Constrained Optimization of Latent Distributions to Support Neural Network Training

CS164 Final Project
Yoav Rabinovich

April 2020

---

1 Problem Description

1.1 Introduction

This paper presents the technique described in Pathak, Krähenbühl, and Darrell 2015 for augmenting neural networks for weak-labeled image segmentation problems with domain-knowledge constraints to a derived latent distribution. It also demonstrates the principle by performing one optimization step on an under-trained network and comparing the resultant latent distribution to a fully trained network output.

1.2 Weak-Labeled Image Segmentation

The use-case for the technique described in Pathak, Krähenbühl, and Darrell 2015 is augmenting the ability of neural networks to segment images pixel-by-pixel into classes describing the represented objects, while being trained on weak labels, containing only a list of objects present in each image (tags). While state-of-the-art UNet architectures trained on ground-truth pixel-level labels are capable of reaching great accuracy on published image segmentation data-sets (Ronneberger, Fischer, and Brox 2015), labels of this resolution are costly, and weak labels (such as image-level tags) provide a great obstacle for inference.

1.3 Linear Constraints on a Latent Distribution

Simple assumptions or observations based on domain-knowledge can introduce information that can theoretically be used to great effect by the network, but there is no standard, simple way to introduce such insights into training. Pathak, Krähenbühl, and Darrell 2015 introduce a technique to incorporate domain-based knowledge to neural network through a set of linear constraints. These constraints are difficult to impose on network output

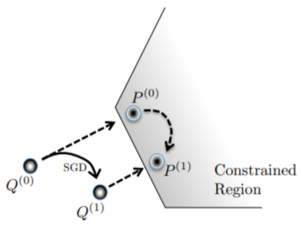

Figure 1: From Pathak, Krähenbühl, and Darrell 2015. In the space of probability distributions of segment classes, with KL divergence defined as a distance function or metric, each training step of the network leading to an output distribution $Q^{(t)}$ is followed by an optimization step, in which a derived latent distribution $P^{(t)}$ is optimized to minimize distance to the fixed output distribution. Then, with a fixed latent distribution, the the network is trained to minimize distance to it, and the process repeats.

distribution directly, so the technique imposes the constraints on a derived "latent" distribution, which the network than aims to approximate. This is done through alternating training steps, in which each distribution tries to minimize its distance to the other while the latter stays fixed. The output distribution can be trained using standard stochastic gradient descent, but the latent distribution, importantly, can be optimized through a simple convex optimization step.

As seen in figure 1, through this alternating training regime, the output distribution learns the linear constraints, without requiring us to impose linear constraints on the gradient descent it undergoes.

2 Constrained Optimization Formulation

A neural network is trained to approximate the probability distribution of segmentation class labels $Q(X)$ given any image I using a set of transformations given by the trained weights θ, which outputs the conditional distribution $Q(X|I, \theta)$. Finding weights θ that satisfy exactly all labels to all input images requires non-convex optimization, performed usually with gradient descent through backpropagating differentiable loss functions through the network's layers. Adding to that a set of linear constraints, $A\vec{Q} \geq \vec{b}$ is not a recipe for success.

However, we can decouple the constraints from the network output to alleviate the problem. By deriving a latent distribution $P(X)$ from $Q(X|\theta)$ which satisfies the constraints while minimizing its distance from $Q(X|\theta)$, we can encourage P and Q to capture the same distribution, and therefore enforce the constraints on Q by proxy. This is done through minimizing their distance in the probability distribution state-space with Kullback-Leibler divergence as a metric or distance function. The constrained optimization formulation is therefore as follows:

$$\min_{\theta,P} KL(P(X)||Q(X|\theta)) \tag{1}$$

$$\text{s.t: } A\vec{P} \geq \vec{b} \tag{2}$$

If the constraints (2) are satisfiable, then $P = Q$ is the global minimizer of the problem.

Alternating training makes optimization simple: when θ and therefore Q is fixed, (1) is convex in P; When P is fixed, (3) is standard cross-entropy loss for gradient descent.

2.1 Convexity

Optimizing (1) with respect to P can be done by examining the Lagrangian dual of the problem:

$$\begin{aligned}\mathcal{L}(P.\lambda) &= KL(P||Q) + \lambda^T(\vec{b} - A\vec{P}) \\ &= KL(P||Q) + \lambda^T\vec{b} - \sum_{i,l} \lambda^T A_{i,l}\vec{P}_l \\ &= KL(P||Q) + \lambda^T\vec{b} - \sum_i \log \sum_l \exp\bigl(\log Q_i(l|\theta) + A_{i,l}^T\lambda\bigr)\end{aligned}$$

Where the right-most term is simply the sum of the softmax of the predicted labels for each picture, taking into account the constraint matrix A. This is a convex objective, and Slater's condition holds for strong duality, as long as all constraints are satisfied.

Minimizing, the left-most term, corresponding to distance, is trivially cancelled, leaving us with a simple objective:

$$\mathcal{L}(\lambda) = \lambda^T\vec{b} - \sum_i \log \sum_l \exp\bigl(\log Q_i(l|\theta) + A_{i,l}^T\lambda\bigr). \tag{3}$$

2.2 Slack Variable Formulation

Notice that above, the condition was stated that Slater's condition holds only when all constraints are satisfied. Adding a slack variable, we are able to relax the problem enough to guarantee strong duality in any case.

Restating the problem with an added slack vector ξ, we get:

$$\min_{\theta,P,\xi\geq 0} KL(P(X)||Q(X|\theta)) + \beta^T\xi \tag{4}$$

$$\text{s.t: } A\vec{P} \geq \vec{b} - \xi \tag{5}$$

and the dual, with added Lagrangian multiplier $\gamma \geq 0$:

$$\begin{aligned} \mathcal{L}(P.\lambda, \gamma) &= KL(P||Q) + \beta^T \xi + \lambda^T (\vec{b} - A\vec{P} - \xi) - \gamma^T \xi \\ \frac{\partial \mathcal{L}}{\partial \xi}(P.\lambda, \gamma) &= \beta - \lambda - \gamma = 0 \\ \lambda &\leq \beta \end{aligned}$$

The weight β, being an upper bound on λ, allows for relaxation of the constraints. The fact that ξ has no effect on the gradient of the dual collapses it back to (3).

3 Demonstration

3.1 Simplification

The original paper is accompanied by a full implementation of a pipeline to train networks in the alternating training method. In this paper, I only undertake a single step of convex optimization on a pre-trained network, to test whether domain-knowledge constraints improve accuracy on the problem.

Additionally, instead of comparing a network trained on weak labels with one trained on the pixel-level ground-truth, I simplify by simply comparing an under-trained network with a fully-trained network, both trained on pixel-level labels.

3.2 Relevant Constraints

Pathak, Krähenbühl, and Darrell 2015 test their technique on the Pascal VOC data-set, which contains a variety of classes sparsely distributed between images, with a substantial background element. They use "suppression constraints" which set the probability of non-present labels in each image to zero, "foreground constraints" which provide a minimum for the total probability for pixels to contain each label in the prediction and "background constraint" which provides an upper and lower bound for the proportion of blank classified pixels.

I use the CamVid data-set, which is used to train autonomous vehicles. My labels contain very little blank pixels, and with only 11 classes, most images contain each and every one of them. These characteristics make the notions of background and suppression constraints irrelevant, so I implemented only foreground constraints, insisting that each image contains at least a proportion α of pixels of each class:0

$$\sum_i P_i(l) \geq \alpha_l \ \forall l \tag{6}$$

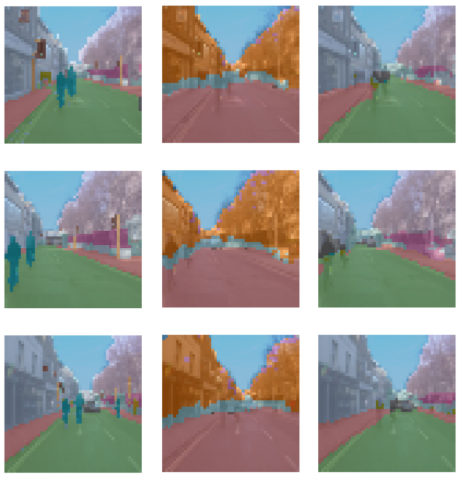

Figure 2: Segmentation of CamVid. Left: Ground-truth. Middle: Under-trained network. Right: Fully-trained network.

3.3 Procedure and Results

I used the *fastai* library to train a UNet on a reduced-size CamVid dataset for only one cycle through the training examples, to produce an under-trained model (accuracy: 66%). I then used the *CVXPY* library to optimize (1) with respect to P with constraints (5) on all labels for all images. Finally, I trained the model thoroughly to produce a fully-trained model to for comparison (accuracy: 87%), and examined the KL divergence between the fully-trained network and the under-trained network versus that between the fully-trained network and the optimized latent distribution.

I found that the optimized latent distribution was indeed closer to the fully-trained distribution than the under-trained distribution was by about 66%, indicating that the domain-knowledge constraints were leading the optimization in the correct direction. All code is available on GitHub in the following URL: `https://git.io/JfLFD`.

References

[PKD15] Deepak Pathak, Philipp Krähenbühl, and Trevor Darrell. "Constrained Convolutional Neural Networks for Weakly Supervised Segmentation". In: *CoRR* abs/1506.03648 (2015). arXiv: `1506.03648`. URL: `http://arxiv.org/abs/1506.03648`.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). arXiv: `1505.04597`. URL: `http://arxiv.org/abs/1505.04597`.