2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

A Simple Weight Recall for Semantic Segmentation: Application to Urban Scenes

Xuhong LI, Franck DAVOINE and Yves GRANDVALET

Abstract— **In many learning tasks, including semantic image segmentation, performance can be effectively improved through the fine-tuning of a pre-trained convolutional network, instead of training from scratch. With fine-tuning, the underlying assumption is that the pre-trained model extracts generic features, which are at least partially relevant for solving a segmentation task, but that would be difficult to extract from the smaller amount of data that is available for training in urban driving scenes segmentation. However, besides the initialization with the pre-trained model and the early stopping, there is no mechanism in classical fine-tuning approaches for keeping the generic features. Even worse, the standard weight decay drives the parameters towards the origin and affects the learned features. In this paper, we show that a simple regularization that uses the pre-trained model as a reference consistently improves the performance when applied to semantic urban driving scene segmentation. Experiments are done on the Cityscapes dataset, with four different architectures of convolutional networks.**

I. Introduction

Recent years have seen an increasing interest in driving scene understanding and semantic image segmentation is one of the important tools that can help to distinguish the different objets in the surrounding neighborhood of a vehicle as well as their complex relationships.

Image segmentation, i.e. the partitioning of an image into sets of pixels that correspond to parts that are coherent in terms of low-level cues such as colors, textures and smoothness of boundaries, has been widely explored in the 90s. Nowadays, semantic image segmentation, i.e. the labeling of each pixel of an image with the category of the object it belongs to, is still a fundamental topic, at the interface of deep learning and computer vision. This research is of broad interest for various applications such as autonomous driving, human-machine interaction or medical image computing to name a few. Contemporary deep networks used for semantic image segmentation (FCN [1], DeepLab [2], PSPNet [3]) are variants of networks initially proposed for image classification (AlexNet, VGG net [4], GoogLeNet [5], ResNet [6]). Most of them are pre-trained for a source task on a large-scale database like ImageNet [7] and then fine-tuned on a smaller database for a more specific target task like object detection, or specialization like driving scene segmentation.

Some form of knowledge is believed to be extracted by learning from large-scale databases of the source task and this knowledge is then transferred to the target task by initializing the network with the pre-trained parameters. However, after fine-tuning, some of the parameters may be quite different from their initial values, resulting in possible losses of general knowledge that may be relevant for the targeted problem. In particular, during fine-tuning, L^2 regularization drives the parameters towards the origin and thereby encourages large deviations of the parameters from their initial values.

In order to help preserve the acquired knowledge embedded in the initial network, we consider using another parameter regularization method during fine-tuning. We argue that the standard L^2 regularization, which drives the parameters towards the origin, is not adequate in the framework of transfer learning where the initial values provide a more sensible reference point than the origin. This simple modification keeps the original control of overfitting, by constraining the effective search space around the initial solution, while encouraging committing to the acquired knowledge. In this paper, we investigate a variant of L^2 penalties using the pre-trained model as reference, which we name L^2*-SP* because the pre-trained parameters represent the starting point (*-SP*) of the fine-tuning process. We show that this simple and easy to implement method produces consistent improvement with noticeable effects in segmentation tasks. We present comprehensive experiments using four different deep networks, i.e. FCN, ResNet, and two ResNet-based networks, DeepLab and PSPNet, pre-trained on ImageNet [7] and on COCO [8] for two of them, and fine-tuned on Cityscapes [9].

II. Related Work

In this section, we recall the existing works dedicated to the improvements of convolutional networks for image semantic segmentation and the existence of similar regularization techniques that were previously applied in different domains.

A. Convolutional networks for segmentation

Besides the success in image classification, convolutional networks have also achieved impressive progress in object detection, e.g. [10], and segmentation, e.g. [1]–[3], relying on fine-tuning to improve the performance over training from scratch.

Fully convolutional network (FCN) [1], based on VGG-16 [4], is the first end-to-end convolutional network for segmentation. FCN convolutionalizes the fully-connected layers, exploits the spatial information from convolution operations and compensates the loss of information in pooling layers by adding skips from intermediate layers and bilinear interpolations. Residual Network (ResNet) [6] is one of the most popular architectures for image classification, and can

The authors are with Sorbonne universités, Université de technologie de Compiègne, CNRS, Heudiasyc Lab., UMR 7253, Compiègne, France.

978-1-5386-4451-5/18/$31.00 ©2018 IEEE

be lightly modified for segmentation tasks. DeepLab [2], constructed on ResNet, integrates dilated convolutional layers (also called as atrous) into the standard ResNet structure, with a fully connected Conditional Random Fields (CRFs) as a post-processing method to improve the segmentation performance. Besides that, DeepLab collects multi-scale features extracted by atrous layers with different scales, to classify pixels in consideration of pixels in a large neighborhood. Similarly, PSPNet [3] gathers pooled feature maps in different scales, and concatenates them with local feature maps to combine local, global and intermediate-scale information.

All these convolutional networks for segmentation are fine-tuned from the pre-trained weights instead of training from scratch. Fine-tuning is the most popular method for transfer learning tasks, including segmentation tasks, when using deep convolutional networks. The success of transfer learning with convolutional networks relies on the generality of the learned representations that have been constructed from a large open database like ImageNet. Yosinski et al. [11] quantified the transferability of these pieces of information in different layers, e.g. the first layers learn general features, the middle layers learn high-level semantic features and the last layers learn the features that are very specific to a particular task. That can be also noticed by the visualization of features [12]. Overall, the learned representations can be conveyed to related but different domains and the parameters in the network are reusable for different tasks.

B. Parameter regularization

Parameter regularization can take different forms in deep learning. L^2 regularization has been used for a long time as a very simple method for preventing overfitting by penalizing the L^2 norm of the parameter vector. It is the usual regularization used in deep learning, including for fine-tuning.

In lifelong learning, where a series of tasks is learned sequentially with a single model with the objective to achieve a good performance on all tasks, and domain adaptation, where the target task is identical to the source task and no (or a small quantity of) target data is labeled, several works attempt to improve the performance by using parameter regularization approaches to preserve pre-trained parameters from source tasks. Li et al. [13] proposed to use the outputs of the target examples, computed by the original network on the source task, to define a learning scheme preserving the memory of the source tasks when training on the target task. They also tried to preserve the pre-trained parameters instead of the outputs of examples but without significant effectiveness. Kirkpatrick et al. [14] developed a similar approach with success, getting sensible improvements by measuring the sensitivity of the parameters of the network learned on the source data thanks to the Fisher information. However, their regularization scheme is designed for lifelong learning tasks and may be thought as being inadequate for inductive transfer learning, where performance is only measured on the target task. Rozantsev et al. [15] introduced a parameter regularization for keeping the similarity between the pre-trained and the fine-tuned models.

Regularization has been a means to build shrinkage estimators for decades. Shrinking towards zero is the most common form of shrinkage, but shrinking towards adaptively chosen targets has been around for some time, starting with Stein shrinkage (see e.g. [16, chapter 5]), where it can be related to empirical Bayes arguments. In transfer learning, it has been used in maximum entropy models [17] or SVM [18]–[20]. These approaches were shown to outperform standard L^2 regularization with limited labeled data in the target task [19], [20]. These relatives differ from the application to deep networks in several respects, the more important one being that they consider a fixed representation, where transfer learning aims at producing similar classification parameters in that space, that is similar classification rules. For deep networks, transfer aims at learning similar representations upon which classification parameters will be learned from scratch. Hence, even though the techniques we discuss here are very similar regarding the analytical form of the regularizers, they operate on very different objects.

Thus, to the best of our knowledge, we present the first results on segmentation with convolutional networks that are based on the L^2-*SP* regularization term described in the following section.

III. Regularizers for fine-tuning

In this section, we detail the penalties we consider for fine-tuning. When learning from scratch, regularization is aimed at facilitating optimization and avoiding overfitting, by implicitly restricting the capacity of the network, that is, the effective size of the search space. In transfer learning, the role of regularization is similar, but the starting point of the fine-tuning process conveys knowledge that pertains to the source problem (domain and task). Hence, the network capacity has not to be restricted blindly: the pre-trained model sets a reference that can be used to define the functional space effectively explored during fine-tuning.

Since we are using early stopping, fine-tuning a pre-trained model is an implicit form of inductive bias towards the initial solution. We explore here how a coherent explicit induction bias, encoded by a regularization term, affects the training process. Section IV shows that this such scheme gets an edge over the standard approaches.

Let $\boldsymbol{w} \in \mathbb{R}^n$ be the parameter vector containing all the network parameters that are to be adapted to the target task. The regularized objective function $\tilde{J}$ that is to be optimized is the sum of the standard objective function J and the regularizer $\Omega(\boldsymbol{w})$. In our experiments, J is the negative log-likelihood, so that the criterion $\tilde{J}$ could be interpreted in terms of maximum *a posteriori* estimation, where the regularizer $\Omega(\boldsymbol{w})$ would act as the log prior of $\boldsymbol{w}$. More generally, the minimizer of $\tilde{J}$ is a trade-off between the data-fitting term and the regularization term.

L^2 penalty: Our baseline penalty for transfer learning is the usual L^2 penalty, also known as weight decay, since

it drives the weights of the network to zero:

$$\Omega(\boldsymbol{w}) = \frac{\alpha}{2} \left\|\boldsymbol{w}\right\|_2^2 \ , \tag{1}$$

where α is the regularization parameter setting the strength of the penalty and $\left\|\cdot\right\|_p$ is the p-norm of a vector.

L^2-SP penalty: Let $\boldsymbol{w}^0$ be the parameter vector of the model pre-trained on the source problem, acting as the starting point (*-SP*) in fine-tuning. Using this initial vector as the reference in the L^2 penalty, we get:

$$\Omega(\boldsymbol{w}) = \frac{\alpha}{2} \left\|\boldsymbol{w} - \boldsymbol{w}^0\right\|_2^2 \ . \tag{2}$$

Typically, the transfer to a target task requires slight modifications of the network architecture used for the source task, such as on the last layer used for predicting the outputs. Then, there is no one-to-one mapping between $\boldsymbol{w}$ and $\boldsymbol{w}^0$, and we use two penalties: one for the part of the target network that shares the architecture of the source network, denoted $\boldsymbol{w}_{\mathcal{S}}$, the other one for the novel part, denoted $\boldsymbol{w}_{\bar{\mathcal{S}}}$. The compound penalty then becomes:

$$\Omega(\boldsymbol{w}) = \frac{\alpha}{2} \left\|\boldsymbol{w}_{\mathcal{S}} - \boldsymbol{w}_{\mathcal{S}}^0\right\|_2^2 + \frac{\beta}{2} \left\|\boldsymbol{w}_{\bar{\mathcal{S}}}\right\|_2^2 \ . \tag{3}$$

We have tested several different forms of regularization approaches on classification tasks [21] and conclude that for target tasks, L^2-*SP* is the most efficient among those regularization approaches that preserve the learned knowledge from source domains (at par with a more complex penalty based on Fisher information). In this article, we prove that on segmentation tasks L^2-*SP* also behaves better than the standard L^2 that is currently used.

IV. Experiments

We evaluate the L^2 and L^2-*SP* penalties on Cityscapes [9] with several different networks. L^2-*SP* can be applied to all layers except new layers, and parameters in new layers are regularized by L^2 penalty as described in Section III.

A. Databases and networks

Source Databases: ImageNet [7] and Microsoft COCO [8] are used as sources. Both of them are large-scale databases: ImageNet for image classification; Microsoft COCO for object detection and semantic image segmentation. Pre-training on ImageNet can largely increase the performance for most transfer learning tasks, moreover pre-training on ImageNet and then on COCO can further raise percentages in segmentation performance. In our experiments, we compare L^2-*SP* with the standard L^2 regularization approach using these two pre-training schemes.

Target Database: Cityscapes [9] is a database of real-world urban driving scenes for segmentation. The database splits 5000 finely labeled images into a training set (2975 images), validation set (500 images) and test set (1525 images). The ground truth of test set is not available on public so in our experiments we train our networks on the training set and evaluate the performance on the validation set. Meanwhile, we have also submitted some of our results on the Cityscapes benchmark to evaluate our method on the test set. Cityscapes has 20000 additional images with coarse annotations but in this paper, we did not use them.

TABLE I: Mean IoU scores on Cityscapes. The second column recalls the results obtained in previous works and the two last columns show our results with L^2 and L^2-*SP* fine-tuning. Note that the initial models for DeepLab-COCO and PSPNet-COCO are pre-trained on ImageNet and then on Microsoft COCO, the other four are only pre-trained on ImageNet. The reported mIoU scores are on the Cityscapes validation set, except for the two scores marked (test set) that have been computed on the Cityscapes test set.

	L^2 in [2], [3], [9]	L^2	L^2-*SP*
FCN	65.3 (test set)	66.9	**67.9**
ResNet-101	66.6	68.1	**68.7**
DeepLab	—	68.6	**70.4**
DeepLab-COCO	70.4	72.0	**73.2**
PSPNet	76.0 (test set)	75.1	**76.1**
PSPNet-COCO	—	78.0	**79.0**

Convolutional Networks for Semantic Image Segmentation: FCN [1] is one of the most classical structures for segmentation. Deeplab [2] and PSPNet [3] stayed for some time on the top of the Cityscapes benchmark and are two favored structures. Unfortunately, we do not have enough GPU resources to perfectly reproduce the performance of PSPNet, as explained in the training details. Nonetheless, our experimental work here focuses on four quite diverse structures of networks, FCN, the standard ResNet-101, DeepLab, and PSPNet for demonstrating the consistency of L^2-*SP*.

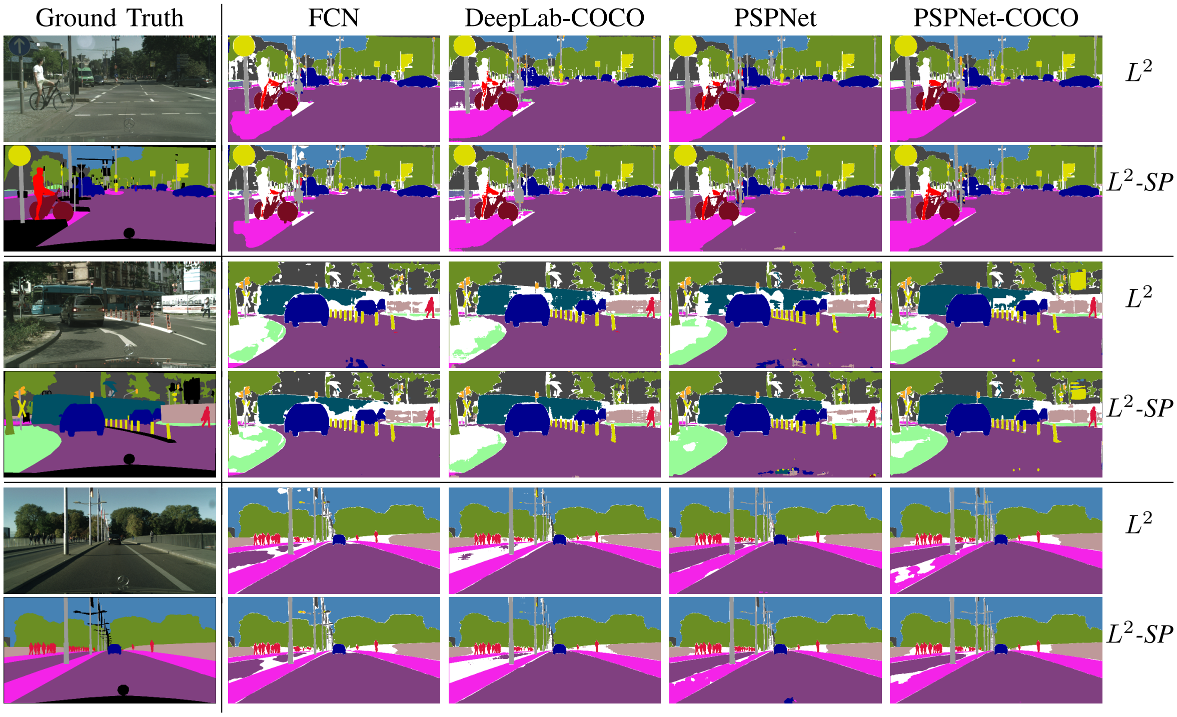
B. Training details

All images in Cityscapes are color images. All training examples are pre-processed as follows: we subtract the mean activity computed over the training set from each channel, then we adopt random blur, random mirror and random crop for data augmentation. The network parameters are regularized using the two techniques as described in Section III. We have also separated all weights to $\boldsymbol{w}_{\mathcal{S}}$ and $\boldsymbol{w}_{\bar{\mathcal{S}}}$ for L^2, using α and β respectively as regularization hyperparameters. According to experiments in [21], we tried $\{10^{-2}, 10^{-3}, 10^{-4}\}$ for α and $\{10^{-3}, 10^{-4}\}$ for β. Regarding testing, we compute the mean intersection-over-union metric (mIoU) score [9] over 19 classes in Cityscapes and we do not make use of further post-processing methods to improve the performance. Stochastic gradient descent with momentum 0.9 is used for optimization. We set the learning rate as large as possible, provided the loss can decrease in the first iterations. For FCN, we do not decrease the learning rate as suggested in [1]. For ResNet based networks, we use the polynomial learning rate policy as in [2], [3]. The batch size is 2 and image crops have the size of 800×800 except for PSPNet. In order to stabilize the statistics of batch normalization layers, for PSPNet, we use 16 examples in a mini-batch with image crops of 624×624 pixels. All experiments are performed with Tensorflow [22].

C. Results

We reproduce the experiments of [2], [3], [9] that use the standard L^2 fine-tuning, and compare with L^2-*SP* fine-

Fig. 1: Comparisons of L^2-*SP* and L^2 on three images from the Cityscapes validation set. The first column shows the original image above its ground truth segmentation (overlayed colors encode semantic classes [9], the black color is not included in any evaluation and is treated as *void*). Columns 2 – 5 present four different networks: FCN, DeepLab-COCO, PSPNet, PSPNet-COCO. On the first line, segmentation results with L^2, and on the second line with L^2-*SP*. Pixels that are correctly classified keep their ground truth color. Pixels that are incorrectly classified are set to white (a 100% accuracy segmentation result should not contain any white pixel).



tuning, all other setup parameters being unchanged. The results, computed on the validation set of Cityscapes, are reported in Table I. PSPNet is the only network we could not perfectly reproduce because we had to use smaller input images compared to [3], due to our limited computational ressources. We readily observe that fine-tuning with L^2-*SP* in place of L^2 consistently improves the performance in mean IoU score, for all networks. Several examples are shown in Figure 1. The three images displayed in Figure 1 belong to the validation set. They were chosen so as to display the dissimilarity between the two regularization approaches for the best performing model. More precisely, they have large difference in mean IoU score between PSPNet-COCO-L^2 and PSPNet-COCO-L^2-*SP*. We also include the segmentations obtained for three other models for comparison purposes.

Table II show the per-class results and we can see that fine-tuning with L^2-*SP* maintains the scores in easy classes like road, building, vegetation, sky etc, but also improves the performance in those difficult but important classes, like people and vehicles. Some image details centered on certain objects are shown in Figure 2. Images in Figure 2 are chosen in the same way as Figure 1.

Table I and II report the results evaluated on the validation set. Table III shows the results on the test set. The numerical results in Table III have been submitted to the Cityscapes evaluation server, and will be made public on the official website.

D. Analysis and discussion

1) Theoretical insights: Analytical analyses are very difficult in the deep learning framework. Under some (highly) simplifying assumptions, the effect of L^2 regularization can be analyzed by doing a quadratic approximation of the objective function around the optimum [23, Section 7.1.1]. This analysis shows that L^2 regularization rescales the parameters along the directions defined by the eigenvectors of the Hessian matrix. This scaling is equal to $\lambda_i/(\lambda_i+\alpha)$ for the i-th eigenvector of eigenvalue λ_i. A similar analysis can be used for the L^2-*SP* regularization.

We recall that $J(\boldsymbol{w})$ is the unregularized objective function, and $\tilde{J}(\boldsymbol{w}) = J(\boldsymbol{w}) + \alpha \left\|\boldsymbol{w}-\boldsymbol{w}^0\right\|_2^2$ is the regularized objective function. Let $\boldsymbol{w}^* = \mathrm{argmin}_{\boldsymbol{w}} J(\boldsymbol{w})$ and $\tilde{\boldsymbol{w}} = \mathrm{argmin}_{\boldsymbol{w}} \tilde{J}$ be their respective minima. The quadratic approximation of $J(\boldsymbol{w}^*)$ gives

$$\mathbf{H}(\tilde{\boldsymbol{w}}-\boldsymbol{w}^*)+\alpha(\tilde{\boldsymbol{w}}-\boldsymbol{w}^0)=0\ , \tag{4}$$

where $\mathbf{H}$ is the Hessian matrix of J w.r.t. $\boldsymbol{w}$, evaluated at $\boldsymbol{w}^*$. Since $\mathbf{H}$ is positive semidefinite, it can be decomposed as $\mathbf{H} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T$. Applying the decomposition to Equation (4), we obtain the following relationship between $\tilde{\boldsymbol{w}}$ and $\boldsymbol{w}^*$:

$$\mathbf{Q}^T\tilde{\boldsymbol{w}} = (\boldsymbol{\Lambda}+\alpha\mathbf{I})^{-1}\boldsymbol{\Lambda}\mathbf{Q}^T\boldsymbol{w}^* + \alpha(\boldsymbol{\Lambda}+\alpha\mathbf{I})^{-1}\mathbf{Q}^T\boldsymbol{w}^0\ . \tag{5}$$

We can see that with L^2-*SP* regularization, in the direction defined by the i-th eigenvector of $\mathbf{H}$, $\tilde{\boldsymbol{w}}$ is a convex combination of $\boldsymbol{w}^*$ and $\boldsymbol{w}^0$ in that direction since $\lambda_i/(\lambda_i+\alpha)$ and

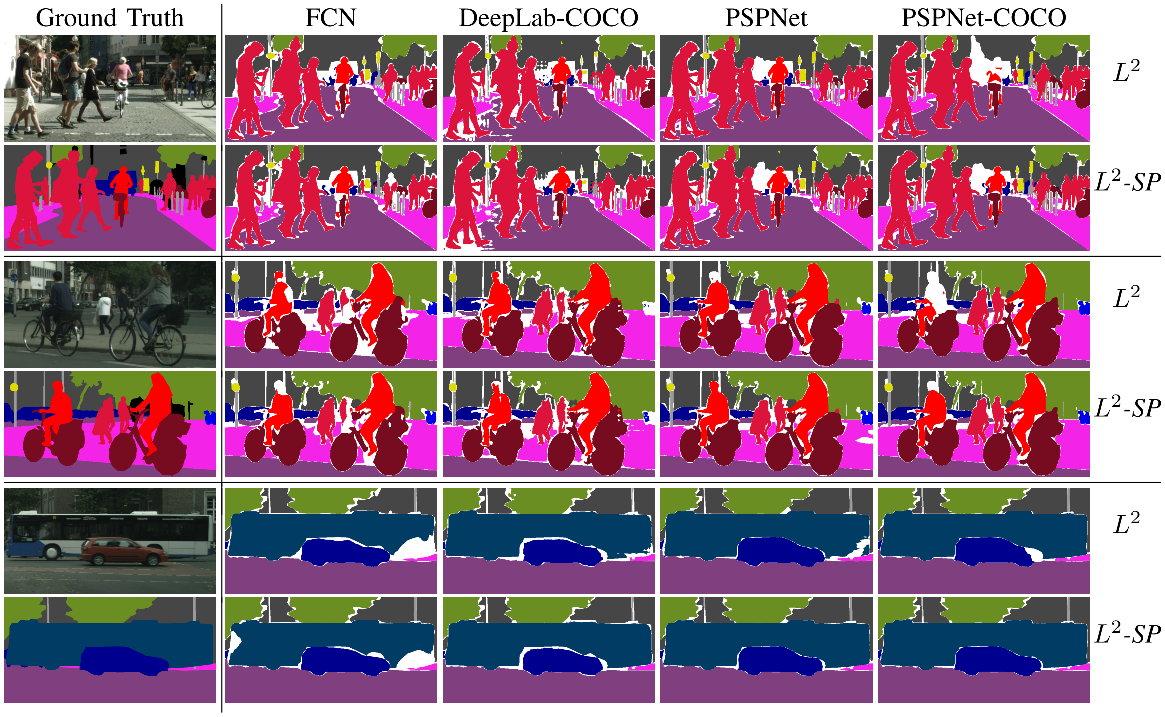
TABLE II: IoU scores for each object class of the Cityscapes Validation set. *-COCO means that the model is first pre-trained on ImageNet and then on Microsoft COCO.

	road	sidew.	build.	wall	fence	pole	t.light	t.sign	veg.	ter.	sky	pers.	rider	car	truck	bus	train	m.bike	bike	mIoU
FCN.L^2	97.1	**79.1**	89.2	36.6	48.7	51.9	57.5	69.5	90.7	**58.1**	**92.4**	75.1	**51.5**	91.8	46.8	69.5	48.3	47.5	70.5	66.9
FCN.L^2-*SP*	97.1	79.0	**89.4**	**42.8**	**49.4**	**55.0**	**59.9**	**71.1**	**90.8**	56.8	90.6	**75.3**	51.0	**92.2**	**49.1**	**70.2**	**50.6**	**48.6**	**71.3**	**67.9**
ResNet-101.L^2	97.4	79.9	90.3	44.8	48.7	52.8	58.9	68.6	91.3	58.5	92.7	76.4	52.0	92.3	49.0	66.1	**48.9**	52.4	72.6	68.1
ResNet-101.L^2-*SP*	**97.5**	**80.7**	**90.6**	**45.4**	**50.2**	**54.0**	**61.9**	**70.8**	**91.4**	**58.8**	**93.2**	**77.0**	**52.8**	**92.7**	**49.9**	**66.4**	45.8	**53.9**	**73.3**	**68.7**
DeepLab.L^2	97.3	79.7	90.3	**49.0**	50.4	51.8	55.1	66.2	90.8	57.9	93.1	75.3	51.2	92.1	52.7	70.8	56.4	53.4	70.2	68.6
DeepLab.L^2-*SP*	**97.4**	**80.2**	**90.7**	48.7	**52.5**	**53.4**	**58.4**	**68.2**	**91.0**	**59.0**	**93.5**	**76.3**	**54.0**	**92.5**	**58.4**	**74.5**	**62.7**	53.4	**71.5**	**70.3**
DeepLab-COCO.L^2	97.6	**81.5**	90.9	46.8	50.4	**54.4**	**61.2**	**71.6**	91.2	59.7	**93.4**	78.0	56.5	93.3	67.8	81.3	62.8	56.5	73.0	72.0
DeepLab-COCO.L^2-*SP*	97.6	81.2	90.9	**48.0**	**51.4**	54.2	60.8	70.9	91.2	**60.3**	93.3	**78.3**	**57.5**	**93.6**	**71.9**	**84.1**	**72.4**	**59.3**	**73.5**	**73.2**
PSPNet.L^2	**98.0**	**84.0**	**92.0**	46.3	**57.8**	**64.0**	**71.0**	**78.1**	92.3	64.2	**94.6**	**81.7**	61.7	94.8	68.7	80.3	54.2	65.0	**77.6**	75.1
PSPNet.L^2-*SP*	97.9	83.4	91.8	**50.5**	55.2	61.9	68.3	76.8	92.3	**65.0**	94.3	80.7	61.7	94.8	**74.6**	**84.2**	**70.1**	**65.6**	76.7	**76.1**
PSPNet-COCO.L^2	98.2	85.3	92.4	46.2	58.7	**65.8**	**72.7**	**80.5**	92.4	64.8	94.7	83.7	64.0	95.5	81.3	87.2	71.7	**67.8**	**78.8**	78.0
PSPNet-COCO.L^2-*SP*	98.2	**85.5**	**92.5**	**52.0**	**61.5**	65.2	72.1	80.0	**92.6**	**65.6**	94.7	**83.9**	**65.8**	95.5	**82.6**	**89.0**	**78.9**	67.2	78.7	**79.0**

TABLE III: IoU scores for each object class of the Cityscapes Test set. *-COCO means that the model is first pre-trained on ImageNet and then on Microsoft COCO.

	road	sidew.	build.	wall	fence	pole	t.light	t.sign	veg.	ter.	sky	pers.	rider	car	truck	bus	train	m.bike	bike	mIoU
DeepLab-COCO.L^2	97.9	81.1	90.6	42.0	45.8	53.1	**61.3**	**68.0**	91.8	68.6	94.1	80.2	58.3	93.8	54.4	64.3	59.6	58.9	69.2	70.2
DeepLab-COCO.L^2-*SP*	97.9	**81.2**	90.6	**42.8**	**47.2**	53.1	60.8	67.9	91.8	**68.7**	**94.2**	**80.5**	**59.9**	**94.0**	**56.4**	**66.1**	**61.9**	**60.2**	**69.6**	**70.8**
PSPNet.L^2	**98.4**	**84.6**	**92.1**	45.6	53.1	**63.2**	**71.2**	**75.3**	**93.0**	71.5	**95.1**	**84.1**	**65.9**	95.0	61.0	**74.3**	62.8	**64.0**	**73.9**	74.9
PSPNet.L^2-*SP*	98.3	83.8	92.0	**50.7**	**53.5**	61.0	68.5	73.5	92.9	71.5	94.9	83.3	65.8	95.0	**66.2**	73.9	**66.3**	63.0	72.8	**75.1**
PSPNet-COCO.L^2	98.5	**85.5**	92.7	**53.5**	57.5	**65.7**	**74.2**	**77.9**	93.4	**73.1**	**95.4**	**86.0**	**69.4**	**95.7**	59.2	73.5	63.6	**69.7**	**76.2**	76.9
PSPNet-COCO.L^2-*SP*	98.5	85.0	92.7	53.3	**58.0**	64.4	73.0	77.3	93.4	72.2	95.3	85.7	69.2	95.5	**63.7**	**73.6**	**69.9**	68.9	75.8	**77.1**

Fig. 2: Comparisons of L^2-*SP* and L^2: zoom on some object classes like rider, person, and different kinds of vehicles. Images are from the validation set and pixels are colored in the same way as in Figure 1.



$\alpha/(\lambda_i+\alpha)$ sum to 1. $\tilde{\boldsymbol{w}}$ of the regularized objective function with L^2-*SP* is a compromise between $\boldsymbol{w}^*$ and $\boldsymbol{w}^0$, precisely an affine combination along the directions of eigenvectors of the Hessian matrix of the unregularized objective function.

This contrasts with L^2 that leads to a compromise between $\boldsymbol{w}^*$ and the origin. Clearly, searching a solution around the pre-trained parameter vector is intuitively much more appealing, since it is the actual motivation for using the pre-trained parameters as the starting point of the fine-tuning process. Hence, the regularization procedures resulting in the compromise with the pre-trained parameter encode a penalty that is coherent with the original motivation.

2) Coherent motivation from shrinkage estimation: Using L^2-*SP* instead of L^2 can also be motivated by an analogy with shrinkage estimation see e.g. [16, chapter 5]. Although it is known that shrinking toward any reference is better than raw fitting, it is also known that shrinking towards a value that is close to the "true parameters" is more effective. The notion of "true parameter" is not applicable to deep networks, but the connection with Stein shrinking effect may be inspiring by surveying the literature considering shrinkage towards other references, such as linear subspaces. In particular, it is likely that manifolds of parameters defined from the pre-trained network would provide a better reference than the single parameter value provided by the pre-trained network.

3) Computational efficiency: L^2-*SP* penalty introduces no extra parameters, and only increases slightly the computational burden, by less than 1% of the number of floating point operations of ResNet-101. At little computational cost, we can thus obtain 1$\sim$2% improvements in mIoU score, and no additional cost is experienced at test time.

4) Proportion of new parameters: Most layers of the network used for segmentation were pre-trained on a classification task, but for solving the segmentation tasks, some changes in the network are necessary. All the networks evaluated in this paper require new layers and thereby new parameters for solving the Cityscapes segmentation task. However, the ratio of new parameters is quite different among the four networks: FCN has less than 0.1% new parameters, ResNet-101 has 0.1%, DeepLab 3.4% and PSPNet 37.5%. Since new parameters are penalized with L^2, the proportion of new parameters reduces the relative effect of L^2-*SP*. This is visible with PSPNet: the scores of several classes are degraded despite one percent improvement in average. As for PSPNet-COCO, only five classes are moderately degraded. We argue that most new parameters in the PSPNet model were trained on Microsoft COCO and can be used as reference when training on Cityscapes, so the effect of L^2-*SP* remains. Note that DeepLab models are different from PSPNet in architecture, and training on COCO does not reduce the proportion of new parameters. So when using the two DeepLab models, the trends of L^2-*SP*, compared with L^2, are both positive.

V. Conclusion

In this paper, we described and tested a variant of the L^2 penalty, L^2-*SP*, that uses pre-trained model parameters as a reference to encode an explicit bias towards the solution learned on a source task. The L^2-*SP* penalty has been already used for other purposes but we demonstrate here its relevance in segmentation tasks with different deep convolutional networks, through an evaluation on the Cityscapes database. This penalty is very simple to implement and quite more effective than the standard L^2 penalty that is currently used in fine-tuning. We also provide theoretical hints motivating L^2-*SP* by the effective hypothesis space explored during optimization and shrinkage estimation. We recommend this simple L^2-*SP* scheme as the legitimate baseline fine-tuning strategy for segmentation tasks.

Acknowledgements: This work was carried out within SIVALab, a joint laboratory between Renault and Heudiadyc CNRS/UTC Lab., with the support of the China Scholarship Council. We acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research.

References

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. of CVPR*, Boston, USA, 2015.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Tr. on PAMI*, 2017.

[3] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. of CVPR*, Hawaii, USA, 2017.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. of CVPR*, Boston, USA, June 2015.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of CVPR*, Las Vegas, USA, June 2016.

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. of ECCV*, Zurich, September 2014, pp. 740–755.

[9] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of CVPR*, Las Vegas, USA, June 2016.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2980–2988.

[11] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. of NIPS*, Montreal, Canada, 2014.

[12] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. of ECCV*, Zurich, September 2014.

[13] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[14] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, 2017.

[15] A. Rozantsev, M. Salzmann, and P. Fua, "Beyond sharing weights for deep domain adaptation," *arXiv:1603.06432*, 2016.

[16] E. L. Lehmann and G. Casella, *Theory of point estimation*, 2nd ed. Springer, 1998.

[17] C. Chelba and A. Acero, "Adaptation of maximum entropy capitalizer: Little data can help a lot," *Computer Speech & Language*, vol. 20, no. 4, pp. 382–399, 2006.

[18] J. Yang, R. Yan, and A. G. Hauptmann, "Adapting svm classifiers to data with shifted distributions," in *ICDM Workshops*, Omaha, Nebraska, USA, 2007.

[19] Y. Aytar and A. Zisserman, "Tabula rasa: Model transfer for object category detection," in *Proc. of ICCV*, Barcelona, November 2011.

[20] T. Tommasi, F. Orabona, and B. Caputo, "Learning categories from few examples with multi model knowledge transfer," *IEEE Tr. on PAMI*, vol. 36, no. 5, May 2014.

[21] X. Li, Y. Grandvalet, and F. Davoine, "Explicit inductive bias for transfer learning with convolutional networks," in *arXiv:1802.01483*, February 2018.

[22] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, available from tensorflow.org.

[23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, ser. Adaptive Computation and Machine Learning. MIT Press, 2017. [Online]. Available: http://www.deeplearningbook.org