Abstract

Two fundamental problems in the context of probabilistic graphical models are learning and inference. Many traditional probabilistic methods resort to approximations in either learning, inference, or even both steps due to their large complexity cost. This leads to algorithms where the learning and inference steps are disjoint, which often degrades algorithm performance. In this work, we present a Local Perturb-and-MAP (locPMAP) method, a novel framework for structured prediction based on local optimization over randomly perturbed potential functions. Unlike most prior methods, our proposed scheme does not rely on approximations and also unifies the learning and inference steps. This allows our approach to outperform other methods on several vision tasks, where structured models are commonly used. Additionally, we demonstrate how to apply our proposed scheme for an end-to-end training of a deep structured network. Finally, our framework gives a novel interpretation for pseudolikelihood, enabling various powerful extensions, such as the use of pseudolikelihood under the Bayesian framework.

1 Introduction

Probabilistic graphical models, such as Markov random fields, and conditional random fields provide a powerful framework for solving challenging learning problems that require structured output prediction. The use of graphical models consists of two main steps: learning, which estimates the model parameters from the data, as well as inference, which makes predictions based on the learned model. Unfortunately, both maximum likelihood learning and probabilistic inference involve calculating the normalization constant (also known as partition function), which is intractable to compute in general. In practice, approximation methods, such as variational inference and MCMC, are widely used for inference and learning. There also exist alternative consistent learning methods such as the maximum pseudo-likelihood (PL) estimator [2] which does not require calculating likelihood and is computationally tractable.

It is well known that there are strong interactions between learning and inference. As a result, the choice of the learning and inference algorithms should be considered jointly, an idea which is referred to as “inferning” [1]. Although it is relatively easy to identify “inferning” pairs when using variational or MCMC approximations, it is unclear what the natural inference counterpart of pseudo-likelihood (PL) is. For instance, even if PL learning estimates a true model, a poor choice of a subsequent approximate inference algorithm may deteriorate the overall prediction accuracy.

An alternative way to achieve “inferning” is to employ models that are computationally more tractable. One such framework is the Perturb-and-MAP model [13, 16] which involves injecting noise into the

[1] see [http://inferning.cs.umass.edu](http://inferning.cs.umass.edu)
log-probability (the potential function), and generating random samples to find the global maximum, i.e., the maximum a posteriori (MAP) estimate of the perturbed probability.

These models have a sound probabilistic interpretation, which can be exploited to make predictions with an uncertainty measure. Another benefit is that models from the Perturb-and-MAP class require combinatorial optimization that can be solved easier than probabilistic inference models that require marginalizing over variables or drawing MCMC samples.

Unfortunately, despite being easier than marginalization, MAP estimation still requires a global optimization that is generally NP-hard. Thus, this prevents using perturb-and-MAP for generic graphical models. In addition, the perturb-and-MAP model can be viewed as a hidden variable model with deterministic constraints, and its training casts another challenging learning task that involves maximizing a non-convex likelihood function.

In this work, we propose “Local Perturb-and-MAP” (locPMAP) which only requires finding a local maximum of the perturbed potential function. We show that by employing random perturbations over the potential functions, the likelihood of our locPMAP model becomes the same as the pseudo-likelihood of the original graphical model. As a result, our locPMAP inference method forms an “inferning” counterpart for the pseudo-likelihood learning objective, which allows us to avoid approximations in the inference step. This new interpretation of PL as a true likelihood also opens up opportunities for many useful extensions, such as the use of PL under the Bayesian framework.

We test our approach on two different vision tasks. We show that the consistency between learning and inference of our method yields better results than those achieved using approximate inference techniques, on both of these tasks. In addition, we demonstrate that we can integrate our method in the fully convolutional network framework [11] to address the complexity limitation of log-linear CRF models and further improve our performance on challenging structured-prediction problems.

2 Related Work

The idea of perturb-and-MAP was motivated by the classical “Gumbel-max-trick” that connects logistic function with discrete choice theory. It was first applied to graphical model settings by [13] and [8]. This work gave rise to a rich line of research [9, 6, 16]. We remark that these methods all require difficult global optimization, in contrast with the local optimization in our method.

The idea of “inferning”, which enforces the consistency between learning and inference, was discussed by [17], who showed that when exact inference is intractable, it is better to learn a “wrong” model to compensate the errors made by the subsequent approximate inference methods. A line of work has been developed to explicitly tune parameters in approximate inference procedures [12] [5].

3 Background

In subsection 3.1 we introduce some background information on conditional random fields (CRFs). Additionally, in subsection 3.2, we present some key ideas related to the Gumbel-Max trick and Perturb-and-MAP. We will use all of these ideas to introduce our method in Section 4.

3.1 Structured Prediction with CRFs

CRFs provide a framework for solving challenging structured prediction problems. Let $x$ be an input (e.g., an image), and $y$ a set of structured labels (e.g., a semantic segmentation). A CRF assumes that the labels $y$ are drawn from an exponential family distribution

$$p(y|x; w) = \frac{1}{Z(\theta)} \exp(\theta(y, x, w)) \quad (1)$$

where $\theta$ is a potential function and $w$ are model parameters that need to be estimated from data; the normalization constant $Z(x, w) = \sum_y \exp(\theta(y, x, w))$ is difficult to compute unless the corresponding graph is tree structured.
Now assume that we are given a set of annotated training data examples \( \{x^i, y^i\} \). A typical maximum likelihood estimator learns parameters \( w \) by maximizing the log likelihood function:

\[
\hat{w} = \arg \max_w \sum_i \log p(y^i|x^i; w). \tag{2}
\]

With the estimated parameters \( \hat{w} \) we can then make predictions for a new testing image \( x^* \):

\[
\hat{y} = \arg \max_y p(y|x^*; \hat{w}) = \arg \max_y \theta(y, x^*; \hat{w}). \tag{3}
\]

However, both the learning and inference steps in Equations \(2\)–\(3\) are computationally intractable for general loopy graphs. Instead, a popular computationally efficient alternative for MLE is the pseudolikelihood (PL) estimator \( [2]\), defined as:

\[
\hat{w} = \arg \max_w \sum_i \sum_j \log p(y^i_j | x^i; \hat{w}), \tag{4}
\]

where \( \neg j \) refers to the neighborhood of the nodes in the graph that are connected to the node \( j \). Based on this formulation, each conditional likelihood does not involve \( Z \), and can be calculated efficiently. The PL is known to be an asymptotically consistent estimator, meaning that \( \hat{w} \) approaches the true parameter \( w \) as the size of the dataset is increased.

However, even if PL estimates the true parameters perfectly, the prediction step in \(3\) still requires approximation. Iterated Conditional Modes (ICM) is one of the simplest inference algorithm that returns a local maximum from the neighborhood of nodes. Other widely used inference techniques include loopy belief propagation (LBP), the mean field algorithm (MF), and Gibbs sampling.

The problem with using approximate inference algorithms is that they may not work well together with the PL learning algorithm. This is because learning and inference are performed disjointly without considering how one may affect the other.

### 3.2 Perturb-and-MAP

In contrast to CRF, the perturb-and-MAP model \( [8, 13]\) considers distributions of the form

\[
\Pr[y \in \arg \max \{\theta(y) + \epsilon(y)\}], \quad \epsilon \sim q \tag{5}
\]

where we drop the dependency on \( x \) to simplify the notation. That is, we first perturb the potential function \( \theta(y) \) with random noise \( \epsilon(y) \) from distribution \( q \) and then draw the sample by finding the maximum \( y \). Consider special perturbation noise \( \epsilon(y) \) drawn i.i.d. from the zero mean Gumbel distribution with cumulative distribution function \( F(t) = \exp(-\exp(-t+c)) \), where \( c \) is the Euler constant. The Gumbel-max trick shows that the perturb-and-MAP model is then equivalent to the distribution in \( [1]\), that is:

\[
\Pr[y \in \arg \max \{\theta(y) + \epsilon(y)\}] = \frac{\exp(\theta(y))}{\sum_y \exp(\theta(y))}. \tag{6}
\]

This connection provides a basic justification for perturb-and-MAP models.

### 4 Local Perturb-and-MAP Optimization

A major limitation of Perturb-and-MAP, even when using Gumbel noise, is that it requires global optimization over the perturbed potentials. We address this problem by replacing the global optimum with a local optimum. We start by defining the notion of local optimality.

**Definition 4.1.** Let \( \mathcal{B} = \{\beta_k\} \) be a set of non-overlapping sets of variable indices such that \( \beta_k \cap \beta_l = \emptyset \) for \( \forall k \neq l \). Then we say that \( y \in \text{Loc}\{\theta(y); \mathcal{B}\} \)

if \( y_\beta \in \arg \max_{y^i} \theta(y^i | y_\neg\beta) \) for \( \forall \beta \in \mathcal{B} \), where \( \neg \beta = [p] \setminus \beta \). This implies that \( y_\beta, y_\neg\beta \) is no worse than \( y^i_\beta, y^i_\neg\beta \) for any \( y^i_\beta \in Y_\beta, y_\neg\beta \in \mathcal{B}. \) In other words, \( y \) is a block-coordinate-wise maximum of \( \theta(y) \) on the set \( \mathcal{B}. \)
We are now ready to establish our main result. We show that by exploiting random Gumbel perturbations over the potential functions, we can formulate a Local Perturb-and-MAP model, for which the pseudolikelihood function becomes the true likelihood.

**Theorem 4.2.** Let us now perturb $\theta(y)$ to get $\tilde{\theta}(y) = \theta(y) + \sum_{\beta \in B} \epsilon(\beta)$, where each element $\epsilon(\beta)$ is drawn i.i.d. from a Gumbel distribution with CDF $F(t) = \exp(-\exp(-t+c))$, where $c$ is the Euler constant. Then we have

$$\Pr\left(y \in \text{Loc}[\tilde{\theta}(y); B]\right) = \prod_{\beta \in B} p(\beta; y_{-\beta}; \theta).$$

Note that the right hand side has a form of composite likelihood, which reduces to the pseudolikelihood when taking $B = \{k: k \in [n]\}$.

**Proof:** Based on our definition of Loc() we can write $\text{Loc}[\tilde{\theta}(y); B] = \cap_{\beta \in B} A_\beta$, where $A_\beta = \{y: y_\beta \in \arg \max_{y_\beta} [\tilde{\theta}(y_\beta, y_{-\beta}) + \epsilon(\beta)]\}$. Then, we can write:

$$\Pr(y \in A_\beta) = \Pr(y_\beta \in \arg \max_{y_\beta} [\tilde{\theta}(y_\beta, y_{-\beta}) + \epsilon(\beta)])$$

$$= \frac{\exp(\tilde{\theta}(y_\beta, y_{-\beta}))}{\sum_{y_\beta} \exp(\tilde{\theta}(y_\beta, y_{-\beta})}$$

$$= p(y_\beta; y_{-\beta}; \theta)$$

where we use Equation 6 to derive these equalities. Note that Equation 6 results from the application of the Gumbel-Max trick. In the context of our problem, this equation holds because $\epsilon(\beta)$ are drawn i.i.d from a zero mean and a unit variance Gumbel distribution. Additionally, since $\epsilon(\beta)$ are drawn independently from each other, the events $[y \in A_\beta]$ are independent too. Therefore, we can write:

$$\Pr(y \in \text{Loc}[\tilde{\theta}(y); B]) = \Pr(y \in \cap_{\beta \in B} A_\beta)$$

$$= \prod_{\beta \in B} p(y_\beta; y_{-\beta}; \theta).$$

## 5 The locPMAP model based on Random Potential Perturbation & ICM

Our result motivates a new Local Perturb-and-MAP (locPMAP) model for structured prediction, in which we assume that $y$ is generated by the following generative model: 1) Potential functions are perturbed using i.i.d Gumbel noise. 2) The ICM algorithm is applied on these randomly perturbed potential functions and the local maximum is returned for each node.

We outline these two steps in Algorithm[1]. We assume a simple case when $B$ consists of the set of single variables, and hence the likelihood of locPMAP corresponds to the pseudolikelihood of the original model. For simplicity, we drop the dependency on $B$ and use the notation $\text{Loc}[\tilde{\theta}(y, x; w)]$. Given a set of training data $\{x^i, y^i\}$, we then estimate the parameters $w$ by maximizing the likelihood of this model. Using our proof from Section[4] we can write this likelihood as:

$$\hat{w} = \arg \max \sum_i \log \Pr(y \in \text{Loc}[\tilde{\theta}(y, x; w)])$$

$$= \arg \max \sum_i \log \prod_j p(y_j^i | y_{-j}^i; \theta)$$

$$= \arg \max \sum_j \sum_i \log p(y_j^i | x, y_{-j}^i; w)$$

We know that the expression above is equivalent to the pseudolikelihood, which we discussed in Section[5.1]. Thus, to estimate the parameters of the model according to the true likelihood, we can simply maximize the pseudolikelihood function. Note that while for most prior CRF-based methods the pseudolikelihood is considered just a surrogate for the true likelihood function, for our formulated model the pseudolikelihood is the true likelihood function.
Algorithm 1 Generative Model via Random Potential Perturbation & ICM Algorithm

1. Let $\tilde{\theta}(y, x; w) = \theta(y, x; w) + \epsilon(y)$, where $\epsilon(y)$ are drawn i.i.d. from Gumbel $\sim \text{G}(0, 1)$.

2. Run ICM on the perturbed potentials $\tilde{\theta}(y, x; w)$ to get $y_j = \arg \max_y \tilde{\theta}(y_j, y_{-j}, x; w), \forall j$.

Such a formulation then allows our model to eliminate the need for approximate learning and inference methods. Additionally, our model enforces consistency between learning and inference. As a result, after estimating the parameters by maximizing the PL, we can then perform inference by running Algorithm 1, which is consistent with the PL parameter learning.

Because the result of Algorithm 1 is not deterministic, we can repeatedly run it for several iterations, and then take the mode of the returned samples. Due to the consistency with the PL learning such prediction scheme should not only produce good results but, unlike most inference methods, it can also produce probabilistic outputs with error bars indicating the variance of the structured prediction.

6 Learning Deep Unary Features with FCNs and Pseudolikelihood Loss

Background. Under log-linear CRF models, we typically assume that the potential function is a weighted combination of features:

$$\theta^U_i = \exp \sum_j w_{ij}^U f^U_j(x_i) \quad \theta^P_{ij} = \exp \sum_k w_{ik}^P f^P_k(x_i, x_j)$$

where $\theta^U_i$, $w^U_i$ and $f^U(x_i)$ denote unary potentials for node $i$, learnable unary feature parameters, and unary features, respectively. Similarly, $\theta^P_{ij}$, $w^P$ and $f^P(x_i, x_j)$ are pairwise potentials between nodes $i$ and $j$, learnable pairwise feature parameters, and pairwise features, respectively.

However, there are several important limitations related to log-linear CRF models. Such models have only a linear number of learnable parameters, which limits the complexity of the model that can be learned from the data. One way to address this is to construct highly nonlinear and complex features that would work well even with a linear classifier. However, hand-engineering complex features is a challenging and time-consuming task requiring lots of domain expertise. To address both of these limitations, we train a deep network that optimizes a pseudo-likelihood criterion and automatically learns complex unary features for our model.

Recently, deep learning methods have been extremely successful in learning effective hierarchical features that achieve state-of-the-art results on a variety of vision tasks. A particularly useful model for structured prediction on images is the Fully Convolutional Network (FCN) [11] used in combination with CRF models. These models combine the powerful methodology of deep learning for hierarchical feature learning with the effectiveness of CRFs for modeling structured pixel output, such as the class labels of neighboring pixels in semantic segmentation. While in early approaches the FCN and the CRF were learned separately [3], more recently there has been some successful work that has integrated CRF learning into the FCN framework [18]. Additionally, learning the parameters of the CRF in the neural network model has been addressed in [14][4].

However, we note that these approaches use loss functions that are approximations of the true likelihood behind the corresponding CRF model. Instead, in our approach we minimize the loss with respect to a pseudolikelihood, which is the true likelihood of our model and can also be computed exactly. Thus, by using FCNs we increase the complexity of our model, while still maintaining...
consistent learning and inference steps and avoiding any approximation methods. We optimize the entire FCN via backpropagation with respect to the pseudolikelihood loss as explained below.

**Optimizing the Pseudolikelihood Loss.** Let our input be an image of size $h \times w \times c$, where $h, w$ refer to the height and width of the image and $c$ is the number of input channels ($c = 3$ for color RGB images, $c = 1$ for grayscale images). Then assume that our goal is to assign one of $K$ possible labels to each pixel $(i, j)$. The label typically denotes the class of the object located at pixel $(i, j)$ or the foreground/background assignment. Our conditional pseudolikelihood probability is then:

$$p(y(i, j) = l | x, y_{-(i, j)}) = \frac{\exp(\theta_{i,j,l})}{\sum_{k=1}^{K} \exp(\theta_{i,j,k})}$$

(7)

where $\theta_{i,j,l}$ refers to the potential function values for label $l$ at pixel $(i, j)$, and where $-(i, j)$ indicates all the nodes connected to node $(i, j)$. More specifically, $\theta_{i,j,l}$ denotes the product of a unary potential at a node $(i, j)$ and all the pairwise potentials that are connected to the node $(i, j)$. The subscript $l \in \{1, \ldots, K\}$ in the probability notation denotes that this is a potential associated with the class label $l$. Then, to obtain a proper probability distribution we can normalize this potential value as shown in Equation 7. Finally, we can write the loss of our FCN and its respective gradient as:

$$L_{i,j,l} = -\log p(y(i, j) = l | x, y_{-(i, j)}; \theta_{i,j,l})$$

(8)

$$\frac{\partial L_{i,j,l}}{\partial \theta_{i,j,l}} = p(y(i, j) = l | x, y_{-(i, j)}) - 1\{y_{i,j} = l\}$$

(9)

where the last term in the equation is simply an indicator function denoting whether ground truth label $y_{i,j}$ is equal to the predicted label $l$. This gradient is computed for every node $(i, j)$ and then the sum across all the nodes is backpropagated to all layers of the FCN. We provide more details about the training details in the experimental section.

### 7 Experimental Results

In this section, we evaluate our Local Perturb-and-MAP (locPMAP) method results against other inference techniques on two different tasks. In all our experiments, we use the following setup. First, we learn the parameters of a CRF-based model using the PL learning criterion. We note that the PL learning is done once, and the same learned parameters are then used for both our method and the other baseline inference techniques. This is done to illustrate the benefit of consistent learning and inference of our method, and also to compare the performance between our method and inference techniques that are not consistent with the PL learning step. We note that in the case of the baseline methods, the PL learning objective is only an approximation to the true likelihood function. However, for our locPMAP method the PL learning objective is the true likelihood of our model. Our study is aimed to show that by enforcing consistency between learning and inference, our approach can outperform other inference techniques, even methods known for their strong optimization performance.
Since our method relies on ICM to make predictions, we compare our approach with traditional ICM. We also compare against an iterative version of ICM (ICM-iter) which is executed for the same number of iterations as our method (50 iterations), in order to give both methods the same computational budget. In this iterative version of ICM, at each iteration we randomly perturb the potentials by setting a small fraction (e.g., 0.1) of them to zero. In all experiments, for our method and ICM-iter, we only perturb the unary potentials. Additionally, we use a grid-based graph model, with each node connected to its 4 neighbors. The details of the pairwise potentials are discussed separately below for each task. We also tested inference of the learned model using loopy belief propagation (LBP), mean field (MF), simulated annealing (SA) and Gibbs sampling (MCMC). For both tasks we show that our Local Perturb-and-MAP method consistently outperforms other inference techniques, thus demonstrating the benefit of optimizing the true likelihood function, and maintaining consistent learning and inference steps. We now present each of our experiments in more detail.

### 7.1 Denoising Handwritten Digits

For our first evaluation we use the MNIST dataset which contains black and white images of handwritten digits. We corrupt each $28 \times 28$ image using Gumbel noise with 0.25 signal-to-noise ratio. This produces corrupted grayscale images, which are used as input to our system. The objective is to recover the original black/white (background/foreground) value of each pixel.

In our experiments, we used 5000 images for training and 5000 images for testing. We performed two types of experiments on this task. First, we evaluated all methods using the corrupted pixel intensity values as unary features. For pairwise features between nodes $i$ and $j$, we used the corrupted pixel intensity values as unary features. For pairwise features between nodes $i$ and $j$, we used the corrupted pixel intensity values as unary features.

For the second experiment, we trained a fully convolutional network (FCN) to learn the unary features. To optimize the pseudolikelihood criterion, we used pairwise potential parameters that were learned using the corrupted potentials. We kept the pairwise potential parameters fixed and performed gradient backpropagation only through the unary feature parameters.

To train the FCN, we used an architecture composed of 5 convolutional layers with kernel size of $3 \times 3$ for the first four layers and kernel size of $1 \times 1$ for the last layer. The output plane dimensions for the convolutional layers were 64, 126, 256, 512 and 2 respectively. To avoid the reduction in the resolution inside the deep layers, we did not use any pooling layers. We trained our FCN to minimize the pseudolikelihood loss for $\approx 3000$ iterations.

In Table 1, we present quantitative results of our method and several top baseline methods. The performance of each method is evaluated in terms of IoU metric for background and foreground classes. We separately evaluate each baseline inference method using corrupted pixel intensities as unary features (Raw) versus deep unary features learned via FCNs (Deep). We compare the results when using raw corrupted pixel intensities as unary features (Raw) and also using deep features (Deep). The results demonstrate that our locPMAP method outperforms the other baseline inference techniques in both cases.

| Method  | Background | Foreground | Mean |
|---------|------------|------------|------|
| LBP     | 0.846      | 0.854      | 0.846|
| MF      | 0.864      | 0.877      | 0.864|
| ICM     | 0.722      | 0.797      | 0.722|
| ICM-iter| 0.806      | 0.837      | 0.806|
| SA      | 0.730      | 0.807      | 0.730|
| ICM     | 0.840      | 0.840      | 0.840|
| LocPMAP | 0.751      | 0.826      | 0.751|

Table 1: Results of handwritten digit denoising task. Performance is measured according to the Intersection over Union (IoU) for both the foreground and the background mask. We compare the results when using raw corrupted pixel intensities as unary features (Raw) versus deep unary features learned via an FCN (Deep). Our locPMAP method outperforms the other baseline inference techniques in both cases.

In Table 1, we present quantitative results of our method and several top baseline methods. The performance of each method is evaluated in terms of IoU metric for background and foreground classes. We separately evaluate each baseline inference method using corrupted pixel intensities as unary features (Raw) and also using deep features (Deep). The results demonstrate that our locPMAP method outperforms other inference baselines, thus suggesting that consistent learning and inference is critical for good performance. Additionally, we note that learning unary features via FCNs substantially improves the accuracy for most methods. We also present qualitative results in Figure 1.
We note that the idea of “inferning” is still relatively unexplored. In this work, we designed a model that unifies the learning and inference steps for structured prediction by employing random potential function perturbations and a local ICM inference method. Our model is formulated in such a way that the pseudolikelihood learning objective is the true likelihood of our model. We have shown that the consistency between learning and inference in the objectives optimized by our model, leads to improved results compared to other popular inference methods on two vision tasks. Additionally, we have demonstrated that using the pseudolikelihood objective, we can learn effective deep unary features using fully convolutional networks. This allows us to address the complexity limitation of log-linear CRF models, and also to avoid the costly feature engineering process.

We note that the idea of “inferning” is still relatively unexplored. In this work, we designed a model that gives a novel interpretation for the pseudolikelihood learning objective, avoids approximations and also introduces a consistent learning and inference framework. Our proposed model leaves space for useful extensions. For instance, the fact that the pseudolikelihood becomes the true likelihood of our model, allows us to reason about our model in Bayesian terms, and thus possibly build more successful models. Additionally, we note that ICM is one of the simplest inference techniques. Due to the limited space, we only included the results of the top baseline methods.

### 7.2 Scene Labeling

As our other task, we consider the problem of scene labeling. Our goal is to assign every pixel to one of 8 possible classes: sky, tree, road, grass, water, building, mountain, and a foreground object. For this task, we use the Stanford background dataset [7], which has per-pixel annotations for a total of 715 scene color images of size $240 \times 320$. In this case the input to the system is the RGB image and the objective is to label every pixel in the scene with a correct class. We split the dataset randomly and use 600 images for training and the remaining 115 images for testing.

Once again we perform two experiments for this task. First, we use the boosted unary potentials provided by [7] as the unary features in our CRF model. Next, to construct pairwise potentials we extract HFL boundaries [1] from the images. We then compute the gradient on the boundaries, and use it as pairwise pixel features. Then, we learn CRF parameters by optimizing the pseudolikelihood objective, and perform the inference using the learned parameters.

For the second experiment, we learn deep features using fully convolutional network architecture based on DeepLab [3]. During training, we fix the pairwise potential parameters, and only learn the parameters associated with the unary terms. After the unary learning is done, we learn new pairwise parameters given the learned unary features.

In Table 2 we present our results for the scene labeling. The results show that our locPMAP method outperforms other inference methods in both scenarios: using boosted [7] and also using our learned deep features. In Figure 2, we also show our qualitative results. Note that compared to the results achieved by [7], our predictions are spatially smoother. Similarly, relative to the ICM predictions, our Local Perturb-and-MAP results have crispier boundaries around the objects and are also more spatially consistent. We also note that unlike most inference methods, which can only predict the discrete label, our method also outputs the prediction confidence and variance for every pixel (See Figure 5).

### 8 Discussion

In this work, we introduced Local Perturb-and-MAP (LocPMAP), a method that unifies the learning and inference steps for structured prediction by employing random potential function perturbations and a local ICM inference method. Our model is formulated in such a way that the pseudolikelihood learning objective is the true likelihood of our model. We have shown that the consistency between learning and inference in the objectives optimized by our model, leads to improved results compared to other popular inference methods on two vision tasks. Additionally, we have demonstrated that using the pseudolikelihood objective, we can learn effective deep unary features using fully convolutional networks. This allows us to address the complexity limitation of log-linear CRF models, and also to avoid the costly feature engineering process.

We note that the idea of “inferning” is still relatively unexplored. In this work, we designed a model that gives a novel interpretation for the pseudolikelihood learning objective, avoids approximations and also introduces a consistent learning and inference framework. Our proposed model leaves space for useful extensions. For instance, the fact that the pseudolikelihood becomes the true likelihood of our model, allows us to reason about our model in Bayesian terms, and thus possibly build more successful models. Additionally, we note that ICM is one of the simplest inference techniques. Due to the limited space, we only included the results of the top baseline methods.
to its simplicity, it would be beneficial to develop a model that employs a more advanced inference techniques and still has consistent learning and inference steps. Finally, for many applications pseudolikelihood may not be an appropriate loss function. In future work we will consider more complex loss functions to be integrated in our framework.

References

[1] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision. In The IEEE International Conference on Computer Vision (ICCV), December 2015.

[2] Julian Besag. Statistical analysis of non-lattice data. Journal of the Royal Statistical Society, Series D (The Statistician), 24(3):pp. 179–195, 1975.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In ICLR, 2015.

[4] Trinh-Minh-Tri Do and Thierry Artieres. Neural conditional random fields. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, volume 9, 5 2010.

[5] Jens Domke. Learning graphical model parameters with approximate marginal inference. volume 35, pages 2454–2467. IEEE, 2013.

[6] Andreea Gane, Tamir Hazan, and Tommi S. Jaakkola. Learning with maximum a-posteriori perturbation models. In International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014, pages 247–256, 2014.

[7] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In Proceedings of the International Conference on Computer Vision (ICCV), 2009.

[8] Tamir Hazan and Tommi S. Jaakkola. On the partition function and random maximum a-posteriori perturbations. In Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012, 2012.

[9] Tamir Hazan, Subhransu Maji, Joseph Keshet, and Tommi Jaakkola. Learning efficient random maximum a-posteriori predictors with non-decomposable loss functions. In Advances in Neural Information Processing Systems 26, pages 1887–1895. Curran Associates, Inc., 2013.

[10] Alexander Kirillov, Dmitrij Schlesinger, Walter Forkel, Anatoly Zelenin, Shuai Zheng, Philip H. S. Torr, and Carsten Rother. Efficient likelihood learning of a generic CNN-CRF model for semantic segmentation. CoRR, abs/1511.05067, 2015.

[11] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. CVPR, November 2015.

[12] Ofer Meshi, David Sontag, Tommi Jaakkola, and Amir Globerson. Learning efficiently with approximate inference via dual losses. International Machine Learning Society, 2010.

[13] George Papandreou and Alan L. Yuille. Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc J. Van Gool, editors, ICCV, pages 193–200. IEEE, 2011.

[14] Jian Peng, Liefeng Bo, and Jinbo Xu. Conditional neural fields. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, Advances in Neural Information Processing Systems 22, pages 1419–1427. Curran Associates, Inc., 2009.

[15] Veselin Stoyanov and Jason Eisner. Minimum-risk training of approximate crf-based nlp systems. In 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 120–130, 2012.

[16] Daniel Tarlow, Ryan Prescott Adams, and Richard S. Zemel. Randomized optimum models for structured prediction. In Neil D. Lawrence and Mark Girolami, editors, AISTATS, volume 22 of JMLR Proceedings, pages 1221–1229. JMLR.org, 2012.

[17] Martin J. Wainwright. Estimating the wrong graphical model: Benefits in the computation-limited setting. Journal of Machine Learning Research, 7:1829–1859, 2006.

[18] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. Conditional random fields as recurrent neural networks. In International Conference on Computer Vision (ICCV), 2015.