

Multiscale Conditional Random Fields for Machine Vision

by

David Duvenaud

B. Sc. Hons., University of Manitoba, 2006

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES
(Computer Science)

The University Of British Columbia
(Vancouver)

July 2010

© David Duvenaud, 2010

Abstract

We develop a single joint model which can classify images and label super-pixels, based on tree-structured conditional random fields (CRFs) derived from a hierarchical image segmentation, extending previous work by Reynolds and Murphy, and Plath and Toussaint. We show how to train this model in a weakly-supervised fashion, in which some of the images only have captions specifying which objects are present; this information is propagated down the tree and thus provides weakly labeled data at the leaves, which can be used to improve the performance of the super-pixel classifiers. After training, information can be propagated from the super-pixels up to the root-level image classifier (although this does not seem to help in practice compared to just using root-level features). We compare two kinds of tree: the standard one with pairwise potentials, and one based on noisy-or potentials, which better matches the semantics of the recursive partitioning used to create the tree. However, we do not find any significant difference between the two.

Table of Contents

Abstract	ii
Table of Contents	iii
List of Figures	vi
Acknowledgments	ix
1 Introduction	1
1.1 Motivation	1
1.2 Image Models	1
1.3 Multi-scale Approaches	2
1.3.1 Previous Approaches to Multiscale CRFs	3
1.3.2 Our Approach	4
1.3.3 Under-segmentation vs Over-segmentation	5
1.4 Performance Metrics	5
1.5 Pascal VOC Dataset	7
1.5.1 Image Features	7
1.6 Supervised and Semi-Supervised Learning	7
1.7 Thesis Contributions	9
2 Pairwise Trees and Learning	10
2.1 Model Semantics	10
2.1.1 Local Evidence Potentials	10
2.1.2 Independent Model	11
2.1.3 Pairwise Potentials	11

2.1.4	Regularization of Image Feature Weights	12
2.1.5	Regularization of Pairwise Potentials	12
2.2	Likelihood	12
2.3	Learning	13
2.3.1	Computational Issues	14
2.4	Experimental Setup	15
2.4.1	Cross-validation	15
2.4.2	Evaluation Criteria	15
2.5	Results	16
2.6	Discussion	16
3	Noisy-Or Factors	20
3.1	Motivation for Noisy-Or Tree Models	20
3.2	Definition	20
3.3	Likelihood	21
3.3.1	Computing Expected Complete Likelihood	21
3.4	Evidence Flow	22
3.5	Training	25
3.5.1	Learning the Noisy-Or Failure Rate Parameter	26
3.6	Results	26
4	Model Comparison	30
4.1	Comparing Models	30
4.1.1	Independent Model	30
4.2	Performance Measures	30
4.3	Performance Comparison	31
4.3.1	Pixel-level Accuracy	31
4.3.2	Global-level Accuracy	31
4.3.3	Remedies	36
4.4	Qualitative Evaluation	37
4.5	Improving Performance	37
4.6	Introducing an Oracle	39
5	Conclusions and Future Work	41

5.1	Future Work	41
5.1.1	Structure-Adaptive Potentials	41
5.1.2	Bounding Box Data	43
5.1.3	Combined Grid and Tree Structures	43
5.1.4	Joint Learning Over All Classes	44
5.1.5	Large-Scale Experiments	44
5.2	Concluding Remarks	44
	Bibliography	46

List of Figures

Figure 1.1	An example of a tree-structured CRF built on an exact recursive segmentation of an image.	3
Figure 1.2	An example of a CRF defined for the presence of the sheep class over multiple scales of a recursively segmented image.	4
Figure 1.3	The maximum attainable accuracy given the segmentation used.	6
Figure 1.4	Example multi-scale segmentations from the VOC 2008 dataset. Rows one to four: Image segmentation at progressively finer levels of detail. Bottom row: Pixel-level class labels.	8
Figure 2.1	Pixel-level test accuracy on the tree model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.	16
Figure 2.2	Global-level test accuracy on the tree model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.	17
Figure 2.3	Change in pixel-level test accuracy after training with partially labeled data.	18
Figure 2.4	Change in global-level test accuracy after training with partially labeled data.	18
Figure 2.5	A plot of the change in test error versus the pixel-level accuracy on the test set after supervised training, when the global node was set to the true value.	19

Figure 3.1	An example of belief propagation. Left: A situation in which there is no local evidence for a class being present, except in one leaf node. Middle: Marginals after BP in a pairwise tree. Right: Marginals after BP in a noisy-or tree.	22
Figure 3.2	Left: A situation in which there is local evidence in two adjacent leaf nodes. Middle: Marginals after BP in a pairwise tree. Right: Marginals after BP in a noisy-or tree.	23
Figure 3.3	Left: A situation in which there is strong evidence at the global scale, and weak local evidence at one of the leaf nodes. Middle: Marginals after BP in a pairwise tree. Right: Marginals after BP in a noisy-or tree.	23
Figure 3.4	An example of belief propagation and evidence flow in a noisy-or tree, trained on real data. Node size is proportional to probability.	24
Figure 3.5	An example of belief propagation and evidence flow in the pairwise tree model, trained on real data. Node size is proportional to probability.	24
Figure 3.6	The segmentation of the image used in figures 3.5 and 3.4. . .	25
Figure 3.7	Pixel-level test accuracy on the noisy-or model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.	27
Figure 3.8	Global-level test accuracy on the noisy-or model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.	28
Figure 3.9	Change in test pixel-level accuracy after training with partially labeled data.	29
Figure 3.10	Change in test global-level test accuracy after training with partially labeled data.	29
Figure 4.1	Pixel-level test accuracy on the three models, with and without semi-supervised training.	32
Figure 4.2	Global-level test accuracy on the three models, with and without semi-supervised training.	33

Figure 4.3	Pixel-level test accuracy across all models.	34
Figure 4.4	Global-level test accuracy across all models.	34
Figure 4.5	Mean test cross entropy over all nodes across all models. Lower is better.	35
Figure 4.6	Left: Histogram of the number of neighbours of each node in the training set. Nodes with one neighbour are necessarily leaves. Right: Histogram of the number of neighbours of global-level nodes in the training set.	35
Figure 4.7	Mean cross-entropy per node versus the number of neighbours of a node on the training set.	36
Figure 4.8	Detecting a dog. Top left: Original image. Top center: Segmentation at bottom level. Top right: True pixel labels. Bottom left: Pixel probabilities for independent model. Bottom center: Pixel probabilities for pairwise model. Bottom right: Pixel probabilities for noisy-or model.	38
Figure 4.9	Detecting a person. Top left: Original image. Top center: Segmentation at bottom level. Top right: True pixel labels. Bottom left: Pixel probabilities for independent model. Bottom center: Pixel probabilities for pairwise model. Bottom right: Pixel probabilities for noisy-or model.	39
Figure 4.10	Pixel-level test accuracy across all models, including the case where the global-level nodes were clamped to their true value.	40
Figure 5.1	Learned α_g versus the number of neighbours.	42

Acknowledgments

There are many people who made my time at UBC both instructive and fruitful. First and foremost: my advisor Kevin Murphy, by giving his support, as well as by his patience in steering my ideas towards useful research. I'd like to thank Nando de Freitas for his many helpful ideas. I'd like to profusely thank Benjamin Marlin for sharing the recursive segmentation and feature extraction code, as well as for being an example to me.

I'd like to thank Kevin Swersky and Bo Chen for raising the bar, Mark Schmidt for showing me not to be afraid to start at square one, and Emtiyaz Khan for helping me push through to the end.

Chapter 1

Introduction

1.1 Motivation

A central problem in learning object localization models is that pixel-labeled images are rare and costly to produce. In contrast, data that have only weak labeling information, such as captions, are relatively abundant. In the extreme case, one may wish to train a model on a massive set of images drawn from an automated search. This is essentially the data collector's dream: To generate a training dataset for a cat detector by simply performing a web image search for 'cat'.

Such datasets can easily be used to train an image classifier. However, in order to train a model to perform object localization on such data, we require a unified model of object presence at both the scale of the whole image as well as the scale of individual super-pixels.

1.2 Image Models

For the tasks of image classification and object localization, we are interested in predicting whether a given image region contains an example of a given class c somewhere within it. For the tasks of image classification or object detection, the region of interest is the entire image. For the task of image segmentation or object localization, the regions of interest are on the scale of pixels or super-pixels.

We can cast both of these tasks as inference problems in the following way. We

define random variables Y_1, Y_2, \dots, Y_n denoting the presence or absence of class c in image regions $1 \dots n$. We are then interested in computing $P(Y_1, Y_2, \dots, Y_n | x)$ where x are the image features. We may then assume some structure on the conditional probability $P(Y_1, Y_2, \dots, Y_n | x)$ in order to make learning and inference tractable. Such structured conditional models are called Conditional Random Fields (CRFs).

In this work, we further develop the use of CRFs which are structured in such a way as to combine evidence from multiple scales of segmentation. This model structure allows us to perform semi-supervised learning to take advantage of weakly labeled data as described above. There are two additional motivations for this model structure. First, it can effectively incorporate evidence from other image classifiers to perform better localization on unlabeled images (as shown in Chapter 4). Second, it can combine evidence from multiple scales and image locations in a simple, consistent way, allowing one model to do both classification and localization.

Figure 1.1 shows an example of a multi-scale CRF defined over multiple scales of an image.

1.3 Multi-scale Approaches

Recent work on image classification and segmentation has shown that incorporating evidence from multiple scales is an effective strategy for image classification [15] [19] [20]. This can be explained by the fact that labels at different scales must agree with each other to some degree. Evidence about segments at one level of detail can help in classifying segments at neighbouring levels of detail. For example, knowing that there is a person in a small segment of the image means that there is also a person in any larger segment containing the first segment.

This observation motivates the idea of combining evidence at multiple scales of the image by performing segmentation at different scales, estimating a class' presence or absence separately for each segment, and then combining these local estimates with a conditional random field.

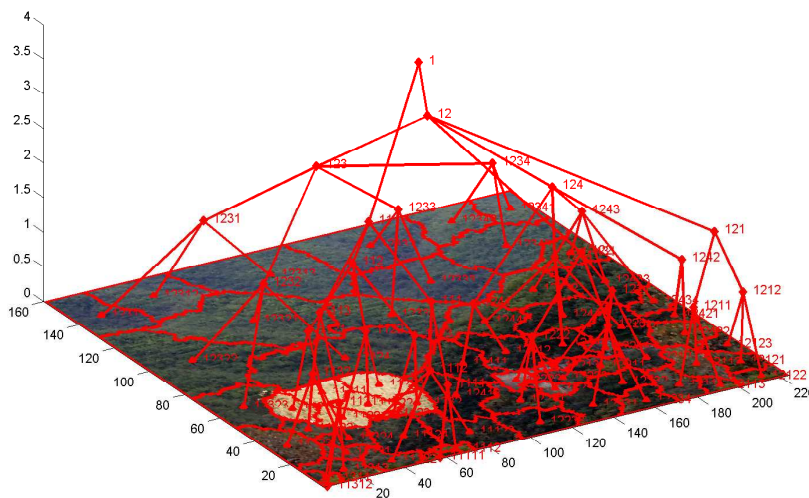


Figure 1.1: An example of a tree-structured CRF built on an exact recursive segmentation of an image.

1.3.1 Previous Approaches to Multiscale CRFs

The construction of multi-scale CRFs was previously demonstrated in [19] and [20]. In these models, each image was segmented at multiple scales, and each segment was assigned a label node designating whether the class of interest was contained in that segment. A node at a given level of detail was connected to the node at the next coarser level of detail according to the degree of overlap between their corresponding image regions. This procedure results in a tree-structured CRF with one image-level node representing the presence or absence of a class in the image, as well as one node for each segment at the finest level.

In these works, the image segmentations at different levels of detail were constructed independently of one another. This approach has the advantage that the segmentations at each layer can be constructed independently, but has the disadvantage that the segments at one level of detail may not significantly overlap with

algorithm.

As in previous work, we constructed the CRF by connecting nodes representing regions with maximal overlap. In our case, however, each node is completely contained in exactly one other node by design. We denote the containing region to be a parent region, and the nodes contained within it to be the children of that region.

One benefit of constructing a recursive image segmentation is that the resulting label structure allows the use of factors joining parent and child nodes that have stricter agreement constraints than a typical pairwise-factored CRF. In Chapter 3, we explore the use of such “noisy-or” factors in tree-structured CRFs.

1.3.3 Under-segmentation vs Over-segmentation

Constructing this recursive segmentation raises the question: What level of detail is appropriate for the finest level of segmentation?

The cost of over-segmentation is that our CRF will be unnecessarily large, slowing down inference. However, inference in this model is linear in the number of nodes in the CRF.

The cost of under-segmentation is that the bottom-layer segments will contain pixels from classes other than the task of interest. This will put an upper bound on the accuracy of our model’s pixel-level labels.

In the experiments performed in this thesis, we truncated the recursive segmentation at four levels of recursion, leaving approximately 30 segments per image at the finest level of detail. Figure 1.3 shows the maximum attainable VOC accuracy (defined below) for this four-layer deep recursive segmentation is 50.2%. For comparison, the best-performing algorithm on the PASCAL VOC 2008 segmentation challenge achieved a mean accuracy of 25.4%.

1.4 Performance Metrics

Performance is measured for both tasks by the accuracy a as defined in the VOC 2008 challenge as

$$a = \frac{tp}{tp + fp + fn} \quad (4.1)$$

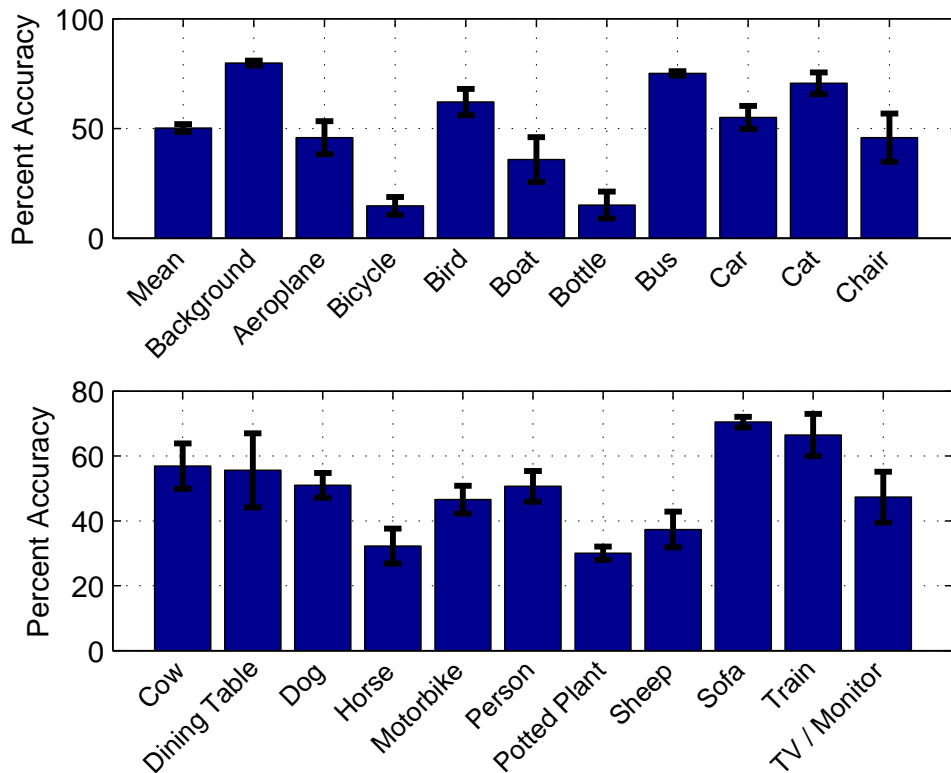


Figure 1.3: The maximum attainable accuracy given the segmentation used.

where tp , fp , and fn mean true positive, false positive, and false negative, respectively [19]. True positive is the number of foreground image pixels that are correctly predicted. False positive is the number of background pixels that are incorrectly predicted. False negative is the number of foreground pixels that are incorrectly predicted. Here, “foreground” refers to the parts of the image containing the class of interest.

Note that this performance measure effectively ignores true negatives, which generally comprise the majority of all predictions for this task. This measure is useful, but is quite different from the objective function maximized by our learning algorithms.

1.5 Pascal VOC Dataset

The pixel-labeled data used for these experiments were gathered from the training and validation sets in the PASCAL Visual Object Classes(VOC) Challenge 2008 dataset [5]. This dataset contains approximately 1000 images, each approximately 500x500 pixels, in which each pixel is either assigned to one of 20 classes, assigned to the “background” class, or labeled as “don’t care”. The predicted labels of pixels labeled “don’t care” do not count towards the accuracy score.

1.5.1 Image Features

The image features used were: colour histograms (100 dimensions), histogram of oriented gradients (200 dimensions) [2], textons (500 dimensions) [31], and the 5x5 discretized location of the segment (25 dimensions). With a bias term added, each feature vector had 826 dimensions. However, the model is somewhat agnostic to the image features computed, and allows the use of different feature vectors at different levels in the segmentation hierarchy. For instance, one may want to use GIST [17] features at the highest spatial scale, as in [16]. However, exploratory experiments did not show a significant different in performance by replacing the top level features with a GIST vector.

1.6 Supervised and Semi-Supervised Learning

We conducted two sets of experiments: In the first, models were trained using pixel-labeled training data. Given an image that is fully labeled at the pixel level, we can compute the labels for regions at all spatial scales. Thus, this phase of learning is fully supervised.

In the second set of experiments, the pixel-labeled training data was augmented with an equal amount of “caption-only” data, which was unlabeled at every level of the hierarchy except at the global level. Data labeled in this way corresponds to knowing only whether or not a certain type of object is found anywhere in the image. Because all but one node per image is unlabeled, we call this regime “semi-supervised” training. Here belief propagation is used to compute the expected sufficient statistics of the unlabeled nodes conditioned on the global label, which can then be used to train parameters at finer spatial scales.



Figure 1.4: Example multi-scale segmentations from the VOC 2008 dataset. Rows one to four: Image segmentation at progressively finer levels of detail. Bottom row: Pixel-level class labels.

1.7 Thesis Contributions

The main contribution of this thesis is in demonstrating a concrete way in which caption-only data can be used to train a localization model. In doing so, we develop several refinements to the class of multi-scale image CRFs:

- We develop the use of recursive image segmentation.
- Using the recursive segmentation, we introduce the noisy-or structured tree model.
- In contrast to previous efforts[16][19][20], which first learn local image patch classifiers and then connect them with a CRF, we show how to learn the parameters of the local classifiers in concert, as a structured prediction problem.

Chapter 2

Pairwise Trees and Learning

In this chapter we precisely define a tree-structured conditional random field with pairwise potentials. We give formulas for the likelihood, show how learning can be done via the Expectation-Maximization algorithm, and show results on fully observed and semi-supervised learning.

2.1 Model Semantics

The image segments at all levels of detail can be denoted by $S^{(r)}$, for some integer r . The model contains one label node $Y_c^{(r)}$ for each class c and each element of the recursive image partition $S^{(r)}$. Setting $Y_c^{(r)} = 1$ is interpreted as meaning that the image region defined by segment $S^{(r)}$ contains part of an object from class c , while setting $Y_c^{(r)} = 0$ is interpreted as meaning that the image region $S^{(r)}$ does not contain part of an object from class c .

2.1.1 Local Evidence Potentials

The local evidence log-potential for node $Y_c^{(r)}$ in this model depends linearly on the feature vectors $x^{(r)}$ for the region $S^{(r)}$. We define the local evidence log-potential in Equation 1.1 where W_c^l are the feature-to-label weights for class c and segmentation level l .

$$\phi_f(y_c^{(r)}, x^{(r)}) = y_c^{(r)} (x^{(r)})^T W_c^l \quad (1.1)$$

Weights are shared across all of nodes in a given level of detail l of the segmentation tree within each object class. In the experiments below, the weight vectors W_c^2 and W_c^3 were also constrained to be equal.

2.1.2 Independent Model

As a baseline, we can consider an image model consisting solely of these local evidence potentials. In this “independent” model, every region label is predicted separately, and the model becomes equivalent to a per-region logistic regression on each region’s image features. The likelihood of a node label assignment y is as follows:

$$P(Y = y|x) = \frac{1}{Z} \prod_{l=1}^L \prod_{(i) \in N_l} \exp(\phi_f(y^{(i)}, x^{(i)})) \quad (1.2)$$

Here L is the number of layers in the tree, and $(i) \in N_L$ denotes the nodes in layer L . The parameters of the independent model can be trained on only fully-labeled data.

2.1.3 Pairwise Potentials

In Equation 1.3 we define the pairwise potential between neighbouring nodes. The pairwise potentials depend only on a 2x2 table of parameters θ , indexed by values taken by the nodes at each end of the potential.

$$\phi_{pair}(y_c^{(r_1)}, y_c^{(r_2)}) = \theta(y_c^{(r_1)}, y_c^{(r_2)}) \quad (1.3)$$

In our experiments, three sets of pairwise parameters were learned: One set for the potentials connecting global nodes to their children, one for the potentials connecting nodes in the middle layers to their children, and a third for the potentials

connecting middle-layer nodes to bottom-layer nodes.

2.1.4 Regularization of Image Feature Weights

Learning for the weights of the local evidence log-potentials is done with a simple L_2 regularizer λ on the image feature weights W . In order to avoid choosing a separate hyperparameter for each weightgroup, we choose one regularization setting, kept fixed across different weightgroups. This opens up the issue of how to scale the regularization as the number of training examples changes.

The different weightgroups have a widely varying number of training instances. For instance, the nodes at the bottom (pixel-level) of the trees will have many more training examples as the nodes at the top (global-level), since each image will have only one top-level node and many bottom-level nodes. Following a Bayesian interpretation of regularized optimization as MAP estimation, we view the regularizer as a prior, and do not scale it with the number of training examples. This can be interpreted as giving the same prior to each of the weight groups' parameters, and then conditioning on varying amounts of evidence.

2.1.5 Regularization of Pairwise Potentials

In these experiments, no regularization was used for the pairwise potential parameters, as it was unclear how to scale this regularization with respect to the regularization of the node potentials. Using a separate hyperparameter would have increased the computational cost of cross-validation significantly. However, this was not expected to cause significant over-fitting, since in each model, only three sets of four pairwise potentials were learned.

Note that if the pairwise links are regularized separately from the node potential parameters, strong enough regularization of the pairwise potential parameters effectively makes the pairwise tree model equivalent to the independent model.

2.2 Likelihood

The likelihood of observing a particular configuration of label nodes y given feature vector x is defined as:

$$P(Y = y|x) = \frac{1}{Z} \prod_{l=1}^L \prod_{(i) \in N_l} \exp\left(\phi_f(y^{(i)}, x^{(i)}) + \phi_{pair}(y^{(i)}, y^{parent(i)})\right) \quad (2.4)$$

Here $parent(i)$ denotes the index of parent of node i . As a special case, the root has no parent node, and $\phi_{pair} = 0$.

2.3 Learning

First, let y^{mis} denotes the missing labels while y^{obs} denotes the observed labels. The marginal probability of observing y^{obs} can be obtained by summing out over all joint configurations of the missing labels y^{mis} .

$$P(y^{obs}|x) = \sum_{y^{mis}} P(y^{obs}, y^{mis}|x) \quad (3.5)$$

We also define the posterior probability of the missing labels given the observed labels:

$$P(y^{mis}|y^{obs}, x) = \frac{P(y^{mis}, y^{obs}|x)}{P(y^{obs}|x)} \quad (3.6)$$

The expected complete log-likelihood over all training examples is as follows:

$$E[-\mathcal{L}] = \sum_{n=1}^N \sum_{y_n^{mis}} P(y_n^{mis}|y_n^{obs}, x_n) \log\left(P(y_n^{mis}, y_n^{obs}|x_n)\right) \quad (3.7)$$

We now show how the gradient of the expected complete log likelihood with respect to the feature weights W_c^l can be computed using these two quantities (the L_2 regularization term is omitted for clarity):

$$\begin{aligned}
\frac{\partial E[\mathcal{L}]}{\partial W_c^l} &= \sum_{n=1}^N \sum_{y_n^{mis}} P(y_n^{mis} | y_n^{obs}, x_n) \sum_{(i) \in N_l} \left(\frac{\partial \phi_f(y_{cn}^{(i)}, x_n^{(i)})}{\partial W_c^l} - \sum_{y'} P(y' | x_n) \frac{\partial \phi_f(y_c'^{(i)}, x_n^{(i)})}{\partial W_c^l} \right) \\
&= \sum_{n=1}^N \sum_{(i) \in N_l} \left(E_{P(y_{cn}^{(i)} | y_n^{obs}, x_n)} [y_{cn}^{(i)} x_n^{(i)}] - E_{P(y_c'^{(i)} | x_n)} [y_c'^{(i)} x_n^{(i)}] \right) \quad (3.8)
\end{aligned}$$

$$= \sum_{n=1}^N \sum_{(i) \in N_l} \left(P(y_{cn}^{(i)} | y_n^{obs}, x_n) - P(y_c'^{(i)} | x_n) \right) x_n^{(i)} \quad (3.9)$$

Here N is the number of training examples, and N_l is the number of nodes in layer l of example N . To apply L_2 regularization, we add to the derivative the term $-2\lambda \|W_c^l\|_2$.

We find that the gradient is proportional to the difference between the marginals computed after clamping all observed nodes to their true values, and the marginals computed with nodes observed. We can obtain exact marginals in time linear in the number of nodes by using belief propagation [18].

The gradient of the likelihood with respect to the pairwise parameters $\theta_c(a, b)$ has a similar form:

$$\begin{aligned}
\frac{\partial E[\mathcal{L}]}{\partial \theta_c(a, b)} &= \sum_{n=1}^N \sum_{(i) \in N_L} \sum_{(j) \in N_L} \left(E_{P(y_{cn}^{(i)}, y_{cn}^{(j)} | y_n^{obs}, x_n)} \mathbb{I}[y_{cn}^{(i)} = a, y_{cn}^{(j)} = b] \right. \\
&\quad \left. - E_{P(y_c'^{(i)}, y_c'^{(j)} | x_n)} \mathbb{I}[y_c'^{(i)} = a, y_c'^{(j)} = b] \right) \quad (3.10)
\end{aligned}$$

$$= \sum_{n=1}^N \sum_{(i) \in N_L} \sum_{(j) \in N_L} \left(P(y_{cn}^{(i)} = a, y_{cn}^{(j)} = b | y_n^{obs}, x_n) - P(y_{cn}^{(i)} = a, y_{cn}^{(j)} = b | x_n) \right) \quad (3.11)$$

The gradient of $\theta_c(a, b)$ is simply the expected difference in the number of times we would observe $[y_c = a, y_c = b]$ between the clamped versus the unclamped distributions.

2.3.1 Computational Issues

Learning was broken into three stages as follows:

1. The image feature weights W , initialized to zero, were trained in the independent model by supervised training on fully labeled training images.
2. The pairwise factors ϕ_{pair} were added to the CRF, and the feature weights W along with the pairwise parameters θ were learned by supervised training on the fully labeled training examples.
3. Caption-only data was added to the dataset, and the model was trained in a semi-supervised way using the E-M algorithm shown above.¹

2.4 Experimental Setup

In these experiments, we balanced the dataset for each class separately by removing approximately 80% of images that did not contain the class of interest.

2.4.1 Cross-validation

Error bars depicting one standard error were produced by conducting experiments on five training/test splits of the data. Within each split, the L_2 regularization parameter λ was chosen by nested cross-validation: Each training set was split into five inner training/validation splits. For both the supervised case and the semi-supervised case, the setting of λ that had the best average accuracy on the validation set was chosen to train the model on the whole training set for that fold.

Each outer fold had 400 fully-labeled training examples, 400 caption-only training examples, and 200 test examples.

2.4.2 Evaluation Criteria

At test time, for each example, posterior marginals are computed for each node in the tree conditional on the image feature vectors. Node marginals at the top and bottom of the trees were thresholded at 0.5 to produce the final pixel-level and image-level classifications, respectively.²

¹ The function minimizer used in the M step was *minFunc* by Mark Schmidt, which implements the L-BFGS algorithm. This software is available at <http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html>

²Note that pixel-level accuracy can be improved by allowing node marginals near 0.5 to remain unlabeled. On this dataset, the best improvement in accuracy comes from labeling only those nodes

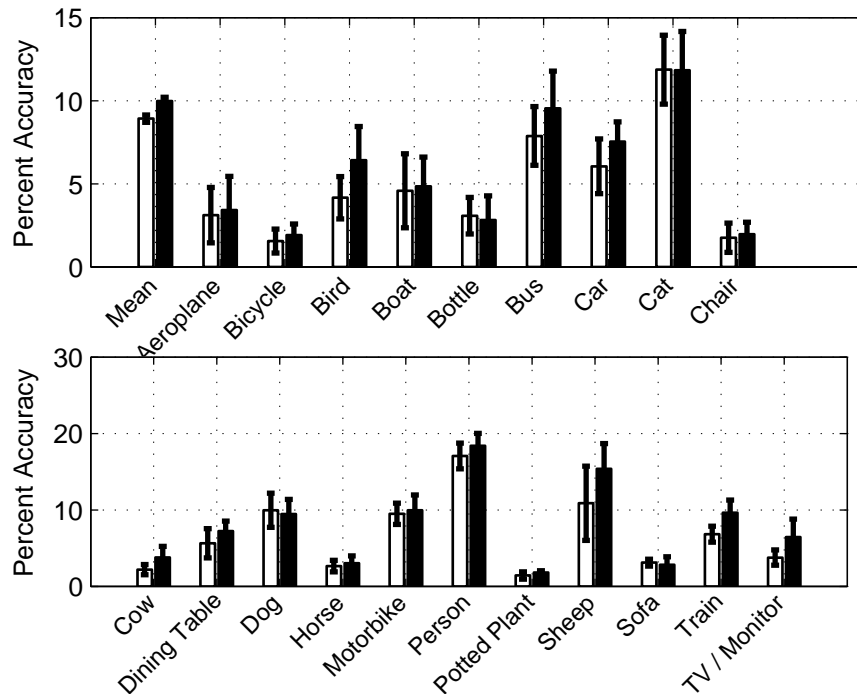


Figure 2.1: Pixel-level test accuracy on the tree model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.

2.5 Results

Figures 2.1, 2.4, show the pixel-level and global-level accuracy after both supervised and semi-supervised training. In these results, “accuracy” refers to the VOC accuracy metric defined in Chapter 1, and the “absolute percentage change” represents the percent accuracy after semi-supervised training minus the percent accuracy before semi-supervised training.

2.6 Discussion

The mean improvement in accuracy after semi-supervised training is statistically significant, but varies significantly between classes. Besides noise, how can we explain the decrease in performance in some classes? Note that while the accuracy

with marginals less than 0.75 or greater than 0.25.

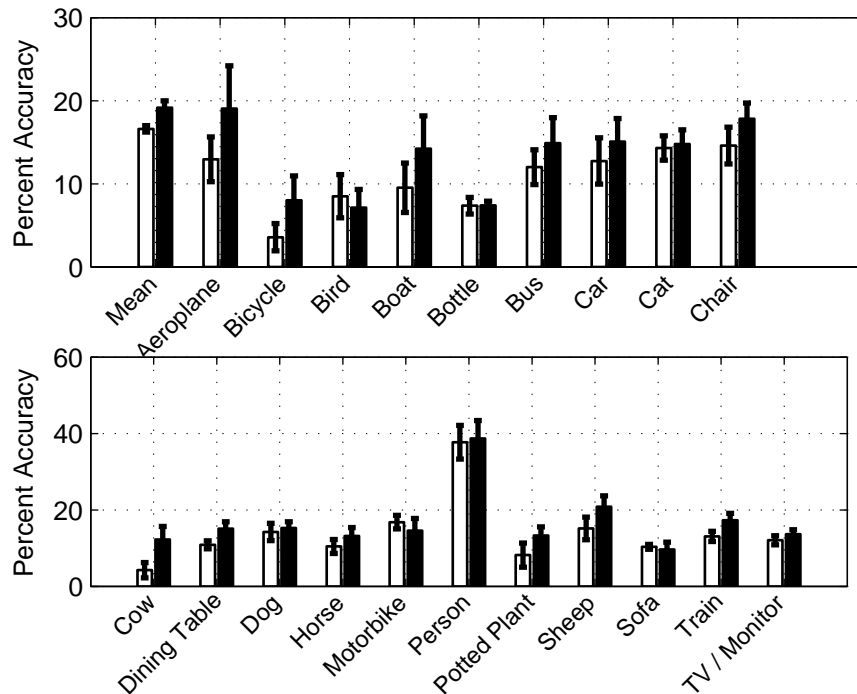


Figure 2.2: Global-level test accuracy on the tree model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.

did not always improve, the accuracy measure used here is quite different from the training objective. As well, in order for the pixel-level accuracy to improve, the model must already be somewhat competent at predicting pixel-level labels given the global label. This is because the weight gradient is a function of the difference in node marginals between the case where the global-level node is unobserved, and the case where it is observed. If the bottom-level marginals do not change significantly (or in the correct direction) when the global-level node is clamped, the model weights will not change.

Figure 2.5 supports this explanation. Plotted is the change in pixel-level accuracy after semi-supervised training, versus the pixel-level accuracy after supervised training when the top-level node was clamped to the true value. This figure shows that the pixel-level accuracy did not improve in classes which had poor localization performance after supervised training. We can also see that the best-localized

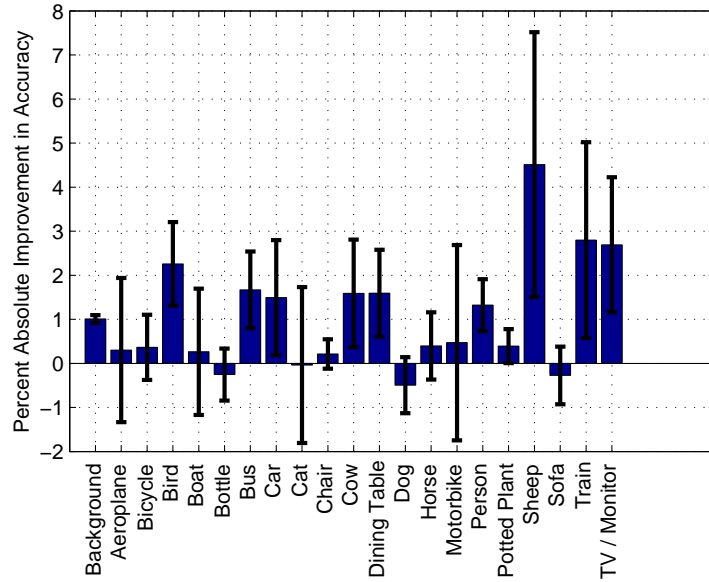


Figure 2.3: Change in pixel-level test accuracy after training with partially labeled data.

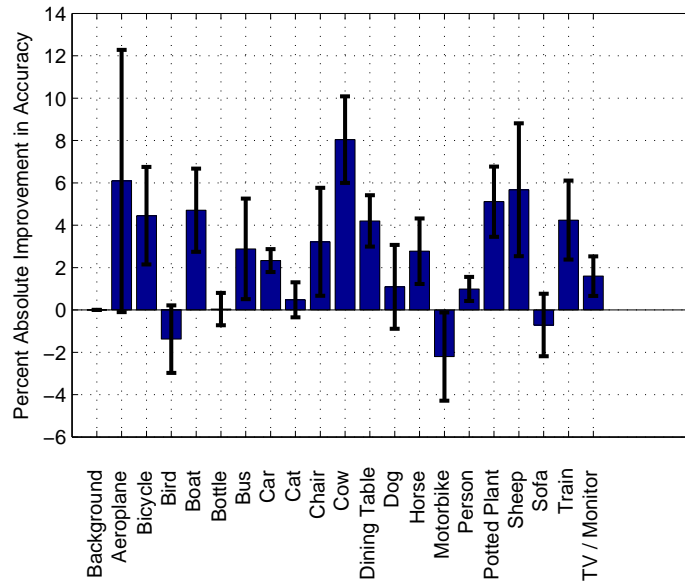


Figure 2.4: Change in global-level test accuracy after training with partially labeled data.

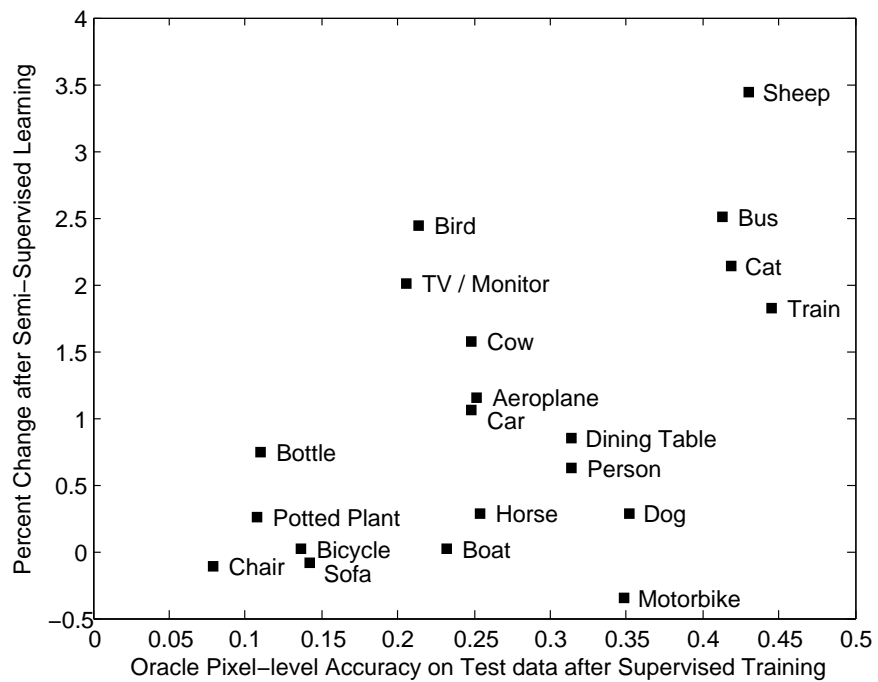


Figure 2.5: A plot of the change in test error versus the pixel-level accuracy on the test set after supervised training, when the global node was set to the true value.

classes also enjoyed the biggest improvement in accuracy from semi-supervised training.

We leave a discussion of global-level accuracy until Chapter 4.

Chapter 3

Noisy-Or Factors

3.1 Motivation for Noisy-Or Tree Models

If one was to learn a fully-parameterized factor over a group of child nodes and their common parent, what sort of factor would we learn? If we had segmented the image recursively, then we would observe a parent node to be on if and only if at least one child node was on. Thus the maximum likelihood solution for a factor joining parents and children would be one that put probability mass only on states where a parent was on if and only if any child was on.

This factor would have the same semantics of a logical OR-gate, and, as noted by [18], its probabilistic analogue, the noisy-or factor, has many desirable properties.

3.2 Definition

The noisy-or factor has the following semantics: The parent¹ node y_p turns on with probability θ independently for each child y^i that is turned on, where i ranges from 1 to g , the number of children of y_p . Thus the noisy-or log-potential can be defined as:

¹Here we are using “parent” and “child” to denote relative position in the image segmentation, not in the sense of a Directed Acyclic Graph.

$$\phi_{no}(y_p, y^1, \dots, y^g) = y_p \log \left(1 - \prod_{i=1}^g (1 - \theta)^{y^i} \right) + (1 - y_p) \sum_{i=1}^g y_i \log(1 - \theta) \quad (2.1)$$

In a form that is easier to read, we can replace the success rate θ with the failure rate $q = 1 - \theta$:

$$\exp(\phi_{no}(y_p, y^1, \dots, y^g)) = \left[1 - \prod_{i=1}^g q^{y^i} \right]^{y_p} \left[\prod_{i=1}^g q^{y^i} \right]^{(1-y_p)} \quad (2.2)$$

As shown in [18], messages for a noisy-or factor can be computed in time linear in the number of children in the factor, giving belief propagation in this model the same time complexity as the pairwise model.

3.3 Likelihood

The likelihood of the noisy-or model is similar to that of the pairwise model. Essentially, each set of pairwise potentials between a parent and all of its children is replaced by one noisy-or factor:

$$P(Y = y|x) = \frac{1}{Z} \prod_{l=1}^L \prod_{(i) \in N_l} \exp \left(\phi_f(y^{(i)}, x^{(i)}) + \phi_{no}(y^{(i)}, y^{children(i)}) \right) \quad (3.3)$$

$$(3.4)$$

Where, as a special case, nodes at the bottom layer of the tree have no children, and $\phi_{no} = 0$.

3.3.1 Computing Expected Complete Likelihood

To compute the expected complete likelihood of the noisy-or factors conditioned on local evidence at each of the child nodes, we could simply use the normal junction-tree algorithm. However, using this method, we must sum over all possible states of each group of parents and children. The length of this sum is exponential in

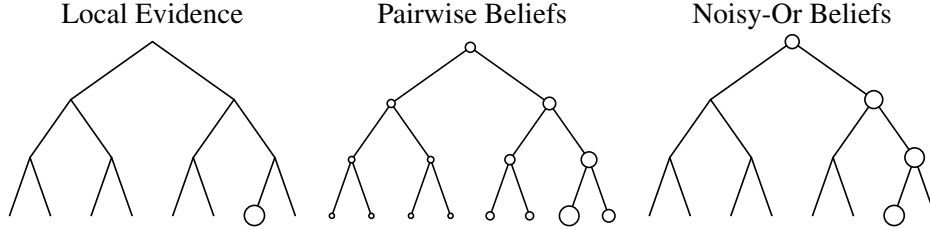


Figure 3.1: An example of belief propagation. Left: A situation in which there is no local evidence for a class being present, except in one leaf node. Middle: Marginals after BP in a pairwise tree. Right: Marginals after BP in a noisy-or tree.

the number of children and may be prohibitively slow. Fortunately, the expected likelihood can be calculated in linear time.

To see that this is the case, consider computing the expected likelihood of a family of nodes $y_p, y_1 \dots y_c$, each with local evidence $P(y_i|e_i)$ representing the contribution from the unary potentials. Note that computing the sum over all child nodes when the parent is off has a factorized form:

$$\sum_{y_1, \dots, y_c} P(y_p = 0 | y_1, \dots, y_c) P(y_1, \dots, y_c | e) = \sum_{y_1, \dots, y_c} \prod_{i=1}^c q^{y_i} P(y_i | e_i) \quad (3.5)$$

Bringing sums inside of products, we obtain the efficient form:

$$\sum_{y_1, \dots, y_c} P(y_p = 0 | y_1, \dots, y_c) P(y_1, \dots, y_c | e) = \prod_{i=1}^c \sum_{y_i} q^{y_i} P(y_i | e_i) \quad (3.6)$$

Since $P(y_p = 1 | y_1, \dots, y_c) = 1 - P(y_p = 0 | y_1, \dots, y_c)$, we can compute every quantity needed efficiently. The normalization constant $P(e)$ can be computed efficiently in the same manner.

3.4 Evidence Flow

In Figure 3.1 we can observe the effects of evidence flowing upwards from the pixel-level labels to the image-level labels. We see that the pairwise tree sends evidence to its neighbours, regardless of spatial distance, while the noisy-or tree only sends evidence to its parents.

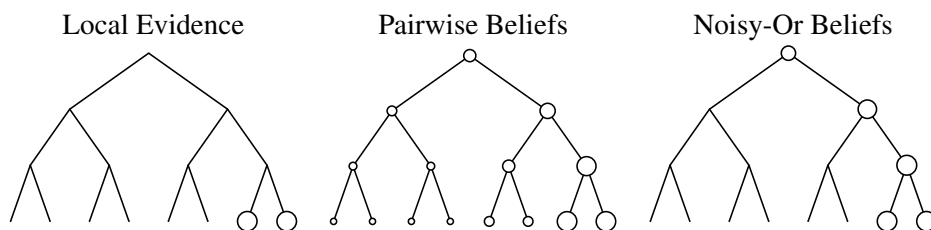


Figure 3.2: Left: A situation in which there is local evidence in two adjacent leaf nodes. Middle: Marginals after BP in a pairwise tree. Right: Marginals after BP in a noisy-or tree.

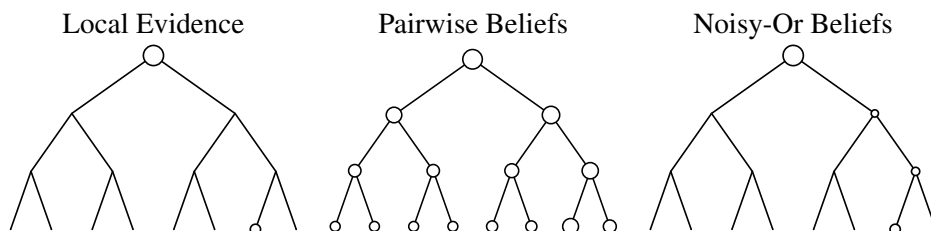


Figure 3.3: Left: A situation in which there is strong evidence at the global scale, and weak local evidence at one of the leaf nodes. Middle: Marginals after BP in a pairwise tree. Right: Marginals after BP in a noisy-or tree.

In Figure 3.2 we can observe the effects of strong evidence in two adjacent nodes. This is equivalent to an object being divided into more than one segment. The pairwise tree significantly increases its confidence that objects are present at other leaf nodes in the image, while the noisy-or tree does not significantly change its marginals.

In Figure 3.3 we can observe manner in which evidence flows down the noisy-or tree. Given strong evidence that a class is present somewhere in the image, and weak evidence that it is present at one location, the pairwise tree adjusts its probability strongly everywhere. The noisy-or tree only adjusts its probability in the regions containing weak evidence.

In Figures 3.4 and 3.5, we can observe the behavior of the two models on a real example. The local evidence potentials are unintuitively small, since they have been calibrated to be combined together across the tree. The most striking feature

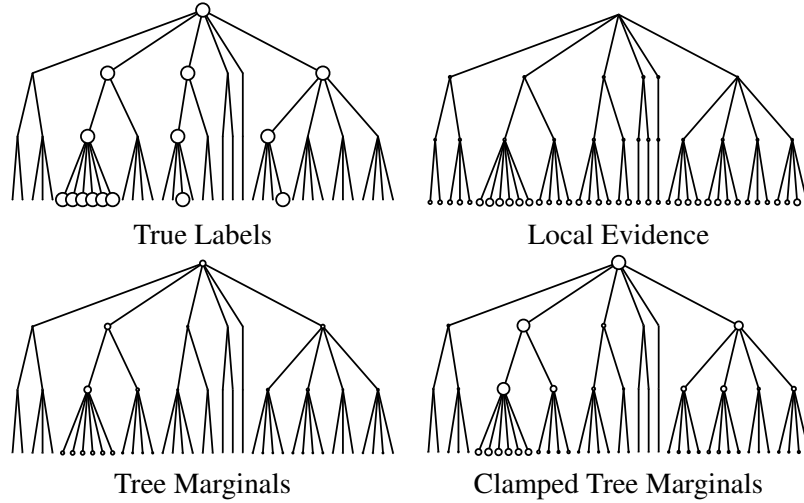


Figure 3.4: An example of belief propagation and evidence flow in a noisy-or tree, trained on real data. Node size is proportional to probability.

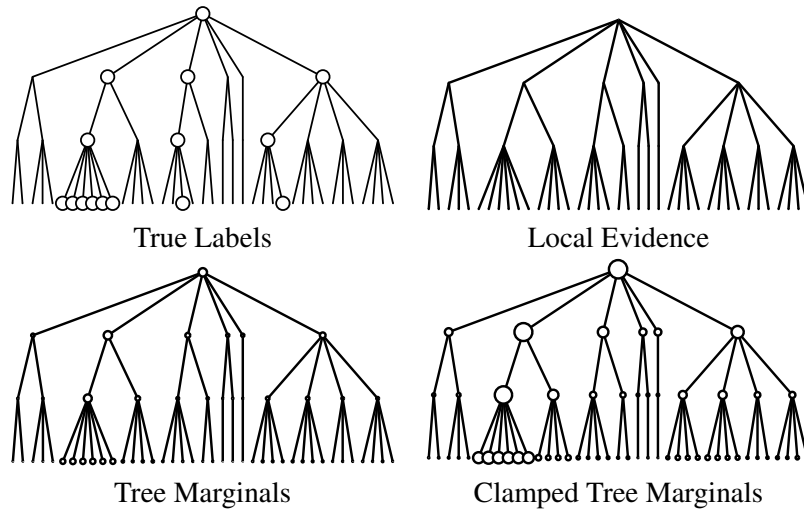


Figure 3.5: An example of belief propagation and evidence flow in the pairwise tree model, trained on real data. Node size is proportional to probability.

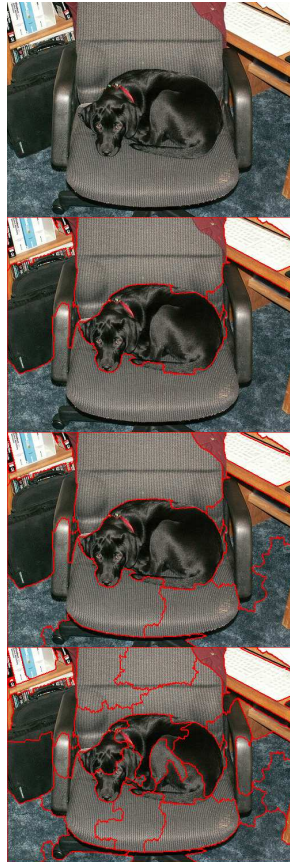


Figure 3.6: The segmentation of the image used in figures 3.5 and 3.4.

of these figures is the difference in the tree marginals before and after the global node has been clamped to the true value.

In these examples, we can again see evidence flowing down the tree from the root, and observe that in the noisy-or model, evidence tends to flow down only one branch of the tree, while in the pairwise model, it tends to flow down all branches to some degree.

3.5 Training

The gradients for the image feature weights W are identical to those of the pairwise tree model once the node marginals have been computed, and can be estimated with

the same E-M algorithm as the pairwise trees.

3.5.1 Learning the Noisy-Or Failure Rate Parameter

In a fully observed, recursively segmented image, the maximum likelihood estimate for the failure rate parameter q will always be zero, since a parent node will be observed to be on if and only if a child is on. However, on partially observed data, this is not necessarily the case.

In initial experiments, the parameter q was learned in parallel with the feature weights W , but as the model converged, the learned q parameter again tended towards zero. For the experiments below, this parameter was fixed to 0.01.

3.6 Results

Shown in figures 3.7, 3.8 and 3.9 are the results on the noisy-or model of a set of experiments identical to those performed on the pairwise model. We observe a similar increase in accuracy after semi-supervised training as in the pairwise model. A detailed comparison is given in Chapter 4.

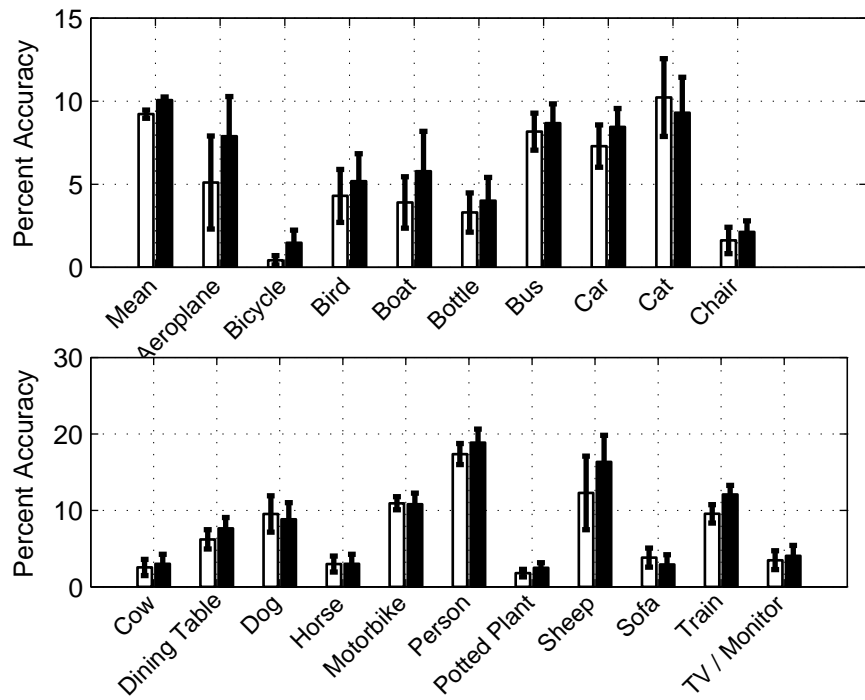


Figure 3.7: Pixel-level test accuracy on the noisy-or model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.

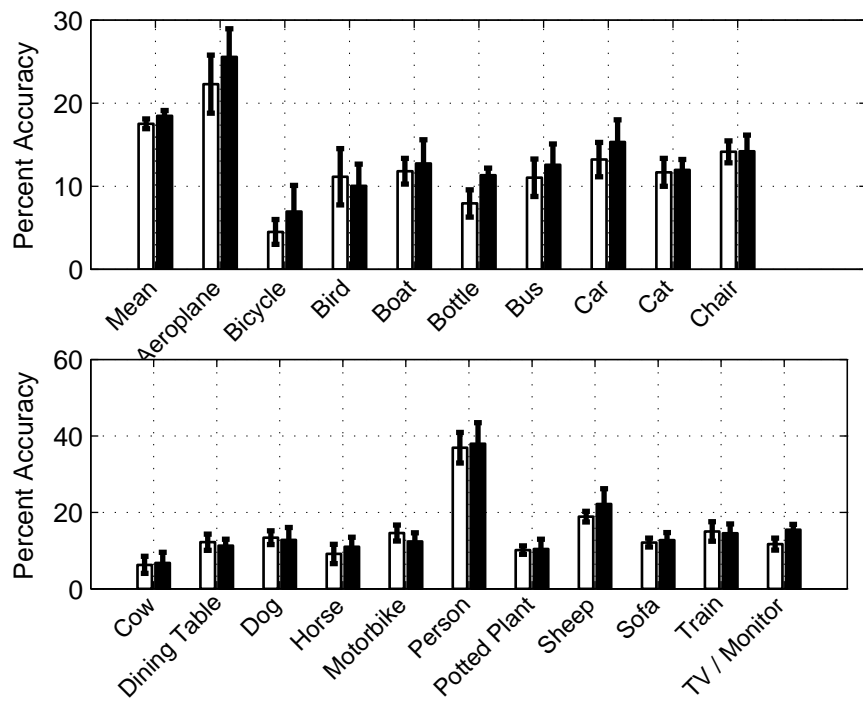


Figure 3.8: Global-level test accuracy on the noisy-or model. White bars indicate performance on fully-labeled data, black bars indicate performance after additional semi-supervised training.

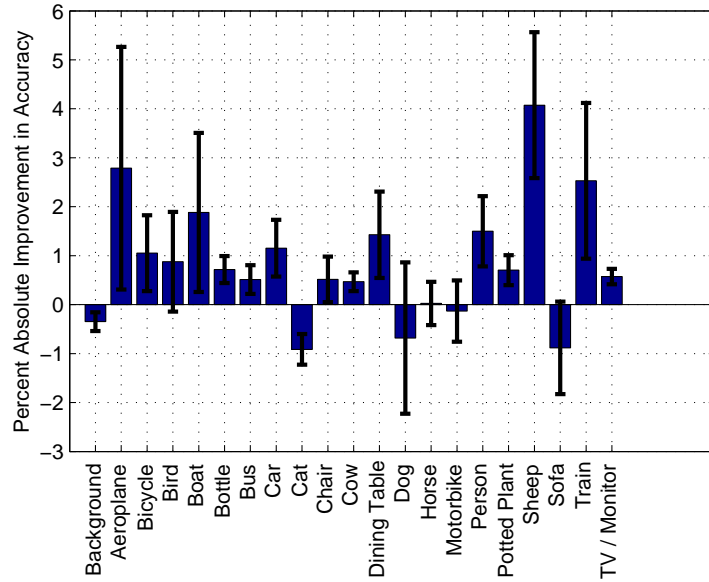


Figure 3.9: Change in test pixel-level accuracy after training with partially labeled data.

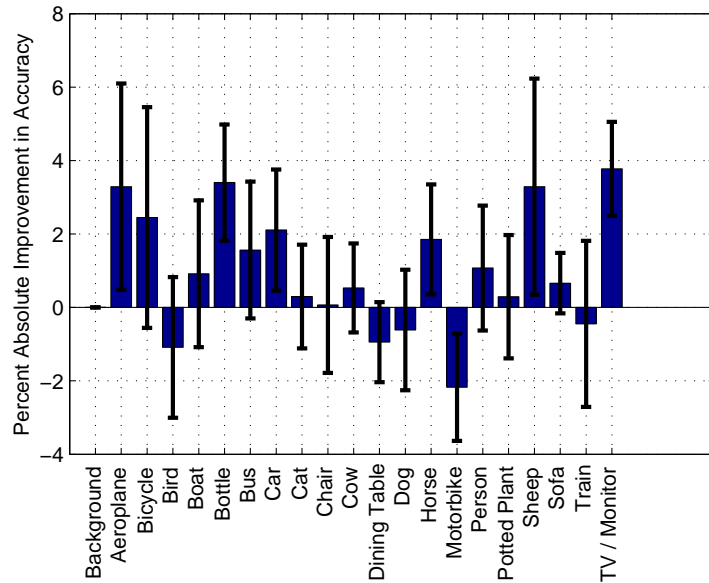


Figure 3.10: Change in test global-level test accuracy after training with partially labeled data.

Chapter 4

Model Comparison

4.1 Comparing Models

In this chapter we compare the performance of the two tree structured models, as well as with the independent model.

4.1.1 Independent Model

As a baseline, we compare our tree-structured CRF models to the independent model. This model has the same node potential factors as the tree models, but with no factors connecting nodes. Thus, every region label is predicted separately, and the model becomes equivalent to a per-region logistic regression on each region's image features. The parameters of the independent model are trained on only the fully-labeled data.

4.2 Performance Measures

Below we show pixel-level labeling VOC accuracy, global-level VOC accuracy, and cross-entropy for the three models. Here the cross-entropy is measured between the true node labels and the node marginals, over all nodes in all trees $T_{1\dots n}$:

$$-\frac{1}{N} \sum_{n=1}^N \sum_{i \in T_n} \sum_{y=0}^1 P_{\text{true}}(y_{ni} = y) \log [\hat{P}(y_{ni} = y)] \quad (2.1)$$

Note that the cross entropy is proportional to the training objective function of the independent model, but not of the tree models.

4.3 Performance Comparison

Figures 4.3, 4.4, and 4.5 show mean performance across all 21 classes in the VOC dataset, averaged over 5 folds. Error bars represent one standard error.

4.3.1 Pixel-level Accuracy

Figure 4.3 compares model performance with respect to pixel-level accuracy. The difference between the three models in the supervised setting is statistically insignificant. However a paired t -test shows significant improvement in accuracy after semi-supervised training for both the pairwise and noisy-or models.

4.3.2 Global-level Accuracy

Figure 4.4 shows a surprising result: The independent model performs slightly better at the global classification task than the supervised-only tree models. This is surprising, since the independent global label prediction is equivalent to a logistic regression on the global-level features of the image, and has no access to image features at finer scales.

How can we explain the poor global-level performance of the tree models, given that their pixel level accuracy was better? We propose two possible explanations.

Our first possible explanation is that the tree models’ training objective is heavily weighted towards matching the labels at the bottom levels of the tree as opposed to the top of the tree, since there are many more bottom-layer nodes than global-layer nodes. To the extent that there is any trade-off between how well a particular set of parameters models these two groups, we would expect the model to tend to choose parameters that accurately model the pixel-level labels at the expense of the global-level labels. The independent model faces no such trade-off, since it models the global-level and pixel-level nodes of the tree independently.

Our second, related explanation is that the tree-based models face a trade-off between modeling nodes with small numbers of neighbours, versus modeling

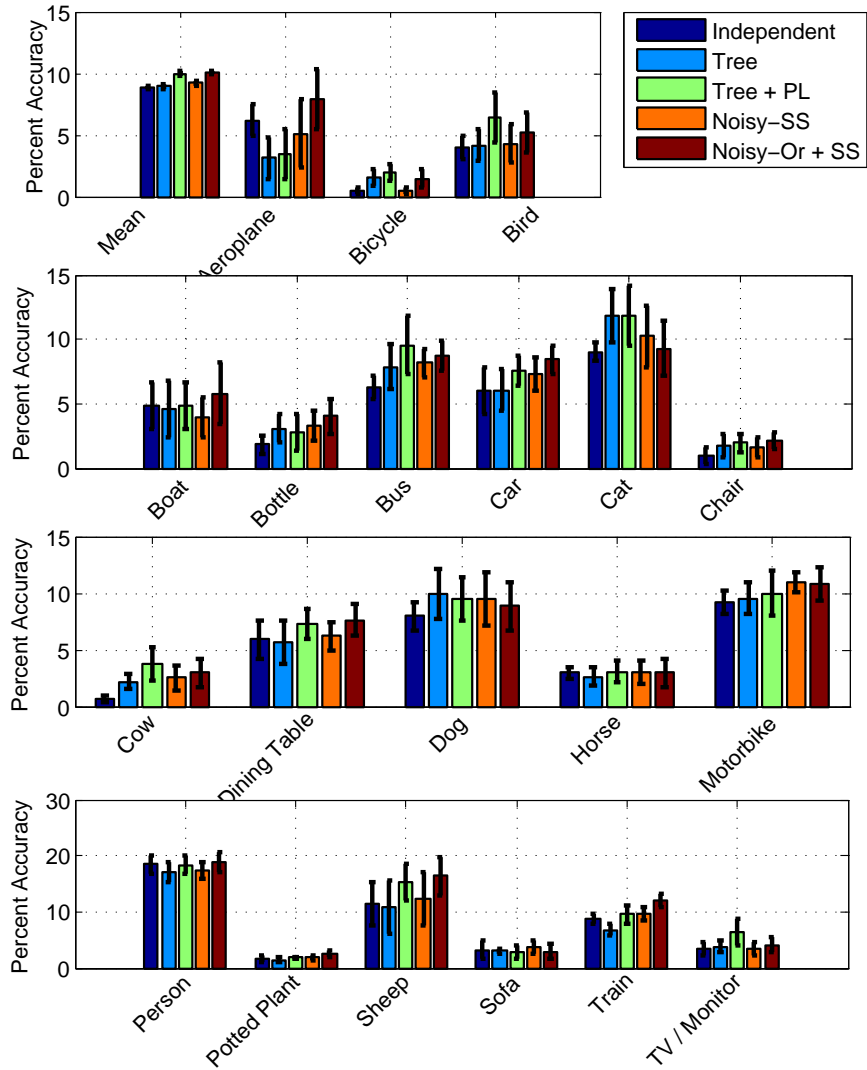


Figure 4.1: Pixel-level test accuracy on the three models, with and without semi-supervised training.

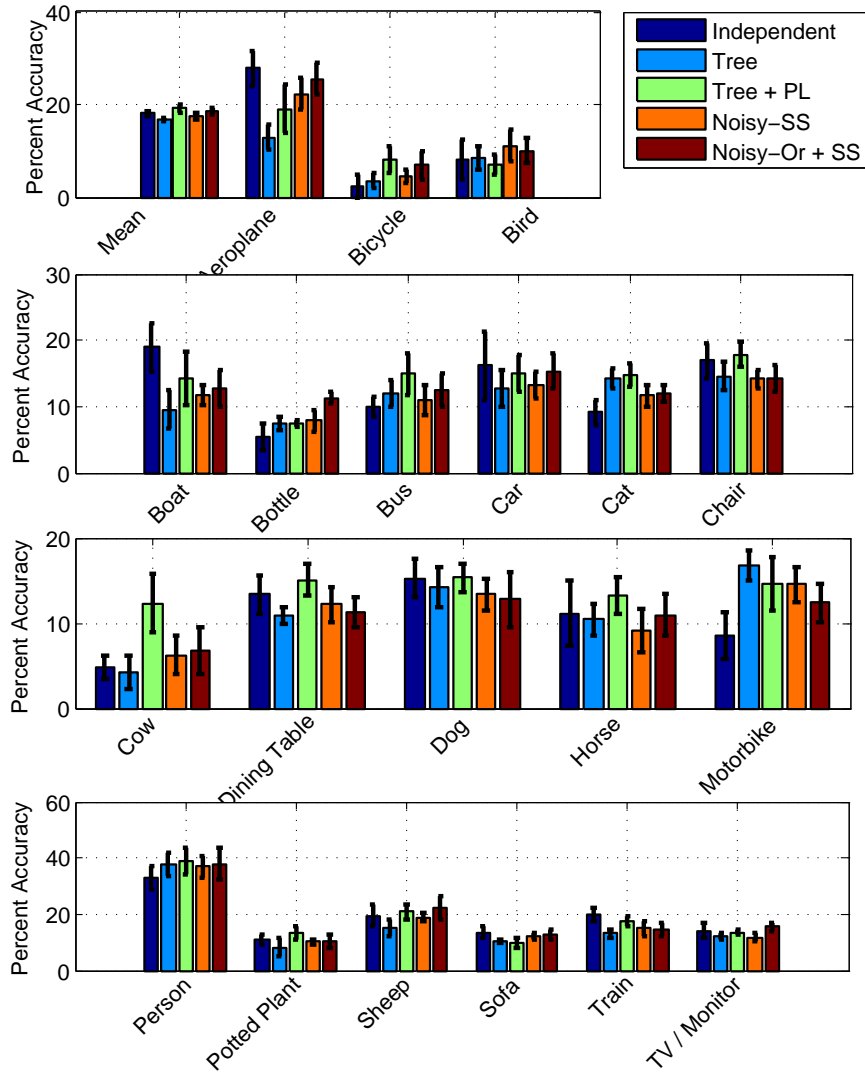


Figure 4.2: Global-level test accuracy on the three models, with and without semi-supervised training.

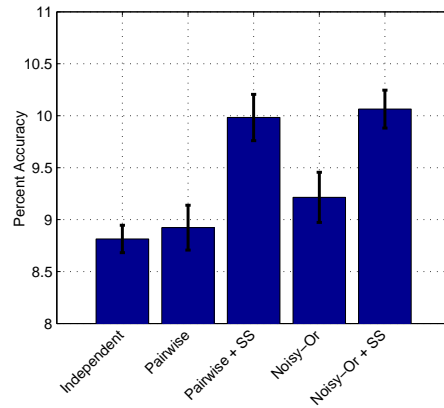


Figure 4.3: Pixel-level test accuracy across all models.

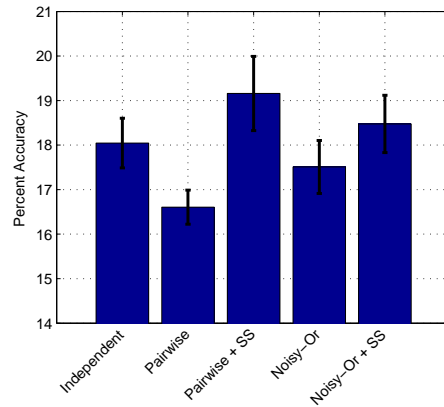


Figure 4.4: Global-level test accuracy across all models.

nodes with many neighbours, and that this trade-off again favours nodes near the bottom of the tree, since there are many more such nodes. Figure 4.6 compares the distribution of degree of all nodes versus only global-level nodes.

Figure 4.7 shows the cross-entropy on the training set versus the number of neighbours a node has. We see that the cross-entropy of the tree models is lowest (best fitting the data) for nodes with a small number of neighbours, and increases with the number of neighbours.

To verify that it is not simply the case that nodes with higher numbers of edges are not simply harder to model for some reason, we also plot the performance of the independent model versus the number of neighbours that each node would

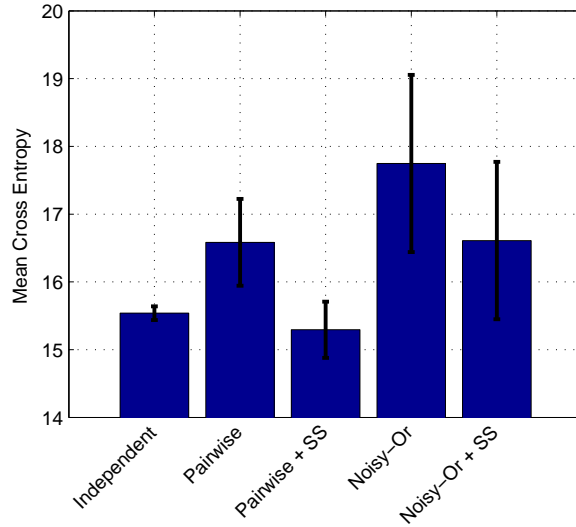


Figure 4.5: Mean test cross entropy over all nodes across all models. Lower is better.

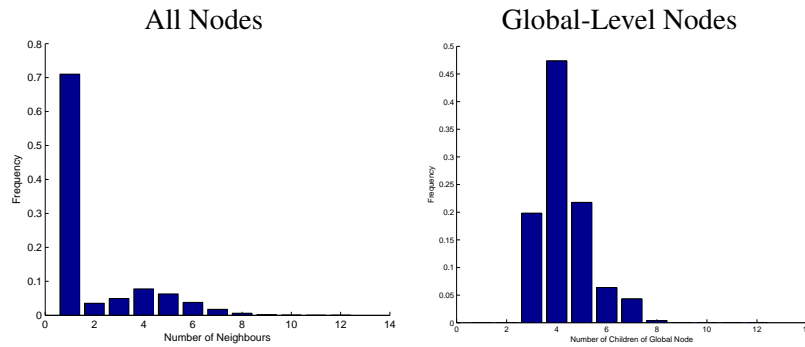


Figure 4.6: Left: Histogram of the number of neighbours of each node in the training set. Nodes with one neighbour are necessarily leaves. Right: Histogram of the number of neighbours of global-level nodes in the training set.

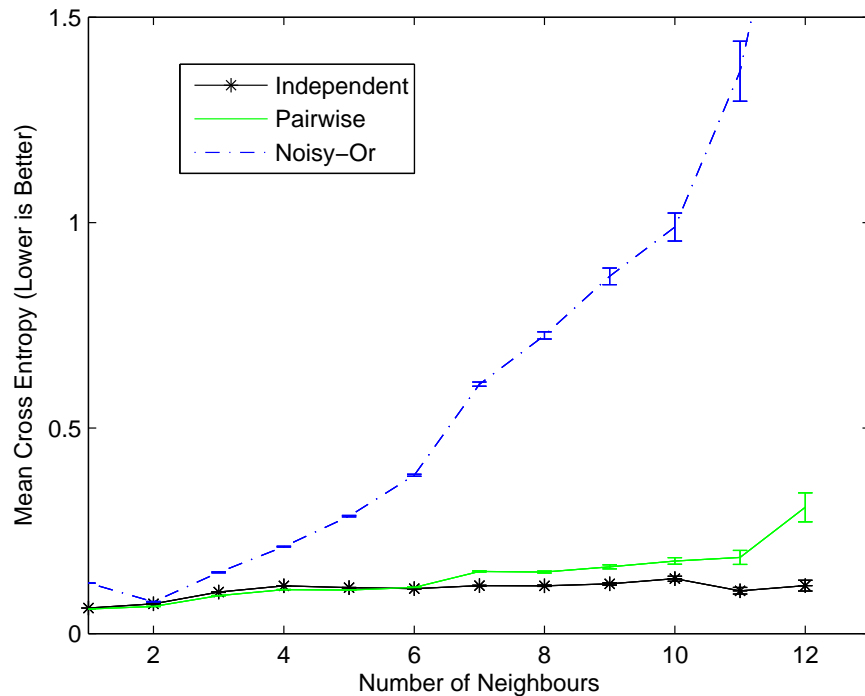


Figure 4.7: Mean cross-entropy per node versus the number of neighbours of a node on the training set.

have had if the graph were connected. We see that the performance in this case is roughly independent of the number of nodes.

While both tree models show roughly the same trend, there still remains the question of why the noisy-or tree matches the node marginals much more poorly on the training set than the pairwise model. This may be due to the fact that the pairwise model has 3 sets of free parameters with which to calibrate the pairwise potentials between nodes, while the noisy-or model has no free parameters to adjust its joint potentials.

4.3.3 Remedies

If our first explanation of the poor global-level performance is correct, then a simple way to improve the performance of the model at the global-level task would be to weight the likelihood contribution of the global node more strongly during

training.

If our second explanation is correct, then the global-level performance can be improved in several ways. First, we may consider constraining the recursive segmentation such that the resulting tree has a restricted range of node degrees. This approach has the disadvantage that it makes the segment joining task computationally difficult, and constrains the segmenter’s ability to group similar regions. Alternatively, we may wish to parameterize the joint factors in a way that takes into account the varying number of neighbours a node has. One approach, which retains the linear-time computation of the pairwise and noisy-or factors, is outlined in Chapter 5.

4.4 Qualitative Evaluation

In Figures 4.8 and 4.9, we compare the pixel level probabilities among the three models for two images in the test set. In both of the images shown, the independent model finds several disparate patches which match the class of interest. In the tree models, this evidence is combined at higher levels in the tree, and we observe a smoother labeling at the pixel level.

4.5 Improving Performance

The performance of the tree models in these experiments is unimpressive relative to the state of the art. However, there are several reasons to expect that significantly better performance can be achieved, at the cost of slower training times¹.

- The recursive segmentation can be made finer. In the experiments performed above, the recursive segmentation was only four levels deep, leaving relatively large segments at the finest scale. Allowing a finer segmentation would raise the upper bound on pixel-level accuracy caused by under-segmentation.
- The training set can be left unbalanced. In the experiments above, the training datasets were balanced by removing approximately 80% of images that

¹ In the experiments above, the tree models take approximately 2 hours to train per class, for a given setting of the hyperparameters. The main bottleneck in training the model is in performing inference at each step on each of the the partially-labeled examples. However, this inference step can be computed in parallel over all examples.

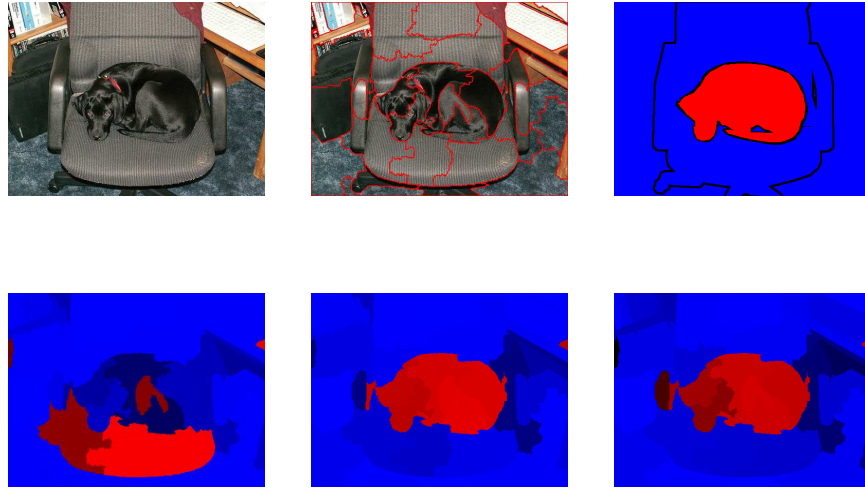


Figure 4.8: Detecting a dog. Top left: Original image. Top center: Segmentation at bottom level. Top right: True pixel labels. Bottom left: Pixel probabilities for independent model. Bottom center: Pixel probabilities for pairwise model. Bottom right: Pixel probabilities for noisy-or model.

did not contain the class of interest.

- Greater care can be put into selecting image features for the node potentials.
- The number of unlabeled examples can be increased relatively easily. To gather training data for the “dog” image model, for example, it suffices to merely find images that somewhere contain a dog, with no further labeling required. Note that these models can be trained on images with probabilistic labels, allowing the use of automatically gathered datasets where it is known that some percentage of images do not contain the class of interest.
- Performance on the pairwise model could potentially be improved by regularizing the edge-potential parameters (of which there were twelve in the pairwise tree model used). However, this would entail another hyperparameter on which to perform cross-validation.



Figure 4.9: Detecting a person. Top left: Original image. Top center: Segmentation at bottom level. Top right: True pixel labels. Bottom left: Pixel probabilities for independent model. Bottom center: Pixel probabilities for pairwise model. Bottom right: Pixel probabilities for noisy-or model.

4.6 Introducing an Oracle

Although one motivation of tree-structured CRF models is that they can perform simultaneous image classification and object localization, it is certainly the case that a model dedicated to image classification may perform better at that task than our model. However, we can easily incorporate evidence from a separate image classification algorithm into our model by introducing an extra factor at the global-level node. This evidence would then be propagated by belief propagation down to the bottom-level nodes, where pixel-level accuracy may be improved.

Following [19], we calculate an upper bound on the possible performance boost in pixel-level accuracy attainable by incorporating evidence from a better global-level classifier. In Figure 4.10 we show the pixel-level accuracy of the models in the setting where an oracle tells us the correct image classification. We see that both

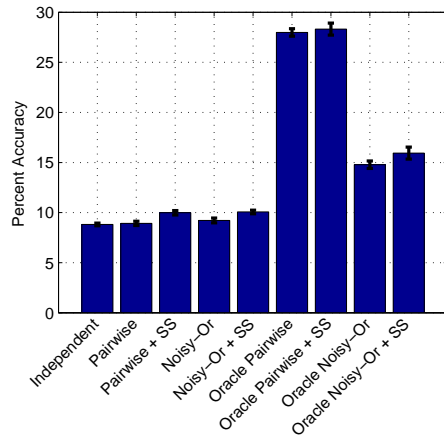


Figure 4.10: Pixel-level test accuracy across all models, including the case where the global-level nodes were clamped to their true value.

models obtain a large increase in accuracy when combined with an oracle. This result is consistent with results in [19], who also report a large boost in pixel-level accuracy by coupling with an oracle.

We also observe that the pairwise model receives a much greater boost in accuracy from the oracle than the pairwise model. The cause of this result is unclear. One possible explanation is suggested by in Figure 3.3, where, upon clamping the root node to be on, the pairwise model increases node marginals among all the bottom-level nodes, while the noisy-or model increases node marginals only at bottom-level nodes whose marginals are already high. This behavior suggests that the noisy-or model might be inappropriately “explaining away” the oracle’s being on by increasing the marginal probability of only a small number of bottom-level nodes.

Chapter 5

Conclusions and Future Work

In this chapter, we discuss some well-motivated possible extensions of our work, and attempt to characterize the different models studied.

5.1 Future Work

5.1.1 Structure-Adaptive Potentials

As mentioned in Chapter 4, the tree models had difficulty modeling the marginals of nodes with high degree. To address this, we may wish to learn a more flexible factor computing the joint probability of neighbouring nodes. For instance, we could learn a tabular factor with a different potential for each setting of parents and children. However, without introducing additional structure to these factors, this approach has two major disadvantages: It would no longer be possible to compute messages or likelihoods efficiently, and we would need to learn a number of parameters exponential in the number of children.

We can attempt to correct the problem of the parent's local evidence being either over- or under-weighted in the following way: We can redefine the local evidence factors to be scaled according to a parameter α_g that varies with the number of children g a node has.

For the pairwise model, the unary potential for node r was defined as:

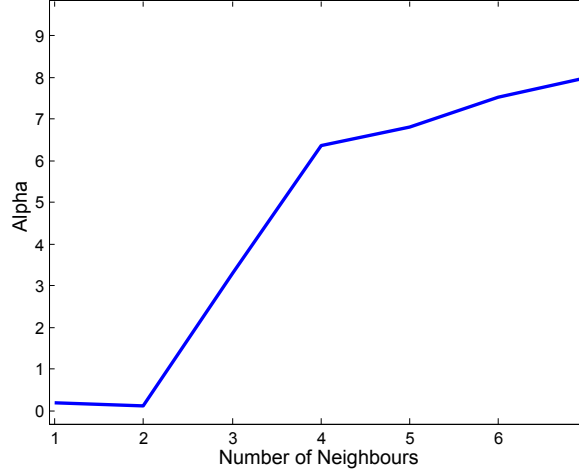


Figure 5.1: Learned α_g versus the number of neighbours.

$$\phi_f \left(y_c^{(r)}, x^{(r)} \right) = y_c^{(r)} (x^{(r)})^T W_c^l \quad (1.1)$$

The new joint potential would be defined as:

$$\phi_f \left(y_c^{(r)}, x^{(r)} \right) = \alpha_g y_c^{(r)} (x^{(r)})^T W_c^l \quad (1.2)$$

Where α_g is the reweighting parameter for nodes with g children.

In a preliminary experiment, we first learned W as normal, then optimized α in a second step by numerical differentiation of the expected data log-likelihood. Because so few nodes have more than 9 children, we tied together α_g for all values of $g > 9$.

We would expect α_g to increase monotonically with the number of children, and this is what we see in Figure 5.1. After α has been learned, the local evidence is weighted more strongly in nodes that have many neighbours.

5.1.2 Bounding Box Data

One major advantage to using noisy-or tree models over pairwise tree models is that they can more consistently incorporate information from bounding boxes in an image.

When viewed as information about an image, all a bounding box tells us is that a certain area of an image contains a certain class. In a pairwise model, the most straightforward way to incorporate this evidence into the model is by asserting that all of the pixels in the bounded area belong to the specified class, with some probability higher than they would have otherwise had.

In the noisy-or model, we can incorporate bounding box information consistently, by setting a node containing the entire bounding box to be 'on'. If the shape of the smallest region containing the bounding box closely matches the shape of the bounding box, we will be effectively incorporating the information provided by the bounding box, while giving the model more flexibility in assigning individual pixel labels.

These properties of noisy-or models suggest a set of experiments incorporating bounding-box labeled image data into the semi-supervised training phase.

5.1.3 Combined Grid and Tree Structures

The most popular form of CRFs on images is the grid-structured CRF, where neighbouring pixels or regions are connected. One problem inherent in multi-scale tree models is that evidence may be unable to flow directly between neighbouring regions in the image in this way. We can combine these models by adding pairwise connections between neighbouring nodes at each level of the tree. The disadvantage to adding potentials to the tree model is that we lose the ability to do efficient exact inference, and to compute the expected data log-likelihood exactly. However, using loopy belief propagation, we can compute approximate marginals as well as an approximate gradient of the expected log-likelihood, allowing us to optimize the parameters using gradient descent.

5.1.4 Joint Learning Over All Classes

If we are willing to sacrifice exact inference, there is another interesting extension of the tree models: Connecting the trees of all classes together at the pixel-level nodes, where a mutual exclusion between classes is enforced. In this way, if one class is successfully identified in part of an image, the remaining classes can receive evidence that they are not contained in that region. This effect could be expected to improve both the efficiency of semi-supervised learning and test-time localization performance.

5.1.5 Large-Scale Experiments

Of the 400 caption-only VOC 2008 images used for semi-supervised learning, on most classes only 20-50 images in that set actually contained the class of interest. Given the improvement in performance obtained by adding only this small number of examples to the test set, it seems worth noting that a large weakly-labeled dataset could easily be constructed for a small number of classes, to evaluate the effectiveness of yet adding more caption-only data.

5.2 Concluding Remarks

When we began this investigation, we had three motivations for looking at multi-scale CRFs built on a recursive segmentation.

Our motivation for using a recursive segmentation was that it affords the use of noisy-or factors, which were expected to be better able to model the node labels in this problem. Surprisingly, the noisy-or model did not perform significantly better than the pairwise model, and on one task (pixel-level labeling with a global oracle) performed worse. Our explanation follows from the following observation: in the lower levels of the tree, it is often the case that when a parent node is on, most of its children are also on. The pairwise factors can model this fact relatively well, by simply giving each child a high likelihood of being on independently. However, the noisy-or factor is constrained to give nearly equal likelihood to all cases where at least one child is on. Thus while the noisy-or factor correctly models the state of the parent node given the child nodes, it cannot easily match the empirical distribution over children when conditioned on the parent. Adding links between the child

nodes may be necessary to make the noisy-or model perform well.

The second motivation for the use of a multi-scale model was that it allows the same model to be used for both pixel-level labeling as well as global-level classification. However, the results in chapter 4 suggest that combining both of these goals into one objective function may have caused unwanted trade-offs during training, and that it may be best to optimize only one of these objectives at a time.

Our central motivation for using multi-scale CRFs was their ability to learn from weakly labeled data. This ability was clearly demonstrated: during semi-supervised training, we were able to observe evidence flowing from the labeled global-level nodes to the unlabeled pixel-level nodes, and at test time, we observed an increase in pixel-level accuracy. This ability to learn from caption-only data will only become more useful, as datasets such as ImageNet[3] continue to grow. We therefore find this result highly encouraging.

Bibliography

- [1] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Computer Vision and Pattern Recognition*, volume 2, pages 1124–1131, 2005. → pages
- [2] N. Dalai, B. Triggs, R. I. Alps, and F. Montbonnot. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1, 2005. → pages 7
- [3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. → pages 45
- [4] L. Du, L. Ren, D. Dunson, and L. Carin. A bayesian model for simultaneous image clustering, annotation and object segmentation. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 486–494. 2009. → pages
- [5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. <http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html>. → pages 7
- [6] X. Feng, C. K. I. Williams, and S. N. Felderhof. Combining belief networks and neural networks for scene segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):467–483, 2002. → pages
- [7] B. Frey and D. Dueck. Mixture modeling by affinity propagation. *Advances in neural information processing systems*, 18:379, 2006. → pages 4
- [8] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. 2009. → pages

- [9] X. He and R. S. Zemel. Learning hybrid models for image annotation with partially labeled data. In *Advances in Neural Information Processing Systems*, 2008. → pages
- [10] T. S. Jaakkola and M. I. Jordan. Variational probabilistic inference and the qmr-dt network. *Journal of Artificial Intelligence Research*, 10:291–322, 1999. → pages
- [11] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289, 2001. → pages
- [12] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *ICML*, 2009. → pages
- [13] L. Liao, D. Fox, and H. A. Kautz. Hierarchical conditional random fields for gps-based activity recognition. In *ISRR*, pages 487–506, 2005. → pages
- [14] D. G. Lowe. Object recognition from local scale-invariant features. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 1150–1157, 1999. → pages
- [15] K. Murphy, A. Torralba, and W. Freeman. Using the forest to see the trees: a graphical model relating features, objects and scenes. *Advances in Neural Information Processing Systems*, 16, 2003. → pages 2
- [16] K. Murphy, A. Torralba, D. Eaton, and W. Freeman. Object detection and localization using local and global features. *Toward Category-Level Object Recognition*, pages 382–400, 2006. → pages 7, 9
- [17] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001. → pages 7
- [18] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988. → pages 14, 20, 21
- [19] N. Plath, M. Toussaint, and S. Nakajima. Multi-class image segmentation using conditional random fields and global classification. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 817–824. ACM, 2009. → pages 2, 3, 6, 9, 39, 40

- [20] J. Reynolds and K. Murphy. Figure-ground segmentation using a hierarchical conditional random field. In *Fourth Canadian Conference on Computer and Robot Vision, 2007. CRV'07*, pages 175–182, 2007. → pages 2, 3, 9
- [21] X. H. Richard, R. S. Zemel, and M. A. C. perpi nán. Multiscale conditional random fields for image labeling. In *CVPR*, pages 695–702, 2004. → pages
- [22] M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. In *ECML '07: Proceedings of the 18th European conference on Machine Learning*, pages 286–297, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-74957-8. → pages
- [23] M. Schmidt, K. Murphy, G. Fung, and R. Rosales. Structure learning in random fields for heart motion abnormality detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008. → pages
- [24] P. Schnitzspan, M. Fritz, S. Roth, and B. Schiele. Discriminative structure learning of hierarchical representations for object detection. In *CVPR*, pages 2238–2245, 2009. → pages
- [25] J. Shi and J. Malik. *Normalized Cuts and Image Segmentation*, 2000. → pages
- [26] A. J. Storkey and C. K. I. Williams. Image modelling with position-encoding dynamic trees. *IEEE Trans. Pattern Anal. Machine Intell*, 25:859–871, 2003. → pages
- [27] M. Szummer, P. Kohli, and D. Hoiem. Learning crfs using graph cuts. In *ECCV*, volume 5303 of *Lecture Notes in Computer Science*, pages 582–595. Springer, 2008. → pages
- [28] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org/>, 2008. → pages 4
- [29] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. *Computer Vision–ECCV 2008*, pages 705–718, 2008. → pages
- [30] J. Verbeek and B. Triggs. Scene segmentation with crfs learned from partially labeled images. In *Advances in Neural Information Processing Systems*, 2007. → pages

- [31] P. Wu, B. Manjunanth, S. Newsam, and H. Shin. A texture descriptor for image retrieval and browsing. In *IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 3–7, 1999. → pages 7
- [32] J. Zhu, E. P. Xing, and B. Zhang. Partially observed maximum entropy discrimination markov networks. In *Advances in Neural Information Processing Systems*, 2008. → pages