Contextually Constrained Deep Networks for Scene Labeling

Taygun Kekeç[1]
taygunkekec@gmail.com
Rémi Emonet[1]
remi.emonet@univ-st-etienne.fr
Elisa Fromont[1]
elisa.fromont@univ-st-etienne.fr
Alain Trémeau[1]
alain.tremeau@univ-st-etienne.fr
Christian Wolf[2]
christian.wolf@liris.cnrs.fr

[1] Université de Lyon, CNRS UMR 5516, Laboratoire Hubert-Curien
Université de Saint-Etienne, F-42000, Saint-Etienne, France
[2] Université de Lyon, CNRS INSA-Lyon, LIRIS, UMR5205, F-69622, France

Deep learning approaches, such as multi-layer neural networks, leverage the amount of available data to learn representations: instead of hand-crafting intermediate features, they are learned directly from the data. This is particularly relevant since there is no universal feature detector performing best for any given problem and these learned features have been shown to outperform hand-crafted features on many perception tasks.

In this work we focus scene labeling task with deep learning strategies. We first learn a CNN (Convolutional Neural Network) to predict contextual information. By forcing this network to capture some context information of our choice, we aim to improve the interpretability of the CNN and obtain meaningful feature maps. In parallel, we learn a second model for the original task assuming that contextual information is obtainable from ground truth labels at training step. Finally, we combine these networks and perform a last training phase with weakened supervision.

In traditional feature learning, the input processing is separated in two parts as illustrated in Figure 1a. The input I is first processed with a function $f(.)$, which has parameters θ_f and produces a set of features F. A predictor $p(.)$ having parameters θ_p takes the features F as input and produces a prediction. To constrain the whole network, we propose to split the function f into two parts: f_d and f_c (Fig. 1b). Function f_c aims at predicting some context and it is learned with additional supervision.

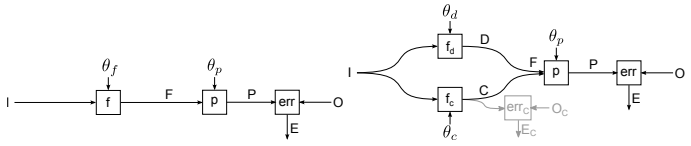

Figure 1: Functional representation of our feature learning approaches. (a) The target function is composed of a feature extraction function f and a prediction function p. (b) Our approach which distinguishes the learning of context features f_c and dependent features f_d.

Learning context – In this step, we start from a random initialization θ_c^0 and learn θ_c^1 where the superscript j in θ_c^j indicates the training stage. The context learning step minimizes the following error function: $\mathcal{L}_c = \sum_{k=1}^{K} \left\| p_{soft}^k(f_c(I,\theta_c) - O_c^k) \right\|^2$ where K is the number of context pixels for a patch I_i, p_{soft}^k is the softmax prediction output for $k'th$ pixel and O^k is the ground-truth label of $k'th$ context pixel.

The context learner is trained with a semantic label map containing the ground truth labels of the pixels to predict. At the end of this training step, the feature maps that correspond to the output of the *Context Learner* will be specialized in modeling the neighboring context of the target pixel.

As a standard CNN focuses only on learning the class of a given patch y_i, it is hard to infer what the last layers are actually learning. In contrast, our learner increases the interpretability of the whole network. In Fig. 2, we show the responses of our context learner maps for some input patches where feature maps learn to capture patch context.

Learning dependent features – The goal of this part of the augmented learner is to learn the parameters (θ_d^2, θ_p^2) from a random initialization of (θ_d^0, θ_p^0) and from parameters θ_c^1 learned in the previous step. We minimize $\mathcal{L}$ while keeping θ_c^1 fixed. Fixing θ_c prevents harming the parameters of the context learner while learning θ_d^2. We stochasticly replace context predictions with some true labels to regularize learning of $f_d(.)$.

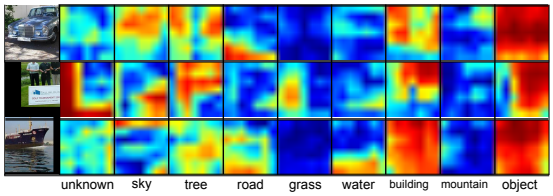

Figure 2: Feature maps of context learner for some input patches.

Fine tuning – In this step, we learn final parameters $\theta^3 = (\theta_c^3, \theta_d^3, \theta_p^3)$. We start from an initial value of $(\theta_c^1, \theta_d^2, \theta_p^2)$, and we minimize $\mathcal{L}$. This idea of this overall refinement step is to weaken the level of supervision and allow both θ_f and θ_d to adjust to this sudden lack of possible ground truth contextual information which is obviously not present during the test step.

Experiments Our approach has been tested on two scene labeling datasets: Stanford Background and SIFT Flow. The Stanford Background dataset contains 715 images of outdoor scenes having 9 classes. Our context learner transforms a 46×46 patch into a 7×7 context output. In the first layer, it has sixteen 7×7 filters and then 2×2 pooling operations for each feature map. Its second layer is composed of K filters (each of size 7×7) each encoding the context of a specific class followed by a 2×2 pooling operation. This layer has thus K output maps, where K corresponds to the number of classes.

(a) msConvnet (b) msAugLearner (c) GroundTruth

Figure 3: Raw image labeling of the multiscale ConvNet, our multiscale augmented learner and ground truth labels.

Both single scale and multiscale variants of the architecture has been analysed. While the accuracy gain varies between singlescale and multiscale implementations, we observe that our approach consistently improves both pixel and class accuracies. The gain on single-scale experiments are higher compared to multiscale implementations. This brings us to the empirical conclusion that contextual cues obtained implicitly through appearance cues of large support size provides valuable contextual information.

From a computational perspective, our approach increases the number of parameters by less than 1% compared to the ConvNet. Overall, we observe that our method provides better results for both the Stanford and the SIFT Flow datasets. Some labeling results from the Stanford dataset are shown in Figure 3. Our approach yields results that are more visually coherent than those obtained with the plain ConvNet architecture.