Stacked Hierarchical Labeling

Daniel Munoz, J. Andrew Bagnell, and Martial Hebert

The Robotics Institute
Carnegie Mellon University
{dmunoz,dbagnell,hebert}@ri.cmu.edu

Abstract. In this work we propose a hierarchical approach for labeling semantic objects and regions in scenes. Our approach is reminiscent of early vision literature in that we use a decomposition of the image in order to encode relational and spatial information. In contrast to much existing work on structured prediction for scene understanding, we bypass a global probabilistic model and instead directly train a hierarchical inference *procedure* inspired by the message passing mechanics of some approximate inference procedures in graphical models. This approach mitigates both the theoretical and empirical difficulties of learning probabilistic models when exact inference is intractable. In particular, we draw from recent work in machine learning and break the complex inference process into a hierarchical series of simple machine learning subproblems. Each subproblem in the hierarchy is designed to capture the image and contextual statistics in the scene. This hierarchy spans coarse-to-fine regions and explicitly models the mixtures of semantic labels that may be present due to imperfect segmentation. To avoid cascading of errors and overfitting, we train the learning problems in sequence to ensure robustness to likely errors earlier in the inference sequence and leverage the stacking approach developed by Cohen *et al.*

1 Introduction

The challenging problem of segmenting and labeling an image into semantically coherent regions can be naturally modeled as a hierarchical process to interpret the scene [23]. Typically, a graphical model is used where each node represents the labels present in some region of the image with dependencies that tie together multiple regions [3,6]. The nodes at the bottom of the hierarchy provide low-level discriminative information, while nodes higher up resolve ambiguities using global information. While these representations seem intuitive, learning the optimal components of the model is practically intractable due to complex dependencies. Furthermore, even with simplified representations exact inference remains intractable [13] and prohibits learning these models. Although training with exact inference is infeasible, a natural alternative is to use *approximate* inference. However, as we discuss in the next section, these approximations during learning can lead to undesirable behavior [16]. Therefore, we move away from a representation for which training is intractable and toward an approach which relies on effective components that are simple to train.

K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part VI, LNCS 6316, pp. 57–70, 2010.

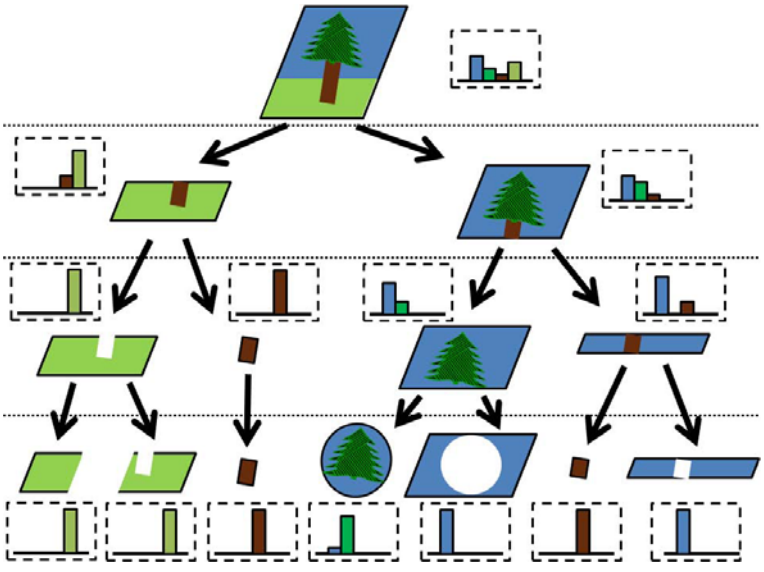


Fig. 1. A synthetic example of our hierarchical labeling process. Given an image and its hierarchical decomposition of regions, we sequentially predict the proportion of labels present (drawn in the dashed boxes) using image features and previous predictions.

In this work, we model low-level information combined with higher-order reasoning using a hierarchical representation. Our approach is similar to previous structured models with the key difference that we no longer attempt the intractable task of finding the mode of the joint posterior distribution using a generic approximate inference algorithm. Instead we simplify the problem into a series of subproblems that are specifically trained to perform well for our task. That is, we train these subproblems to model the relations present in the image so that the overall prediction is correct. One major advantage of this approach is that test-time structured prediction is simply a sequence of predictions. Our contribution is a novel hierarchical algorithm for labeling semantic regions.

An idealized example of our approach is depicted in Fig. 1. We represent the inference process as a series of predictions along the hierarchy from coarse to fine. Given an image, we first create a hierarchy of regions that range from very large regions in the image (including the image itself as one region at the top) down to small regions (*e.g.*, superpixels) at the bottom. We do not rely on each region to contain one label; instead we explicitly model the label proportions in each region. Starting with the entire image, we train a classifier[1] to predict the proportions of labels in the image. As we further discuss in Sect. 3, these predictions are passed to the child level and are used to train another classifier

[1] In this work, we refer to a classifier as an algorithm that predicts a distribution over labels, instead of a single label.

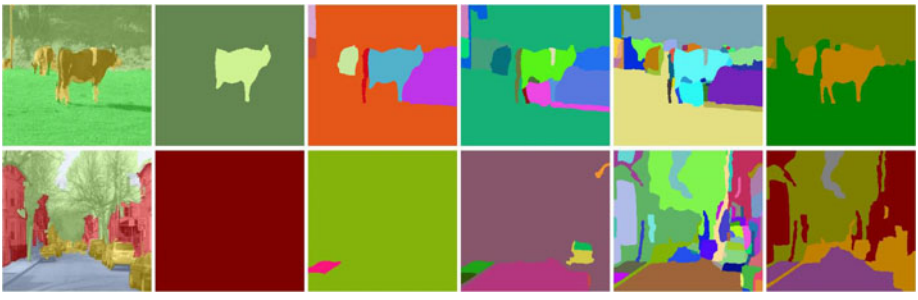


Fig. 2. Hierarchy analysis for two images. From left to right: input image with ground truth overlaid, the segmentation maps for L2 (second level), L4, L6, L8, and most likely label for each region in L8.

over the child subregions. The procedure is repeated until the leaves are reached. Since we model label proportions over regions: we are robust to imperfect segmentation, we can use features defined over large regions, and we do not make hard commitments during inference.

Figure 2 illustrates four levels from the hierarchy on two images from the Stanford Background Dataset (SBD) [8]. Ideally the leaves in the hierarchy are regions that contain only one label, but as Fig. 2 also illustrates, this assumption is not always true, especially for more complex scenes. With our hierarchical approach, we demonstrate state-of-the-art performance on SBD and MSRC-21 [26] with the added benefit of drastically simpler computations over global methods.

2 Background

2.1 Motivation

Random field models in vision have proven to be an effective tool and are also attractive due to their convex nature (assuming no latent states) [19]. Furthermore, although exact inference is NP-hard over these models, there has been much recent progress towards efficient *approximate* inference techniques [12,14]. However, correctly optimizing these convex models requires *exact* inference during learning. Unfortunately, when exact inference cannot be performed, converging to the optimum is no longer guaranteed [16]. For example, Kulesza and Pereira [16] demonstrate a case where learning with bounded approximate inference can prevent the learning procedure from ever reaching a feasible zero empirical risk solution. Similarly in another example, they show that learning with loopy belief propagation can diverge.

As we are forced to use approximate inference during learning, the learned model (*e.g.*, parameters) is tightly tied to the chosen inference procedure in both theory [29] and in practice [17]. However, learning the best model for the chosen inference procedure is often still difficult. Due to this fundamental limitation

when training with approximate inference techniques, we move away from the global probabilistic interpretation used in hierarchical random field formulations, such as [18]. Instead, in a manner inspired by inference procedures over graphical models, we propose a novel method using iterative classifiers that are trained to encode interactions between levels but correspond to no explicit joint probability distribution.

2.2 Related Work

Our hierarchical formulation resembles early directed graphical models from Bouman and Shapiro [3] and Feng *et al.* [6] for scene analysis. Whereas these approaches rely on tree-based interactions to enable tractable learning, we no longer train a graphical model and are not restricted in the types of contextual cues that we can use. Instead we focus on maximizing what we ultimately care about: predicting correct labelings. This idea is analogous to the difficult and non-convex problem of maximizing the marginals [11]. The notion of training the inference algorithm to make correct predictions is also similar to Barbu [2] for image denoising, in which a model is trained knowing that an inaccurate, but fast, inference algorithm will be used. In our approach we break up the complex structured prediction problem into a series of simpler classification problems, inspired by recent works in machine learning focused on sequence prediction [4,5]. In the vision setting, this notion of a series of classification problems is similar to Auto-context [27], in which pixel classifiers are trained in series using the previous classifier's predictions with pairwise information to model contextual cues. In our work, we go beyond typical site-wise representations that require entities to contain one label. Because we model label proportions, we can use features defined over large regions to better represent the context, rather than an aggregation of site-wise labels. Furthermore, the hierarchy provides spatial support context between levels and naturally propagates long-range interactions that may be hard to capture with pairwise interactions. We build on the forward sequential learning approach used and analyzed in [28,10,25] to prevent cascading errors and leverage the sequential stacking idea to minimize cascaded overfitting [30,4,15].

3 Stacked Hierarchical Labeling

3.1 Overview

Given an image and its hierarchical region representation, we train a series of classifiers, from coarse to fine, to predict the label proportions in each region in the level. After a level has been trained, the predicted labels are passed to the child regions to be used as features that model contextual relationships. Figure 3 illustrates (on test data) how the probabilities for the respective labels increase and become more precise along three levels in the lower half of the hierarchy. Our approach is robust to the quality of the segmentation at each level as we explicitly model that regions may contain multiple labels. Therefore, depending on how

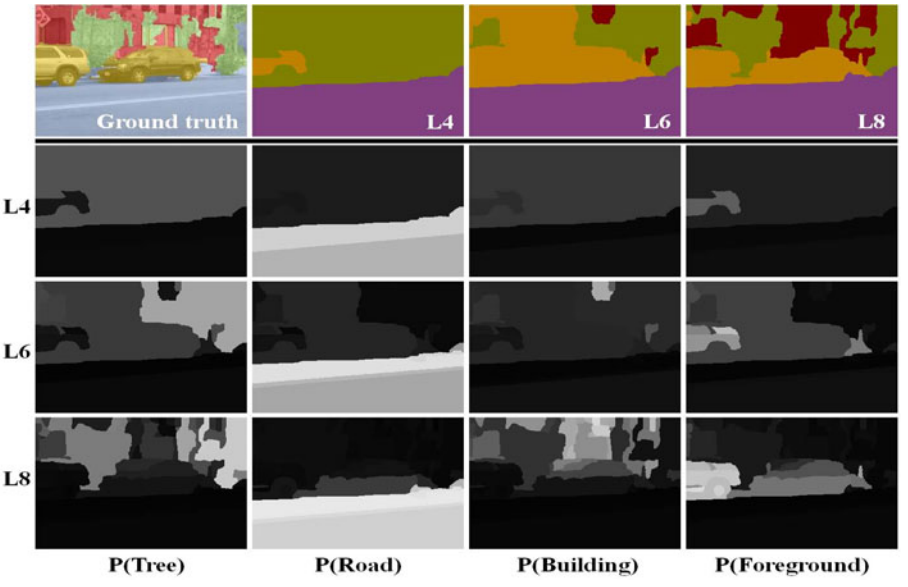


Fig. 3. Refinement in label predictions down the hierarchy. Row 1: Test image (with ground truth overlaid) and predictions for three levels in the hierarchy. Rows 2-4: The respective level's label probability maps, where white indicates high probability.

the hierarchy is constructed, our algorithm will learn how regions for different labels are split between levels. We create the hierarchy using the technique from Arbelaez *et al.* [1,22].

The next subsections describe each component of the training procedure. We first introduce the notations and describe the basic classifier used at each level (Sect. 3.2). We then describe how predictions from a parent region are incorporated as features (Sect. 3.3) and how classifiers are trained across levels in the hierarchy to finalize the procedure (Sect. 3.4).

3.2 Modeling Heterogeneous Regions

We denote by $\mathcal{K}$ the set of possible labels, L the number of levels in the hierarchy, $\mathcal{T}$ the set of training images, $\mathcal{I}_I$ the image data for image I, $\mathcal{R}_I$ its set of regions in the hierarchy, and $\mathcal{R}_{I,\ell}$ the set of regions at level ℓ. For each region $r \in \mathcal{R}_I$, we define Y_r to be the random variable that represents the label of the region. For each level ℓ, we train a probabilistic classifier to match the empirical label distribution of $r \in \mathcal{R}_{I,\ell}$ across all training images. For its simplicity, we use a generalized maximum entropy classifier q_{ϕ_ℓ}, where $\phi_\ell : \mathbb{R}^d \to \mathbb{R}$ is a function that defines the distribution:

$$q_{\phi_\ell}(Y_r = a|\mathcal{I}_I) = \frac{\exp(\phi_\ell(f_I(r,a)))}{\sum_{k\in\mathcal{K}} \exp(\phi_\ell(f_I(r,a)))}, \tag{1}$$

Algorithm 1. `train_maxent`

Inputs: Dataset of region features with true distributions $\mathcal{D} = \{(f_I(r,k), p_{I,r,k})\}_{I,r,k}$ where $p_{I,r,k} = p_I(Y_r = k)$, Step size α_t, Number of iterations T.
$\phi = 0$
for $t = 1 \dots T$ **do**
  $\mathcal{A} = \emptyset$
  for $(f_I(r,k), p_{I,r,k}) \in \mathcal{D}$ **do**
    if $\beta_I(r,k) \neq 0$ **then**
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{(f_I(r,k),\ \beta_I(r,k))\}$
    end if
  end for
  $h_t =$ `train_multi_class_regressor`$(\mathcal{A})$
  $\phi \leftarrow \phi + \alpha_t h_t$ // (or, line-search instead of constant α_t)
end for
Return: MaxEnt classifier ϕ

and $f_I : \mathcal{R}_I \times \mathcal{L} \rightarrow \mathbb{R}^d$ are the feature functions that extract (label-specific) features describing the region from image data $\mathcal{I}_I$, such a texture and color (see Appendix). In the following subsection, we discuss how predictions from parent regions are appended to this vector to model context.

At each level, we match the distributions by minimizing the cross entropy of the empirical label distributions p and the classifier q, which reduces to:

$$\phi_\ell^* = \arg\max_{\phi_\ell} \sum_{I \in \mathcal{T}} \sum_{r \in \mathcal{R}_{I,\ell}} \sum_{k \in \mathcal{K}} p_I(Y_r = k) \log q_{\phi_\ell}(Y_r = k | \mathcal{I}_I). \tag{2}$$

This is a standard maximum log-likelihood estimation where the samples are weighted by $p_I(Y_r = k)$, *i.e.*, the number of pixels labeled k in the region divided by its area. The optimization may be performed through standard convex optimization (*e.g.*, gradient ascent) and provides competitive performance in our experiments; however, we found using a non-linear model further improves performance. We train a non-linear model in a boosting manner through Euclidean functional gradient ascent [24]; the following describes the optimization but it is not specific to our hierarchical procedure.

The functional gradient of the inner term in (2) is $\beta_I(r,k)\delta_{f_I(r,k)}$, where

$$\beta_I(r,k) = p_I(Y_r = k) - q_{\phi_\ell}(Y_r = k | \mathcal{I}_I), \tag{3}$$

and δ_x is the Dirac delta function centered at feature value x. As a form of boosting, we train a new function h to match the functional gradient residuals and add it to ϕ. The residuals indicate how to update the function when evaluated at the respective feature locations so that the predicted and ground truth distributions match. We repeat this procedure until convergence and then return $\phi = \sum_t \alpha_t h_t$, where α_t is the step size. We refer to [24] for more details. The training algorithm is given in Algorithm 1. In our experiments we train a separate Random Forest for each class as the multi-class regressor h.

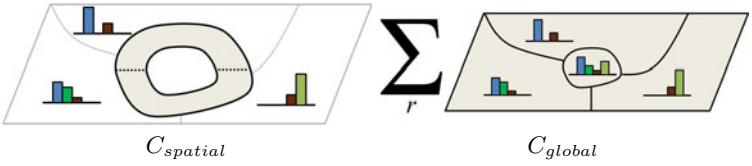


Fig. 4. Illustration of the context features described in Sect. 3.3. Gray indicates the pixels that are being used to compute the feature.

Partial Labelings. Ideally all pixels in the training set are assigned a label; however, most datasets contain many images with unlabeled pixels (such as MSRC-21). We assume that if a class is labeled in the image, then all instances of that class are labeled in that image. In the case a region r is partially labeled, we propose for the classifier to match the proportions of the classes actually present ($\hat{\mathcal{K}}_r$) and to not penalize the predictions made for the classes not present ($\bar{\mathcal{K}}_r$), as the unlabeled pixels may actually contain classes from $\mathcal{K}$. We do this by treating (2) as a negative loss function and by only penalizing the terms with labels in $\hat{\mathcal{K}}_r$ and ignoring the remaining labels, *i.e.*, setting $p_I(Y_r = a) = 0, \forall a \in \bar{\mathcal{K}}_r$ discards losses over the labels not present.

3.3 Context Features

In addition to the image features computed at each level, we need to define the information that is passed from one classifier to the next. It is this information that ties together the individual classifiers trained at each level to yield the global image interpretation generator. Intuitively, using the label distribution predicted by the parent's classifier will make training the child's level distribution predictor an easier problem. At each level, we receive probabilities from the parent level regions. Since this information is of variable length per image, specifically $|\mathcal{R}_{I,\ell-1}| \times |\mathcal{K}|$, we need to summarize it into a fixed-length vector that can be used as input to a generic classifier. For each region in $\mathcal{R}_\ell$, we define three types of contextual features that are computed using the predictions from the regions in $\mathcal{R}_{\ell-1}$. For the first, each region simply uses its parent region's label predictions ($C_{parent} \in \mathbb{R}^{|\mathcal{K}|}$). The next two are illustrated in Fig. 4. The second is the weighted average of the neighboring region's probabilities. The weights are the areas of the region's dilated mask that overlaps with the respective neighbors. In order to describe spatial layout, we compute the averages above and below the region separately ($C_{spatial} \in \mathbb{R}^{2|\mathcal{K}|}$). The third is the weighted average (by size) of the probabilities across all regions in the parent level ($C_{global} \in \mathbb{R}^{|\mathcal{K}|}$); this feature is duplicated for all regions in the current level. These context features are then appended to the respective region's f_I image-based feature vector.

3.4 Hierarchical Stacking

The MaxEnt classifier described in Sect. 3.2 is the basic component used at each level in the hierarchy. Collectively training the classifiers is prone to two problems

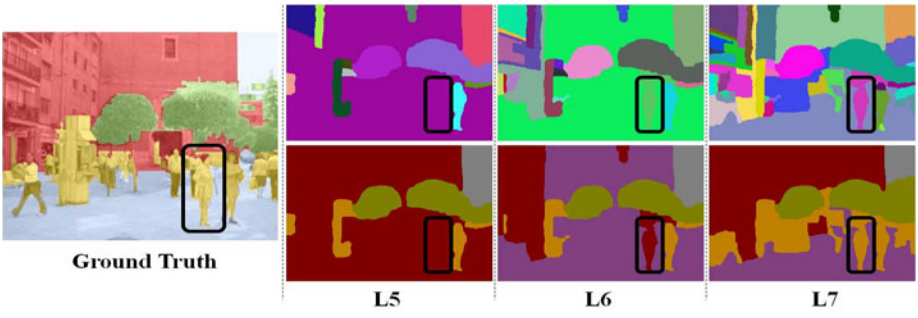


Fig. 5. Example of test-time error recovery. Left: test image with ground truth overlaid. Top: segmentation maps for L5, L6, L7. Bottom: most likely label per region.

of cascading errors. *First*, if we train each level's classifier independently using the parent regions' ground truth, we will have cascading errors at test-time due each classifier being trained with perfect contextual information. Therefore, we have to train the hierarchical procedure in the same way it is executed at test-time: in sequence, using the predictions from the previous level. After predicting the label distributions for each region in a level, we pass this information to the child level, similar to what is done during inference over a graphical model. Similar to other hierarchical methods [18], we pass these predicted per-class probabilities for each region as a vector from which the children construct the context features as described above. Ideally, high levels in the hierarchy can represent the type of environment which "primes" the lower levels with a smaller set of labels to consider. *Second*, now using predictions from the same data used for training is prone to a cascade of errors due to overfitting as subsequent levels will rely heavily on still optimistically correct context. While parent predictions are important, we also want to learn how to recover from mistakes that will be made at test time by trading off between the parent probabilities and image features. To achieve this robust training, we use the idea of stacking [30,4] when training the classifier. Figure 7 illustrates how stacking addresses the overfitting behavior on the MSRC-21 dataset.

Stacking trains a sequence of classifiers where the outputs of one classifier are treated as additional features and are used to train another classifier. In order to avoid overfitting, the outputs are predicted on data that was not used to train the classifier. Obtaining held-out predictions is achieved in a manner similar to cross-validation where the training data is split into multiple subsets that multiple classifiers train on. Because the predictions are made on unseen data, the procedure simulates the test-time behavior and ideally learns how to correct earlier mistakes. An example of this correcting behavior during test-time is illustrated in Fig. 5. In L5, the person is part of a very large region for which label *building* is most confident. In L6, the person is segmented out from its large parent region; however, the most likely label for this region incorrectly follows from the parent's label (*building*). In L7, the region recovers from this error and is correctly labeled *foreground*.

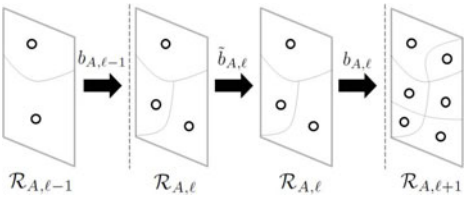


Fig. 6. Predictions between levels during learning/inference for image A

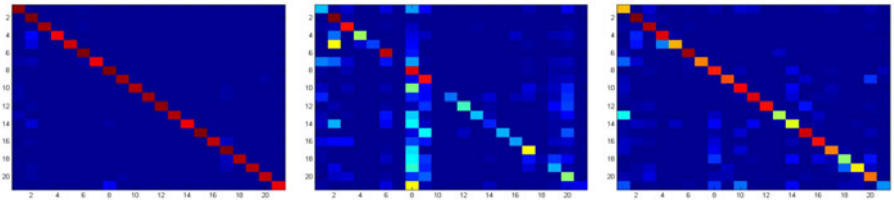
We now describe the stacking procedure in detail (Fig. 6). For each image $I \in \mathcal{T}$, we receive the predictions for each parent region in $\mathcal{R}_{I,\ell-1}$; we denote this $|\mathcal{R}_{I,\ell-1}| \times |\mathcal{K}|$ set of predictions per image I as $b_{I,\ell-1}$. Using $b_{I,\ell-1}$, we compute the context features (Sect. 3.3) for each region in $\mathcal{R}_{I,\ell}$ and append them to its image features f_I. We then generate held-out predictions for all regions at level ℓ (across all training images) by training temporary classifiers on subsets of regions and predicting on the held-out regions. That is, to generate the predictions for regions $\mathcal{R}_{A,\ell}$ in image A, we train a classifier $\tilde{\phi}_{A,\ell}$ over the regions $\cup_{I \in \mathcal{T} \setminus A} \mathcal{R}_{I,\ell}$ and then classify the held-out regions $\mathcal{R}_{A,\ell}$ to generate predictions $\tilde{b}_{A,\ell}$. This process is repeated $|\mathcal{T}|$ times to generate predictions across all images[2]. This stacking procedure is done solely during training to generate predictions to compute the context features. Therefore, we train a final classifier $\tilde{\phi}_\ell$ across all regions at level ℓ to be used at test time. The main idea is that the temporary classifiers simulate the behavior $\tilde{\phi}_\ell$ will have on the unseen test data. Since these classifiers use predictions from the parent level, we refer to them as inter-level classifiers.

One potential problem occurs when a large region at level $\ell - 1$ is split into many small regions at level ℓ. In that case, the context feature $C_{spatial}$ for most of the offspring regions is uninformative because it uses the predictions only from the one parent region without capturing any context. To address this problem, we apply a second round of stacking. In that second round, a new classifier is learned and new predictions $b_{I,\ell}$ are generated by using the same procedure as described above, with the one critical difference that $C_{spatial}$ is computed by using the predictions at level ℓ generated from the classifier learned in the first round, $\tilde{b}_{I,\ell}$, rather than by using the predictions from the previous level, $b_{I,\ell-1}$. In addition, we also append each region's respective prediction from the first round, $\tilde{b}_{I,\ell}$. The resulting set of predictions $b_{I,\ell}$ from this intra-level stacking are then passed to the next level and classifier ϕ_ℓ is saved for test-time. The two-stage process is then repeated for the next level. In practice, we do not do the second stage at the top level.

3.5 Inference

Given a test image I and its hierarchy of regions, inference proceeds in the same cascading manner. At level ℓ, we receive the parent level probabilities $b_{I,\ell-1}$ to

[2] In practice, we hold out 10% of the training images instead of just one.



(a) (b) (c)

Fig. 7. Confusion matrices on MSRC-21 dataset. Performance on training set without stacking (a), and performance on testing set without (b) and with (c) stacking.

create the context features and then use the inter-level classifier $\tilde{\phi}_\ell$ to predict $\tilde{b}_{I,\ell}$. Next, we use $b_{I,\ell-1}$ and $\tilde{b}_{I,\ell}$ to create the same context features from the second stage and then predict $b_{I,\ell}$ with the intra-level classifier ϕ_ℓ. Therefore, performing inference over the hierarchy requires $2L-1$ predictions (since there are no intra-level predictions for the first level).

4 Experiments

We evaluate our algorithm on the MSRC-21 and Stanford Background datasets and demonstrate that we can achieve high performance predictions as with other structured models, even though we never explicitly model global configurations. In both experiments we use the same set of standard image features, mostly computed from the STAIR Vision Library [9], and the same set of learning parameters used to train the hierarchy; see the Appendix for specific details.

4.1 MSRC-21

The MSRC-21 dataset [26] contains a variety of outdoor environments with 21 possible classes; we use the standard evaluation split from [26]. Although not ideal for our hierarchical regions, we use the image-based region features from the flat CRF model of [7] and demonstrate favorable quantitative performance compared to this and other similar recent work. As illustrated in Table 1, we compare with related models that are structured [31,20], use hierarchical regions [21], and sequentially trained (over sites) [27]; "Hier." is our hierarchical approach and "Leaf" is a site-wise classifier trained only over the leaf regions without any context. Although the hierarchical CRF model of [20] demonstrates superior performance, it should be noted that their pixel-wise classifier can obtain an overall accuracy of 80%, which suggests the use of much more discriminative features. In Fig. 8, we quantify the hierarchy's refinement in labeling by plotting, at each level, the accuracies if we assign the regions' pixels their most probable label.

Table 1. Performances on the MSRC-21 dataset. *Overall* is the total number of pixels correct and *Average* is the mean across the columns. *Averaged over 5 different splits.

	Overall	*Average*	Building	Grass	Tree	Cow	Sheep	Sky	Airplane	Water	Face	Car	Bicycle	Flower	Sign	Bird	Book	Chair	Road	Cat	Dog	Body	Boat
[7]*	77	64	72	95	81	66	71	93	74	70	70	69	72	68	55	23	83	40	77	60	50	50	14
[31]	75	65	77	93	70	58	64	92	57	70	61	69	67	74	70	47	80	53	73	53	56	47	40
[21]	–	67	30	71	69	68	64	84	88	58	77	82	91	90	82	34	93	74	31	56	54	54	49
[27]	75	69	69	96	87	78	80	95	83	67	84	70	79	47	61	30	80	45	78	68	52	67	27
[20]	86	75	80	96	86	74	87	99	74	87	86	87	82	97	95	30	86	31	95	51	69	66	9
Leaf	74	60	72	96	85	74	70	91	63	58	65	59	69	58	32	22	84	25	83	55	33	54	4
Hier.	78	71	63	93	88	84	65	89	69	78	74	81	84	80	51	55	84	80	69	47	59	71	24

Fig. 8. Accuracies when assigning regions, at each level, their most probable label

4.2 Stanford Background Dataset

We also evaluate our approach on the recent dataset from [8]. This dataset contains densely labeled images containing eight semantic labels. All results were averaged over five random trials, using the splits described in [8].

Table 2 contains the performances of two structured models and our hierarchical approach. We achieve comparable performance with the global energy model used in [8] while never explicitly modeling the global configurations. Holding segmentation and *image* feature extraction time constant, our hierarchical inference typically takes 12 s/image (10 s of which is spent on computing the *contextual* features), whereas the global energy approach can widely vary from 30 s to 10 min to converge. In Fig. 8, we see a similar label refinement.

4.3 Confident Predictions

Another benefit of our approach over MAP inference techniques (*e.g.*, graph-cuts) is that we never make hard decisions and always predict a distribution of labels. Therefore, when eventually assigning a label to a region, we can extract a notion of confidence in the labeling. We define a labeling as *confident* when the most likely label is 0.2 higher than the runner-up, and otherwise *uncertain*. For example, in Fig. 9, the cars are *confident* in the labeling, but the trees in front of the building are *uncertain*. On MSRC-21, our *confident* predictions constitute 79% of the data and achieve an overall accuracy of 89%, while the *uncertain*

Table 2. Performances on the Stanford Background Dataset

	Overall	*Average*	Sky	Tree	Road	Grass	Water	Bldg.	Mtn.	Fgnd.
[8] Pixel CRF	74.3	66.6	93.9	67.1	90.3	83.3	55.4	71.4	9.3	62.2
[8] Region Energy	76.4	65.5	92.6	61.4	89.6	82.4	47.9	82.4	13.8	53.7
Leaf	72.8	58.0	89.7	58.3	85.8	69.8	15.8	78.1	1.5	64.9
Hierarchy	76.9	66.2	91.6	66.3	86.7	83.0	59.8	78.4	5.0	63.5

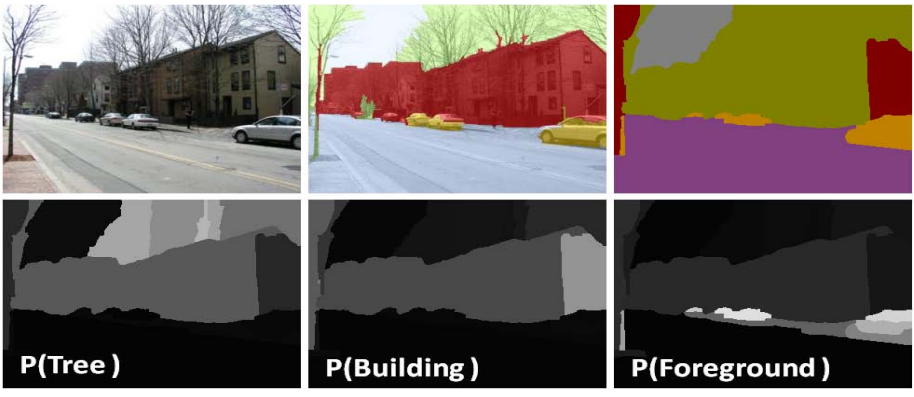


Fig. 9. The ambiguity in ground truth label (top, middle) is correctly modeled in our predictions (bottom row), resulting in a labeling for the building that is *uncertain*

accuracy is 37%. On SBD, our *confident* predictions constitute 87% of the data and achieve an overall accuracy of 82%, while the *uncertain* accuracy is 40%. These numbers indicate that we make most errors when the labeling is *uncertain*.

5 Conclusion

We propose an alternative to the graphical model formulation for structured prediction in computer vision. Our approach is based on training a sequence of simple subproblems that are designed to use context, bypassing the difficulties of training typical structured models. Specifically, we designed an algorithm to train these subproblems in a hierarchical procedure that a) captures the context over large regions b) explicitly models that regions contain mixed labels and c) is trained to follow the same procedure during test-time. Our experiments demonstrate this simple approach is able to capture context and make high performance predictions without a probabilistic model over global configurations.

Acknowledgments. D. Munoz is supported by a QinetiQ North America Robotics Fellowship. This work is supported in part by the ONR MURI grant

N00014-09-1-1052, Reasoning in Reduced Information Spaces. We thank A. Grubb and S. Ross for valuable discussions and B. Becker and A. Collet for CPU resources.

References

1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: From contours to regions: An empirical evaluation. In: CVPR (2009)
2. Barbu, A.: Training an active random field for real-time image denoising. IEEE Trans. on Image Processing 18(11) (2009)
3. Bouman, C.A., Shapiro, M.: A multiscale random field model for bayesian image segmentation. IEEE Trans. on Image Processing 3(2) (1994)
4. Cohen, W.W., Carvalho, V.R.: Stacked sequential learning. In: IJCAI (2005)
5. Daume III, H., Langford, J., Marcu, D.: Search-based structured prediction. Machine Learning Journal 75(3) (2009)
6. Feng, X., Williams, C.K.I., Felderhof, S.N.: Combining belief networks and neural networks for scene segmentation. IEEE T-PAMI 24(4) (2002)
7. Gould, S., Rodgers, J., Cohen, D., Elidan, G., Koller, D.: Multi-class segmentation with relative location prior. IJCV 80(3) (2008)
8. Gould, S., Fulton, R., Koller, D.: Decomposing a scene into geometric and semantically consistent regions. In: ICCV (2009)
9. Gould, S., Russakovsky, O., Goodfellow, I., Baumstarck, P., Ng, A.Y., Koller, D.: The stair vision library, v2.3 (2009), `http://ai.stanford.edu/~sgould/svl`
10. Heitz, G., Gould, S., Saxena, A., Koller, D.: Cascaded classification models: Combining models for holistic scene understanding. In: NIPS (2008)
11. Kakade, S., Teh, Y.W., Roweis, S.: An alternate objective function for markovian fields. In: ICML (2002)
12. Kohli, P., Ladicky, L., Torr, P.H.: Robust higher order potentials for enforcing label consistency. IJCV 82(3) (2009)
13. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE T-PAMI 26(2) (2004)
14. Komodakis, N., Paragios, N., Tziritas, G.: Mrf energy minimization and beyond via dual decomposition. IEEE T-PAMI (in press)
15. Kou, Z., Cohen, W.W.: Stacked graphical models for efficient inference in markov random fields. In: SDM (2007)
16. Kulesza, A., Pereira, F.: Structured learning with approximate inference. In: NIPS (2007)
17. Kumar, S., August, J., Hebert, M.: Exploiting inference for approximate parameter learning in discriminative fields: An empirical study. In: Rangarajan, A., Vemuri, B.C., Yuille, A.L. (eds.) EMMCVPR 2005. LNCS, vol. 3757, pp. 153–168. Springer, Heidelberg (2005)
18. Kumar, S., Hebert, M.: A hierarchical field framework for unified context-based classification. In: ICCV (2005)
19. Kumar, S., Hebert, M.: Discriminative random fields. IJCV 68(2) (2006)
20. Ladicky, L., Russell, C., Kohli, P., Torr, P.: Associative hierarchical crfs for object class image segmentation. In: ICCV (2009)
21. Lim, J.J., Arbelaez, P., Gu, C., Malik, J.: Context by region ancestry. In: ICCV (2009)

22. Maire, M., Arbelaez, P., Fowlkes, C., Malik, J.: Using contours to detect and localize junctions in natural images. In: CVPR (2008)
23. Ohta, Y., Kanade, T., Sakai, T.: An analysis system for scenes containing objects with substructures. In: Int'l. Joint Conference on Pattern Recognitions (1978)
24. Ratliff, N., Silver, D., Bagnell, J.A.: Learning to search: Functional gradient techniques for imitation learning. Autonomous Robots 27(1) (2009)
25. Ross, S., Bagnell, J.A.: Efficient reductions for imitation learning. In: AIStats (2010)
26. Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. IJCV 81(1) (2009)
27. Tu, Z., Bai, X.: Auto-context and its application to high-level vision tasks and 3d brain image segmentation. T-PAMI 18(11) (2009)
28. Viola, P., Jones, M.J.: Robust real-time face detection. IJCV 57(2) (2004)
29. Wainwright, M.J.: Estimating the "wrong" graphical model: Benefits in the computation-limited setting. JMLR 7(11) (2006)
30. Wolpert, D.H.: Stacked generalization. Neural Networks 5(2) (1992)
31. Zhang, L., Ji, Q.: Image segmentation with a unified graphical model. T-PAMI 32(8) (2010)

A Image Features

For the top-level in the hierarchy, we use Gist[3] computed from 64x64 rescaled images at 2 scales with 8 and 4 orientations, Pyramid Histogram of Oriented Gradients[4] with 2 levels, 8 histogram bins and 4 orientations, and a color histogram over $CIELab$ colorspace with 10 bins over L and 20 bins over a and b along with the mean and std. per channel. For the remaining levels in the hierarchy, we primarily use the region appearance features from [7,9]. These features consist of filters, color and bounding box statistics, location, and the weighted average of neighboring regions' features. In addition, we also count the number of vertices on the convex hull enclosing each region and use a hierarchy-based descriptor to model relative relocation. This descriptor consists of the orientation ($[-\pi, \pi]$) and length of the vector extending from the centroid of the parent to the child, normalized by the diagonal of the parent's bounding box.

B Hierarchy

The hierarchy is created by thresholding the scale value from 256 at an interval of -30. During functional gradient boosting, the step size is $\alpha_t = \frac{1.5}{\sqrt{t}}$ for each level while we increase the number of iterations T at each level to handle the increasing amount of data as regions were split: 10, 12, 15, 17, 20, 20, 25, 30. The Random Forest[5] regressor consisted of 10 trees and each tree required at least 15 samples to split a node. We found the entire process is resilient to changes in these parameters.

[3] http://lear.inrialpes.fr/software
[4] http://www.robots.ox.ac.uk/~vgg/research/caltech/phog.html
[5] http://code.google.com/p/randomforest-matlab/