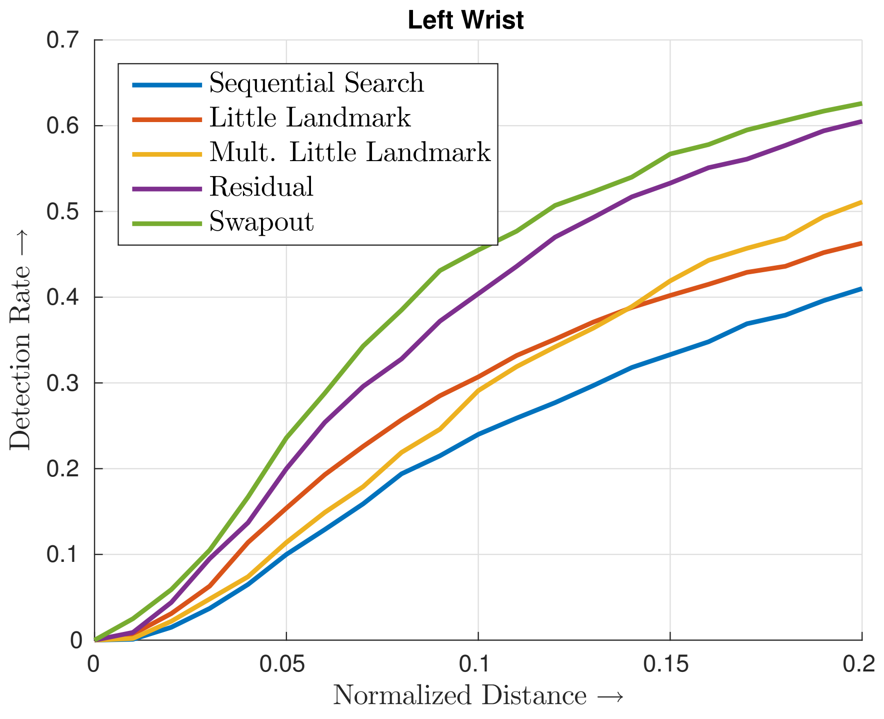


FIGURE 6.6: For left wrist (on Leeds Sports Dataset), the model trained with swapout using stochastic inference provides a significant improvement over other methods discussed in earlier chapters.

Chapter 7

Conclusions

Modeling and reasoning about the context is helpful for landmark localization. Chapter 2 utilized the context provided by easy to detect landmarks to help localize the harder ones. Chapter 3 discovered latent landmarks to help find little landmarks.

Explicit modeling of spatial relations, e.g. using spring based models, between landmarks may not be necessary to achieve good performance. None of the methods described in this thesis utilized explicit modeling but were able to achieve performance close to state of the art.

Residual architectures perform well for landmark localization tasks. Chapter 6 achieved strong performance using a residual architecture without any pretraining or additional data. Note that Leeds Sports Dataset is relatively small consisting of 1000 training images while exhibiting significant pose variation.

Swapout training improves the performance of residual architectures for classification as well as landmark localization. Chapter 5 demonstrated this for the classification task while Chapter 6 presented results for the landmark localization task.

Bibliography

Andriluka, Mykhaylo, Stephan Roth, and Bernt Schiele (2009). "Pictorial Structures Revisited: People Detection and Articulated Pose Estimation". In: *CVPR*.

Ba, Jimmy, Volodymyr Mnih, and Koray Kavukcuoglu (2014). "Multiple object recognition with visual attention". In: *arXiv preprint arXiv:1412.7755*.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.

Barto, Andrew G (1998). *Reinforcement learning: An introduction*. MIT press.

Bengio, Yoshua and Olivier Delalleau (2011). "On the Expressive Power of Deep Architectures". In: *Proceedings of the 22nd International Conference on Algorithmic Learning Theory*.

Bourdev, L. and J. Malik (2009). "Poselets: Body part detectors training using 3d human pose annotations". In: *ICCV*.

Caicedo, Juan and Svetlana Lazebnik (2015). "Semantic Guidance of Visual Attention for Localizing Objects in Scenes". In: *ICCV*.

Carreira, Joao et al. (2015). "Human Pose Estimation with Iterative Error Feedback". In: *arXiv preprint arXiv:1507.06550*.

Chatfield, K. et al. (2014). "Return of the Devil in the Details: Delving Deep into Convolutional Nets". In: *British Machine Vision Conference*. arXiv: 1405.3531 [cs].

Chen, Xianjie and Alan Yuille (2014). "Articulated Pose Estimation by a Graphical Model with Image Dependent Pairwise Relations". In: arXiv: 1407.3399.

Crandall, D., P. Felzenswalb, and D. Huttenlocher (2005). "Spatial Priors for Part-based Recognition using Statistical Models". In: *CVPR*.

Csurka, G. et al. (2004). "Visual categorization with bags of keypoints". In: *Workshop on Statistical Learning in Computer Vision*.

Dantone, M. et al. (2013). "Human Pose Estimation from Still Images using Body Parts Dependent Joint Regressors". In: *CVPR*. to appear. IEEE.

Daumé III, Hal and Daniel Marcu (2005). "Learning as search optimization: Approximate large margin methods for structured prediction". In: *ICML*. ACM.

Deng, J. et al. (2009). "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR*.

Doersch, Carl et al. (2012). "What Makes Paris Look like Paris?" In: *ACM Transactions on Graphics (SIGGRAPH)* 31.4.

Dollár, P. et al. (2009). "Integral Channel Features". In: *BMVC*.

Eichner, M. and V. Ferrari (2009). "Better Appearance Models for Pictorial Structures". In: *ICCV*.

Eichner, Marcin and Vittorio Ferrari (2012). "Appearance Sharing for Collective Human Pose Estimation". In: *ACCV*.

Fei-Fei, L., R. Fergus, and P. Perona (2006). "One-Shot learning of object categories". In: 28.4, pp. 594–611.

Felzenszwalb, Pedro F., Ross B. Girshick, and David McAllester (2010). "Cascade Object Detection with Deformable Part Models". In: *CVPR*.

Felzenszwalb, Pedro F. and Daniel P. Huttenlocher (2005). "Pictorial Structures for Object Recognition". In: *IJCV* 61.1, pp. 55–79.

Felzenszwalb, P.F. et al. (2010). "Object Detection with Discriminatively Trained Part-Based Models". In: *IEEE Trans. on Pattern Analysis and Machine Intelligence*.

Fergus, R., P. Perona, and A. Zisserman (2003). "Object Class Recognition by Unsupervised Scale-Invariant Learning". In: *CVPR*.

— (2005). "A Sparse Object Category Model for Efficient Learning and Exhaustive Recognition". In: *CVPR*.

Ferrari, Vittorio, Manuel Marín-Jiménez, and Andrew Zisserman (2008). "Progressive Search Space Reduction for Human Pose Estimation". In: *CVPR*.

Gal, Yarin and Zoubin Ghahramani (2015). "Bayesian convolutional neural networks with Bernoulli approximate variational inference". In: arXiv: 1506.02158.

Girshick, Ross et al. (2014). "Rich feature hierarchies for accurate object detection a nd semantic segmentation". In: *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, pp. 580–587.

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *International conference on artificial intelligence and statistics*, pp. 249–256.

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio (2011). "Deep sparse rectifier neural networks". In: *AISTATS*.

Grauman, Kristen and Bastian Leibe. "Part-Based Category Models". In: *Visual Recognition*. URL: `http://www.cs.utexas.edu/~grauman/courses/fall2009/papers/partbased.pdf`.

Hardt, Moritz, Benjamin Recht, and Yoram Singer (2015). "Train faster, generalize better: Stability of stochastic gradient descent". In: arXiv: `1509.01240`.

He, Kaiming et al. (2015). "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385.

— (2016). "Identity Mappings in Deep Residual Networks". In: *CoRR* abs/1603.05027.

Huang, Gao et al. (2016). "Deep Networks with Stochastic Depth". In: arXiv: `1603.09382`.

Insafutdinov, Eldar et al. (2016). "DeeperCut: A Deeper, Stronger, and Faster Multi-Person Pose Estimation Model". In: arXiv: `1605.03170`.

Ioffe, Sergey and David Forsyth (2001). "Human Tracking with Mixtures of Trees". In: *ICCV*.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: arXiv: `1502.03167`.

Johnson, Sam and Mark Everingham (2010). "Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation". In: *BMVC*. doi:10.5244/C.24.12.

Juneja, M. et al. (2013). "Blocks That Shout: Distinctive Parts for Scene Classification". In: *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*.

Karlinsky, Leonid et al. (2010). "The chains model for detecting parts by their context". In: *CVPR*, pp. 25–32.

Kiefel, Martin and Peter Vincent Gehler (2014). "Human Pose Estimation with Fields of Parts". In: *ECCV*. Springer.

Krause, Jan et al. (2013). "3d object representations for fine-grained categorization". In: *ICCVW*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *NIPS*.

Larochelle, Hugo and Geoffrey E Hinton (2010). "Learning to combine foveal glimpses with a third-order Boltzmann machine". In: *Advances in neural information processing systems*, pp. 1243–1251.

LeCun, Yann et al. (1998). “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE*.

Lee, Chen-Yu et al. (2015). “Deeply-supervised nets”. In: *AISTATS*.

Leibe, B., E. Seemann, and B. Schiele (2005). “Pedestrian Detection in Crowded Scenes”. In: *CVPR*, I: 878–885.

Leung, T.K., M.C. Burl, and P. Perona (1995). “Finding faces in cluttered scenes using random labelled graph matching”. In: *ICCV*.

Liao, Qianli and Tomaso Poggio (2016). “Bridging the Gaps Between Residual Learning, Recurrent Neural Networks and Visual Cortex”. In: arXiv: 1604.03640.

Lin, Min, Qiang Chen, and Shuicheng Yan (2013). “Network In Network”. In: arXiv: 1312.4400.

Liu, Jiongxin and Peter N Belhumeur (2013). “Bird part localization using exemplar-based models with enforced pose and subcategory consistency”. In: *ICCV*.

Liu, Jiongxin, Yinxiao Li, and Peter N Belhumeur (2014). “Part-Pair Representation for Part Localization”. In: *ECCV 2014*.

Mnih, Volodymyr, Nicolas Heess, Alex Graves, et al. (2014). “Recurrent models of visual attention”. In: *Advances in Neural Information Processing Systems*, pp. 2204–2212.

Mori, Greg et al. (2004). “Recovering Human Body Configuration: Combining Segmentation and Recognition”. In: *CVPR*. Vol. 2, pp. 326–333.

Nair, Vinod and Geoffrey E Hinton (2010). “Rectified linear units improve restricted boltzmann machines”. In: *ICML*.

Ouyang, Wanli, Xiao Chu, and Xiaogang Wang (2014). “Multi-source deep learning for human pose estimation”. In: *CVPR*. IEEE.

Parizi, Sobhan Naderi et al. (2014). “Automatic Discovery and Optimization of Parts for Image Classification”. In: *ICLR*.

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (2012). “On the difficulty of training recurrent neural networks”. In: arXiv: 1211.5063.

Pfister, Tomas, James Charles, and Andrew Zisserman (2015). “Flowing convnets for human pose estimation in videos”. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1913–1921.

Pinheiro, Pedro HO and Ronan Collobert (2013). “Recurrent convolutional neural networks for scene parsing”. In: arXiv: 1306.2795.

Pishchulin, Leonid et al. (2013a). “Poselet conditioned pictorial structures”. In: *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, pp. 588–595.

— (2013b). “Strong Appearance and Expressive Spatial Models for Human Pose Estimation”. In: *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, pp. 3487–3494.

Pishchulin, Leonid et al. (2015). “DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation”. In: arXiv: 1511.06645.

Rahimi, Ali and Benjamin Recht (2007). “Random features for large-scale kernel machines”. In: *NIPS*.

Ramakrishna, Varun et al. (2014). “Pose Machines: Articulated Pose Estimation via Inference Machines”. In: *ECCV*.

Ramanan, Deva (2006). “Learning to parse images of articulated bodies”. In: *NIPS*. Vol. 19.

Ratliff, Nathan D, David Silver, and J Andrew Bagnell (2009). “Learning to search: Functional gradient techniques for imitation learning”. In: *Autonomous Robots*.

Romero, Adriana et al. (2015). “Fitnets: Hints for thin deep nets”. In: *ICLR*.

Sapp, Benjamin, Alexander Toshev, and Ben Taskar (2010). “Cascaded Models for Articulated Pose Estimation”. In: *ECCV*.

Shih, Kevin J et al. (2015). “Part Localization using Multi-Proposal Consensus for Fine-Grained Categorization”. In: *arXiv preprint arXiv:1507.06332*.

Simonyan, K. and A. Zisserman (2014). “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: arXiv: 1409.1556.

Singh, Saurabh, Abhinav Gupta, and Alexei A. Efros (2012). “Unsupervised Discovery of Mid-level Discriminative Patches”. In: *ECCV*.

Singh, Saurabh, Derek Hoiem, and David Forsyth (2015). “Learning a Sequential Search for Landmarks”. In: *Computer Vision and Pattern Recognition*. URL: http://vision.cs.uiuc.edu/projects/lssland/.

— (2016). “Swapout: Learning an ensemble of deep architectures”. In: arXiv: 1605.06465.

Srivastava, Nitish et al. (2014). “Dropout: A simple way to prevent neural networks from overfitting”. In: *The Journal of Machine Learning Research*.

Srivastava, Rupesh K, Klaus Greff, and Jürgen Schmidhuber (2015). “Training very deep networks”. In: *NIPS*.

Su, Yu and Frédéric Jurie (2012). “Improving image classification using semantic attributes”. In: *International journal of computer vision* 100.1, pp. 59–77.

Szegedy, Christian et al. (2014). “Going Deeper with Convolutions”. In: arXiv: 1409.4842.

Tompson, Jonathan et al. (2014). “Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation”. In: *CoRR* abs/1406.2984. URL: http://arxiv.org/abs/1406.2984.

Tompson, Jonathan et al. (2015). “Efficient object localization using convolutional networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 648–656.

Toshev, Alexander and Christian Szegedy (2013). “Deeppose: Human pose estimation via deep neural networks”. In: *arXiv preprint arXiv:1312.4659*.

Tran, Duan and David Forsyth (2010). “Improved Human Parsing with a Full Relational Model”. In: *ECCV*.

Tu, Zhuowen (2008). “Auto-context and its application to high-level vision tasks”. In: *CVPR*.

Tu, Zhuowen and Xiang Bai (2010). “Auto-Context and Its Application to High-Level Vision Tasks and 3D Brain Image Segmentation”. In: *PAMI* 32.10, pp. 1744–1757.

Wah, Catherine et al. (2011). “The caltech-ucsd birds-200-2011 dataset”. In: CNS-TR-2010-001.

Wan, Li et al. (2013). “Regularization of neural networks using dropconnect”. In: *ICML*, pp. 1058–1066.

Wang, Yang and Greg Mori (2008). “Multiple Tree Models for Occlusion and Spatial Constraints in Human Pose Estimation”. In: *ECCV*.

Wang, Yang, Duan Tran, and Zicheng Liao (2011). “Learning Hierarchical Poselets for Human Parsing”. In: *CVPR*.

Wei, Shih-En et al. (2016). “Convolutional Pose Machines”. In: arXiv: 1602.00134.

Welinder, P. et al. (2010). *Caltech-UCSD Birds 200*. Tech. rep. CNS-TR-2010-001. California Institute of Technology.

Xu, Kelvin et al. (2015). “Show, attend and tell: Neural image caption generation with visual attention”. In: arXiv: 1502.03044.

Yang, Yi and Deva Ramanan (2013). “Articulated human detection with flexible mixtures of parts”. In: *PAMI*.

Zeiler, Matthew D and Rob Fergus (2013). “Stochastic pooling for regularization of deep convolutional neural networks”. In: arXiv: 1301.3557.

Zhang, Yuting et al. (2015). “Improving object detection with deep convolutional networks via bayesian optimization and structured prediction”. In: arXiv: 1504.03293.