Pattern Recognition 98 (2020) 107049

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

A baseline regularization scheme for transfer learning with convolutional neural networks

Xuhong Li[*], Yves Grandvalet, Franck Davoine

Alliance Sorbonne Université, Université de technologie de Compiègne, CNRS, Heudiasyc, Compiègne, UMR 7253, France

ARTICLE INFO

Article history:
Received 3 August 2018
Revised 10 July 2019
Accepted 10 September 2019
Available online 11 September 2019

Keywords:
Transfer learning
Regularization
Convolutional networks

ABSTRACT

In inductive transfer learning, fine-tuning pre-trained convolutional networks substantially outperforms training from scratch. When using fine-tuning, the underlying assumption is that the pre-trained model extracts generic features, which are at least partially relevant for solving the target task, but would be difficult to extract from the limited amount of data available on the target task. However, besides the initialization with the pre-trained model and the early stopping, there is no mechanism in fine-tuning for retaining the features learned on the source task. In this paper, we investigate several regularization schemes that explicitly promote the similarity of the final solution with the initial model. We show the benefit of having an explicit inductive bias towards the initial model. We eventually recommend that the baseline protocol for transfer learning should rely on a simple L^2 penalty using the pre-trained model as a reference.

1. Introduction

It is now well known that modern convolutional neural networks [e.g. 20,24,43,44] can achieve remarkable performance on large-scale image databases, e.g. ImageNet [8] and Places 365 [51], but it is really dissatisfying to see the vast amounts of data, computing time and power consumption that are necessary to train deep networks. Fortunately, such convolutional networks, once trained on a large database, can be refined to solve related but different visual tasks by means of transfer learning, using fine-tuning [43,48].

Some form of knowledge is believed to be extracted by learning from the large-scale database of the source task and this knowledge is then transferred to the target task by initializing the network with the pre-trained parameters. However, we will show in the experimental section that some parameters may be driven far away from their initial values during fine-tuning. This leads to important losses of the initial knowledge that is assumed to be relevant for the targeted problem.

We argue that the standard L^2 regularization, which drives the parameters towards the origin, is not adequate in the framework of transfer learning, and thereby provides suboptimal results for the target problem. We advocate for a coherent parameter regularization approach, where the pre-trained model is both used as the starting point of the optimization process and as the reference in the penalty that encodes an explicit inductive bias, so as to help preserve the knowledge embedded in the initial network during fine-tuning. 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. We show that it has noticeable effects in transfer learning scenarios.

The penalties that encourage similarity with the *starting point* of the fine-tuning process will be denoted with the *SP* suffix. Despite the existence of several approaches akin to L^2-*SP*, many works disregard the inconsistency of using L^2 in transfer learning scenarios. In this paper, we evaluate -*SP* regularizers based on the L^2, Lasso and Group-Lasso penalties, which can freeze some individual parameters or groups of parameters to the pre-trained values. We also test the L^2-*SP* and Group-Lasso-*SP* variants that use the Fisher information to measure similarity. We elaborate on [26] by adding experimental evidences in classification and semantic segmentation, using several convolutional network architectures, and additional analyses. Our experiments indicate that all tested parameter regularization methods using the pre-trained parameters as a reference get an edge over the standard L^2 weight decay approach. We eventually recommend using L^2-*SP* as the standard baseline for transfer learning tasks when benchmarking new algorithms.

[*] Corresponding author.
E-mail addresses: xuhong.li@hds.utc.fr (X. Li), yves.grandvalet@hds.utc.fr (Y. Grandvalet), franck.davoine@hds.utc.fr (F. Davoine).

https://doi.org/10.1016/j.patcog.2019.107049
0031-3203/© 2019 Elsevier Ltd. All rights reserved.

2. Related work

The regularization scheme we advocate in this paper is related to several existing approaches. In this section, we first recall the techniques proposed for inductive transfer learning with convolutional networks. We then survey the regularizers that were proposed to encourage similarity of parameters or features across different tasks.

Regarding transfer learning, we follow the nomenclature of [36], who categorized several types of transfer learning according to domain and task settings during the transfer. A domain corresponds to the feature space and its distribution, whereas a task corresponds to the label space and its conditional distribution with respect to features. The initial learning problem is defined on the source domain and source task, whereas the new learning problem is defined on the target domain and the target task.

In the typology of Pan and Yang, we consider the inductive transfer learning setting, where the target domain is identical to the source domain, and the target task is different from the source task. We furthermore focus on the case where a vast amount of data was available for training on the source problem, and some limited amount of labeled data is available for solving the target problem. Under this setting, we aim at improving the performance on the target problem through parameter regularization methods that explicitly encourage the similarity of the solutions to the target and source problems. We also refer to works on new problems that were formalized or popularized after Pan and Yang, such as lifelong learning, but their typology remains valid.

2.1. Representation transfer in convolutional networks

Donahue et al. [10] selected the features computed at different layers of the pre-trained AlexNet [24] and plugged them into an SVM or a logistic regression classifier for learning a new task. This approach outperformed the state of the art of that time on the Caltech-101 database [13]. Similar approaches were proposed by Oquab et al. [35]. Later, Yosinski et al. [48] showed that fine-tuning the whole AlexNet resulted in better performances than using the network as a static feature extractor. Fine-tuning pre-trained VGG [43] on the image classification task of VOC-2012 [12] and Caltech 256 [18] achieved the best results of that time.

Ge and Yu [14] proposed a scheme for selecting a subset of images from the source problem that have similar local features to those in the target problem and then fine-tuned a pre-trained convolutional network. Besides image classification, many procedures for object detection [15,39,40] and image segmentation [6,30,50] have been proposed relying on fine-tuning to improve over training from scratch. These approaches showed promising results in a challenging transfer learning setup, as going from classification to object detection or image segmentation requires rather heavy modifications of the architecture of the network.

The success of transfer learning with convolutional networks relies on the generality of the learned representations that have been constructed from a large database like ImageNet. Yosinski et al. [48] also 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 [49]. Overall, the learned representations can be conveyed to related but different domains and the parameters in the network are reusable for different tasks.

All these state-of-the-art results were obtained while ignoring the inadequacy for transfer learning of the weight decay regularization term, which encourages deviations from the starting point. More appropriate regularizers have been applied in other circumstances, as described below, but, despite its simplicity and efficiency, the L^2-*SP* regularizer we advocate in this paper has never been considered for transferring representation in convolutional networks. To the best of our knowledge, we present the first results on transfer learning with convolutional networks that are simply based on this type of regularization term.

2.2. Regularizers for similar learning problems

Lifelong learning. In lifelong learning [37,45], a series of tasks is learned sequentially by a single model. The knowledge extracted from the previous tasks may be lost as new tasks are learned, resulting in what is known as catastrophic forgetting. In order to achieve good performance on all tasks, Li and Hoiem [27] proposed to use the outputs of the target examples, computed by the original network on the source task, to define a learning scheme retaining 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 they did not obtain interesting results.

Kirkpatrick et al. [23] developed a similar approach with success. They get sensible improvements by measuring the sensitivity of the parameters of the network learned on the source data thanks to the Fisher information. The Fisher information matrix defines a metric in the parameter space, which is used in their regularizer to preserve the representation learned on the source data, thereby retaining the knowledge acquired on the previous tasks. This scheme, named elastic weight consolidation, was shown to avoid forgetting, but fine-tuning with plain stochastic gradient descent was more effective than elastic weight consolidation for learning new tasks. Hence, elastic weight consolidation may be thought as being inadequate for transfer learning, where performance is only measured on the target task. We will show that this conclusion is not appropriate in typical transfer learning scenarios with few target examples.

Domain adaptation. In domain adaptation [31,32], the target task is identical to the source task and no (or few) target examples are labeled. Most approaches are searching for a common representation space for source and target domains to reduce domain shift, like [16,42]. Rozantsev et al. [41] proposed a parameter regularization scheme for encouraging the similarity of representations in the source and target domains. Their regularizer favors similar source and target parameters, up to a linear transformation. Encouraging similar parameters has also been proposed and shown to be helpful in speaker adaptation problems [28,34] and multilingual speech recognition [21].

Beyond deep networks. Regularization has been a means of building 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. 25, chapter 5], where it can be related to empirical Bayes arguments. Shrinking towards a reference has also been used in maximum entropy models [5] or SVM [3,46,47]. These approaches were shown to outperform standard L^2 regularization with limited labeled data in the target task [3,46]. They differ from the application to deep networks in several respects, the more important one being that they consider a fixed representation, with which transfer aims at producing similar classification parameters, 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.

3. Regularizers for fine-tuning

In this section, we detail the penalties we consider for fine-tuning. Parameter regularization is critical when learning from small databases. 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 information 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 inductive bias, encoded by a regularization term, affects the training process. Section 4 shows that all such schemes get an edge over the standard approaches that either use weight decay or freeze part of the network for preserving the low-level representations that are built in the first layers of the network.

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 J^Ω 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 J^Ω 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 minimization of J^Ω is a trade-off between the data-fitting term and the regularization term.

L^2 penalty. The current 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:

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

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

L^2-SP. 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 some 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:

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

L^2-SP-Fisher. Elastic weight consolidation [23] was proposed to avoid catastrophic forgetting in the setup of lifelong learning, where several tasks should be learned sequentially. In addition to preserving the initial parameter vector $\boldsymbol{w}^0$, it consists in using the estimated Fisher information to define the distance between $\boldsymbol{w}_{\mathcal{S}}$ and $\boldsymbol{w}_{\mathcal{S}}^0$. More precisely, it relies on the diagonal of the Fisher information matrix, resulting in the following penalty:

$$\Omega(\boldsymbol{w}) = \frac{\alpha}{2}\sum_{j\in\mathcal{S}}\hat{F}_{jj}\left(w_j - w_j^0\right)^2 + \frac{\beta}{2}\|\boldsymbol{w}_{\bar{\mathcal{S}}}\|_2^2, \tag{4}$$

where $\hat{F}_{jj}$ is the estimate of the jth diagonal element of the Fisher information matrix. It is computed as the average of the squared Fisher's score on the source problem, using the inputs of the source data:

$$\hat{F}_{jj} = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} f_k(\boldsymbol{x}^i;\boldsymbol{w}^0)\left(\frac{\partial}{\partial w_j}\log f_k(\boldsymbol{x}^i;\boldsymbol{w}^0)\right)^2,$$

where the outer average estimates the expectation with respect to inputs $\boldsymbol{x}^i$[] and the inner weighted sum is the estimate of the conditional expectation of outputs given input $\boldsymbol{x}^i$, with outputs drawn from a categorical distribution of parameters $(f_1(\boldsymbol{x}^i;\boldsymbol{w}),\ldots,f_k(\boldsymbol{x}^i;\boldsymbol{w}),\ldots,f_K(\boldsymbol{x}^i;\boldsymbol{w}))$.

L^1-SP. We also experiment the L^1 variant of L^2-*SP*:

$$\Omega(\boldsymbol{w}) = \alpha\left\|\boldsymbol{w}_{\mathcal{S}} - \boldsymbol{w}_{\mathcal{S}}^0\right\|_1 + \frac{\beta}{2}\|\boldsymbol{w}_{\bar{\mathcal{S}}}\|_2^2. \tag{5}$$

The usual L^1 penalty encourages sparsity; here, by using $\boldsymbol{w}_{\mathcal{S}}^0$ as a reference in the penalty, L^1-SP encourages some components of the parameter vector to be frozen, equal to the pre-trained initial values. The penalty can thus be thought as an intermediate between L^2-*SP* (3) and the strategies consisting in freezing a part of the initial network. We explore below other ways of doing so.

Group-Lasso-SP (GL-SP). Instead of freezing some individual parameters, we may encourage freezing some groups of parameters corresponding to channels of convolution kernels. Formally, we endow the set of parameters with a group structure, defined by a fixed partition of the index set $\mathcal{I} = \{1,\ldots,p\}$, that is, $\mathcal{I} = \bigcup_{g=0}^{G}\mathcal{G}_g$, with $\mathcal{G}_g \cap \mathcal{G}_h = \emptyset$ for $g \neq h$. In our setup, $\mathcal{G}_0 = \bar{\mathcal{S}}$, and for $g > 0$, $\mathcal{G}_g$ is the set of fan-in parameters of channel g. Let p_g denote the cardinality of group g, and $\boldsymbol{w}_{\mathcal{G}_g} \in \mathbb{R}^{p_g}$ be the vector $(w_j)_{j\in\mathcal{G}_g}$. Then, the *GL-SP* penalty is:

$$\Omega(\boldsymbol{w}) = \alpha\sum_{g=1}^{G}s_g\left\|\boldsymbol{w}_{\mathcal{G}_g} - \boldsymbol{w}_{\mathcal{G}_g}^0\right\|_2 + \frac{\beta}{2}\|\boldsymbol{w}_{\bar{\mathcal{S}}}\|_2^2, \tag{6}$$

where $\boldsymbol{w}_{\mathcal{G}_0}^0 = \boldsymbol{w}_{\bar{\mathcal{S}}}^0 \triangleq \boldsymbol{0}$, and, for $g > 0$, s_g is a predefined constant that may be used to balance the different cardinalities of groups. In our experiments, we used $s_g = p_g^{1/2}$.

Our implementation of Group-Lasso-*SP* can freeze feature extractors at any depth of the convolutional network, to preserve the pre-trained feature extractors as a whole instead of isolated pre-trained parameters. The group $\mathcal{G}_g$ of size $p_g = h_g \times \mathrm{w}_g \times d_g$ gathers all the parameters of a convolution kernel of height h_g, width w_g, and depth d_g. This grouping is done at each layer of the network, for each output channel, so that the group index g corresponds to two indexes in the network architecture: the layer index l and the output channel index at layer l. If we have c_l such channels at layer l, we have a total of $G = \sum_l c_l$ groups.

Group-Lasso-SP-Fisher (GL-SP-Fisher). Following the idea of L^2-*SP-Fisher*, the Fisher version of *GL-SP* is:

$$\Omega(\boldsymbol{w}) = \alpha\sum_{g=1}^{G}s_g\Big(\sum_{j\in\mathcal{G}_g}\hat{F}_{jj}\left(w_j - w_j^0\right)^2\Big)^{1/2} + \frac{\beta}{2}\|\boldsymbol{w}_{\mathcal{G}_0}\|_2^2.$$

4. Experimental results

We evaluate the aforementioned parameter regularizers for transfer learning on several pairs of source and target domains, and show the improvements of *-SP* regularizers on the standard L^2 in two different visual recognition tasks, image classification and semantic segmentation. We use ResNet [20] as our base network, since it has proven its wide applicability on transfer learning tasks.

4.1. Image classification

The source task is usually a classification task. Conventionally, if the target task is also a classification task, the fine-tuning process

Table 1
Characteristics of the target databases: name and type, numbers of training and test images per class, and number of classes.

Database	task category	# training	# test	# classes
Caltech 256–30	generic object recog.	30	20	257
Caltech 256–60	generic object recog.	60	20	257
MIT Indoors 67	scene classification	80	20	67
Stanford Dogs 120	specific object recog.	100	50	120
Foods 101	specific object recog.	750	250	101

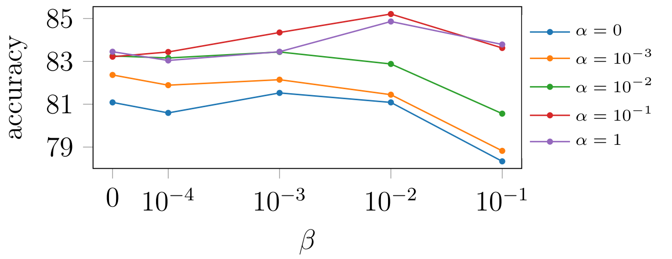

Fig. 1. Classification accuracy (in %) on Stanford Dogs 120 for L^2-*SP*, according to the two regularization hyperparameters α and β respectively applied to the layers inherited from the source task and the last classification layer (see Eq. (3)).

starts by replacing the last layer with a new one, randomly generated, whose size is defined by the number of classes in the target task.

4.1.1. Source and target databases

For comparing the effect of similarity between the source problem and the target problem on transfer learning, we chose two source databases: ImageNet [8] for generic object recognition and Places 365 [51] for scene classification. Likewise, we have four different databases related to four target problems: Caltech 256 [18] contains different objects for generic object recognition; MIT Indoors 67 [38] consists of 67 indoor scene categories; Stanford Dogs 120 [22] contains images of 120 breeds of dogs; Foods 101 [4] collects photos of 101 food categories, and is a much larger database than the previous ones (yet with some noise in terms of image quality and class labels). Each target database is split into training and test sets following the suggestion of their creators, except for Stanford Dogs 120, whose original test set is a subset of ImageNet, thereby biasing the evaluation of fine-tuning algorithms. To evaluate the test performance on Dogs, we use a part of the ImageNet validation set that contains those 120 breeds of dogs. Table 1 collects details for all target databases. In addition, we consider two configurations for Caltech 256: 30 or 60 examples randomly drawn from each category for training, and 20 remaining examples for the test set.

4.1.2. Training details

Most images in those databases are color images. If not, we create a three-channel image by duplicating the gray-scale data. All images are pre-processed: we resize images to 256 × 256 and subtract the mean activity computed over the training set from each channel, then we adopt random blur, random mirror and random crop to 224 × 224 for data augmentation. The network parameters are regularized as described in Section 3. Cross validation is used for choosing the best regularization hyperparameters α and β: α differs across experiments, and $\beta = 0.01$ is consistently picked by cross-validation for regularizing the last layer. Fig. 1 illustrates that the test accuracy varies smoothly according to the regularization strength, and that there is a sensible benefit in penalizing the last layer (that is, $\beta \geq 0$) for the best α values. When applicable, the Fisher information matrix is estimated on the source database. The two source databases (ImageNet or Places 365) yield different estimates. Regarding testing, we use central crops as inputs to compute the classification accuracy.

Stochastic gradient descent with momentum 0.9 is used for optimization. We run 9000 iterations and divide the learning rate by 10 after 6000 iterations. The initial learning rates are 0.005, 0.01 or 0.02, depending on the tasks. Batch size is 64. Then, under the best configuration, we repeat five times the learning process to obtain an average classification accuracy and standard deviation. All the experiments are performed with Tensorflow [1]. The source code is publicly available for reproducibility purposes.[1]

4.1.3. Comparison across penalties, source and target databases

A comprehensive view of our experimental results is given in Fig. 2. Each plot corresponds to one of the four target databases listed in Table 1. The red points mark the accuracies of transfer learning when using Places 365 as the source database, whereas the blue points correspond to the results obtained with ImageNet. As expected, the results of transfer learning are much better when source and target are alike: the scene classification target task MIT Indoor 67 (top left) is better transferred from the scene classification source task Places 365, whereas the object recognition target tasks benefit more from the object recognition source task ImageNet. Besides showing that choosing an appropriate source domain is critical in transfer learning (see [2,9] for example), for our purpose of evaluating regularizers, these results display similar trends for the two source databases: all the fine-tuning strategies based on penalties using the starting point -*SP* as a reference perform consistently better than standard fine-tuning (L^2). There is thus a benefit in having an explicit bias towards the starting point, even when the target task is not too similar to the source task.

Interestingly, the best source database for Foods 101 is Places 365 with L^2 regularization and ImageNet for the penalties using the starting point -*SP* as a reference. Considering the relative failure of L^2-*SP-Fisher*, it is likely that Foods 101 is quite far from the two sources but slightly closer to ImageNet.

The benefit of the explicit bias towards the starting point is comparable for L^2-*SP* and L^2-*SP-Fisher* penalties; the strategies based on L^1 and Group-Lasso penalties behave rather poorly in comparison. They are even less accurate than the plain L^2 strategy on Caltech 256–30 when the source problem is Places 365. Stochastic gradient descent does not handle well these penalties whose gradient is discontinuous at the starting point where the optimization starts. The stochastic forward-backward splitting algorithm [11], which is related to proximal methods, leads to substandard results, presumably due to the absence of a momentum term. In the end, we used plain stochastic gradient descent on a smoothed version of the penalties eliminating the discontinuities of their gradients, but some instability remains.

4.1.4. Fine-tuning from a similar source

Table 2 displays the results of fine-tuning with L^2-*SP* and L^2-*SP-Fisher*, which are compared to the current baseline of fine-tuning with L^2, and the state-of-the-art references [14,33]. We report the average accuracies and their standard deviations on 5 different runs. Since we use the same data and the same starting point, runs differ only due to the randomness of stochastic gradient descent and to the weight initialization of the last layer.

In the first part of Table 2 (first three lines), we observe that L^2-*SP* and L^2-*SP-Fisher* always improve over L^2 by a clear margin, and that this improvement is even more important when less data are available for the target problem (Caltech–30 *vs.* Caltech–60 and Foods *vs.* others). When fewer training examples are available for the target problem, the role of the regularizer is more important.

[1] https://github.com/holyseven/TransferLearningClassification.

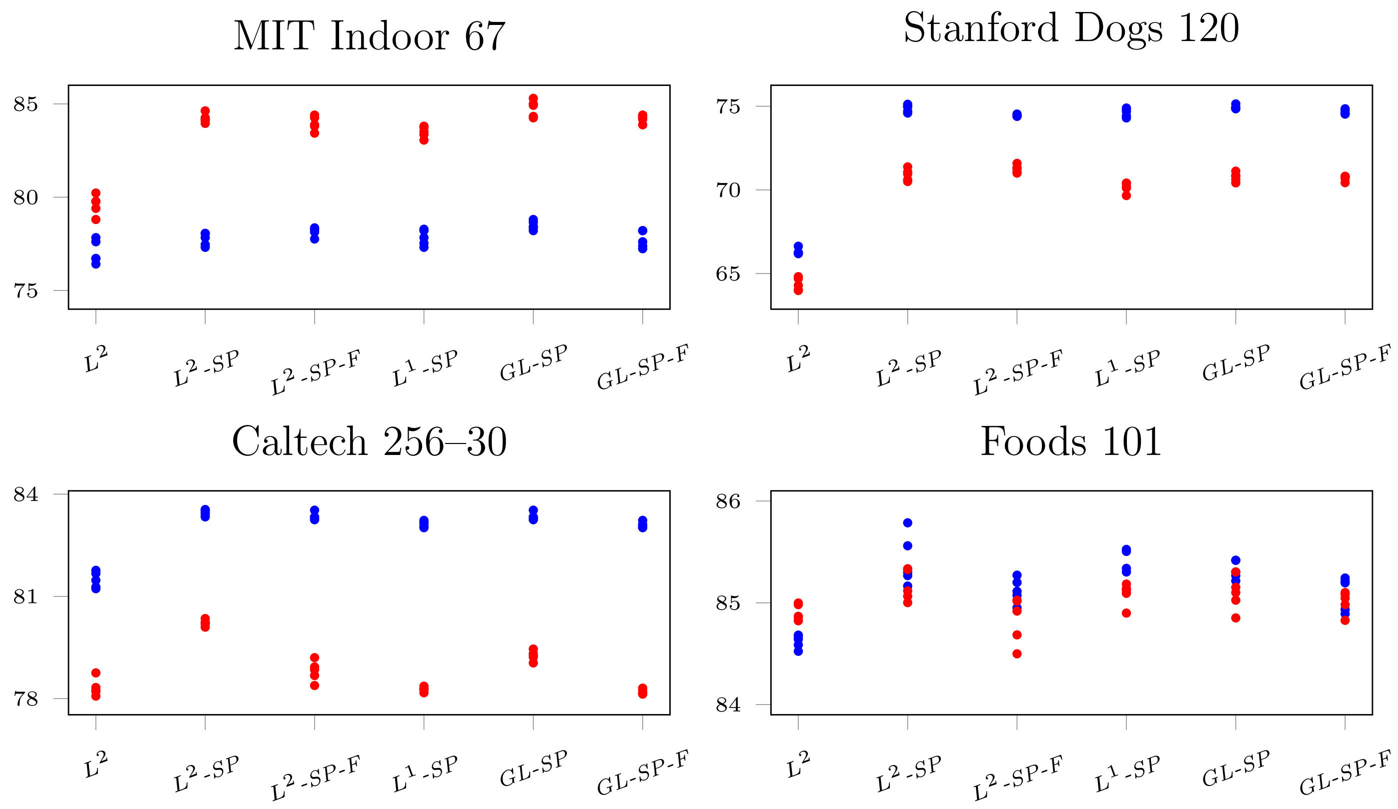

Fig. 2. Classification accuracies (in %) of the tested fine-tuning approaches on the four target databases, using ImageNet (dark blue dots) or Places 365 (light red dots) as source databases. MIT Indoor 67 is more similar to Places 365 than to ImageNet; Stanford Dogs 120, Caltech 256 and Foods 101 are more similar to ImageNet than to Places 365. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Meanwhile, little difference is observed between L^2-*SP* and L^2-*SP-Fisher*. Note that we do not report here the performances of training from scratch, but that transfer learning really helps in these setups: we could only reach 76.9% accuracy on Foods 101 (with 10 times more computing efforts, that is, number of epochs).

In the second part of Table 2, we boost the performance of fine-tuning with L^2, L^2-*SP* and L^2-*SP-Fisher* by exploiting additional training and post-processing techniques, that is, by respecting the aspect ratio of images and by using the 10-crop test, which were not used in this paper except here. We apply these techniques, resizing images with the shorter edge being 256 and keeping the aspect ratio (the standard resizing technique ignores it) for training, and averaging the predictions of 10 cropped patches (the center patch, the four corner patches, and all their horizontal reflections) for testing. The improved results are above state of the art for Caltech–30, and close to state of the art for Indoors, without making use of the advanced techniques employed by Ge and Yu [14,33]. These results show that simply changing the regularizer from L^2 to L^2-*SP* or L^2-*SP-Fisher* is remarkably efficient not only for baseline models, but also for more advanced ones.

4.2. *Semantic image segmentation*

We now evaluate L^2 and L^2-*SP* on transfer learning from classification to segmentation. The task of semantic segmentation differs substantially from classification, thereby requiring important modifications to the network structure. Despite these differences, segmentation still benefits a lot from transfer from an image classification source task.

4.2.1. *Source and target databases*

We used large databases, ImageNet [8] and Microsoft COCO [29], as source databases. ImageNet targets image classification; Microsoft COCO is for object detection and semantic image segmentation. Pre-training on ImageNet can largely increase the performance for most image-related learning tasks, moreover pre-training on ImageNet and then on COCO can further raise segmentation performance. Both pre-training schemes are evaluated.

Two databases for semantic image segmentation are used as targets: Cityscapes [7] and Semantic Boundaries Dataset (SBD) [19]. Cityscapes is a database with an evaluation benchmark for pixel-wise segmentation of real-world urban street scenes. It consists of 20,000 images with coarse annotations, and 5000 images with high quality pixel-wise labeling, which are split into a training set (2975 images), a validation set (500 images) and a test set (1525 images). All images in Cityscapes have a 2048 × 1024 pixel resolution. SBD is an augmented version of the Pascal VOC segmentation database [12], resulting in 10582, 1449, and 1456 images for training, validation, and testing respectively. All images in SBD have a resolution no larger than 500 × 500 pixels, and 20 different categories are considered, plus one for the background.

4.2.2. *Training details*

Most training techniques for segmentation are borrowed from classification and we only present here the differences in Table 3. The full source code is also available.[2] We consider four convolutional networks for image segmentation. FCN [30] is one of the most classical structures for segmentation. ResNet [20] can also be used for image segmentation by removing the global pooling layer. DeepLab [6] and PSPNet [50] stayed top-ranked for some time on the Cityscapes and Pascal VOC benchmarks and are two favored structures.

[2] https://github.com/holyseven/PSPNet-TF-Reproduce.

4.2.3. Results

With L^2 and L^2-*SP*, under the same setting, we use all these architectures for Cityscapes and PSPNet for SBD. Note that a few learning settings, like batch size and crop size, differ from the original work of [50]. This is not essential for our purpose, since we aim at demonstrating the consistency of the improvements, throughout diverse network structures (FCN, ResNet-101, DeepLab, and PSPNet) of L^2-*SP* compared to L^2 regularization.

Table 4 reports the results on Cityscapes validation set. We fine-tuned FCN, ResNet, DeepLab and PSPNet with the standard L^2, and L^2-*SP*, all other things being equal. We readily observe that fine-tuning with L^2-*SP* in place of L^2 consistently improves the performance in mean Intersection over Union (mIoU) score, for all networks. The best model (PSPNet-extra with L^2-*SP*) has been evaluated on the test set and is currently on the public benchmark of Cityscapes,[3] with 80.3% mIoU, to be compared to 80.2% obtained by Zhao et al. [50].

For PSPNet on SBD, we apply the same protocol, except that images are cropped to 480 × 480, enabling a larger batch size of 16. The results on the public validation set are again in favor of L^2-*SP*, which reaches 79.9% in mIoU compared to 78.3% for L^2. On the test set, L^2-*SP* reaches 79.8%[4].

5. Analysis and discussion

Having confirmed the versatility of -*SP* regularizers on classification and segmentation tasks, with different network architectures, we now analyze their behavior. Among all -*SP* methods, L^2-*SP* and L^2-*SP-Fisher* always reach a better accuracy on the target task. We expected L^2-*SP-Fisher* to outperform L^2-*SP* since Fisher information provides a relevant metric in parameter space and was shown to help in lifelong learning, but there is no significant difference between the two options in our setups. Since L^2-*SP* is simpler than L^2-*SP-Fisher*, we recommend the former, and we focus on the analysis of L^2-*SP*, although most of the discussion would also apply to L^2-*SP-Fisher*.

5.1. Behavior on the source task

The variants using the Fisher information matrix behave like the simpler variants using a Euclidean metric on parameters. One reason is that, contrary to lifelong learning, our objective does not favor solutions that retain accuracy on the source task. Hence, the metric defined by the Fisher information matrix is less relevant for our actual objective that only relates to the target task. Table 5 reports the drop in performance when the fine-tuned models are applied on the source task, without any retraining, simply using the original classification layer instead of the classification layer learned for the target task. The performance drop is consistently smaller for L^2-*SP-Fisher* than for L^2-*SP*. This confirms that L^2-*SP-Fisher* is indeed a better approach in the situation of lifelong learning, where accuracies on the source tasks matter. In comparison to L^2-*SP-Fisher* and L^2-*SP*, L^2 fine-tuning results in catastrophic forgetting: the performance on the source task is considerably affected by fine-tuning.

The relative drops in performance with Foods 101 follow the pattern observed for the other databases except that the decrease is much larger. This may be a sign of the substantial divergence of the data distribution of Foods 101 from the one of ImageNet, with a compromise between the source task and the target task met far from the starting point.

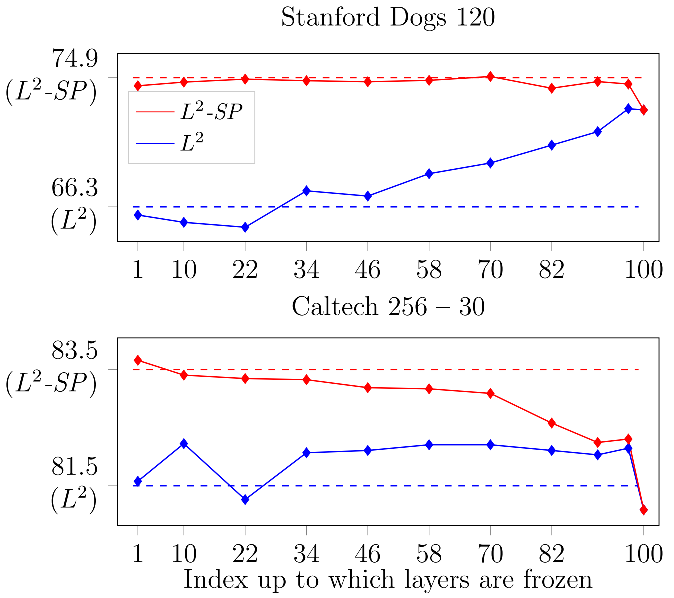

Fig. 3. Classification accuracies (in %) of fine-tuning with L^2 and L^2-*SP* on Stanford Dogs 120 (top) and Caltech 256–30 (bottom) when freezing the first layers of ResNet-101. The dashed lines represent the accuracies reported in Table 2, where no layers are frozen. ResNet-101 begins with one convolutional layer, then stacks 3-layer blocks. The three layers in one block are either frozen or trained altogether.

5.2. *fine-tuning* vs. *freezing the network*

Freezing the first layers of a network during transfer learning [48] is another way to ensure a very strong inductive bias, letting fewer degrees of freedom to transfer learning. Fig. 3 shows that this strategy, which is costly to implement if one looks for the optimal number of layers to be frozen, can improve L^2 fine-tuning considerably, but that it is rather inefficient for L^2-*SP* fine-tuning. Among all possible choices, L^2 fine-tuning with partial freezing is dominated by the plain L^2-*SP* fine-tuning. Note that L^2-*SP-Fisher* (not displayed) behaves similarly to L^2-*SP*.

5.3. Layer-wise analysis

We complement our experimental results by an analysis relying on the activations of the hidden units of the network, to provide another view on the differences between L^2 and L^2-*SP* fine-tuning. Activation similarities are easier to interpret than parameter similarities, as they provide a view of the network that is closer to the functional perspective we are actually pursuing. Matching individual activations makes sense, provided that the networks slightly differ before and after tuning so that few roles are switched between units or feature maps.

The dependency between the pre-trained and the fine-tuned activations throughout the network is displayed in Fig. 4, with boxplots of the R^2 coefficients, gathered layer-wise, of the fine-tuned activations with respect to the original activations. This figure shows that, indeed, the roles of units or feature maps have not changed much after L^2-*SP* and L^2-*SP-Fisher* fine-tuning. The R^2 coefficients are very close to 1 on the first layers, and smoothly decrease throughout the network, staying quite high, around 0.6, for L^2-*SP* and L^2-*SP-Fisher* at the greatest depth. In contrast, for L^2 regularization, some important changes are already visible in the first layers, and the R^2 coefficients eventually reach quite low values at the greatest depth. This illustrates in detail how the roles of the network units are remarkably retained with L^2-*SP* and L^2-*SP-Fisher* fine-tuning, not only for the first layers of the networks, but also for the last high-level representations before classification.

[3] https://www.cityscapes-dataset.com/method-details/?submissionID=1148.
[4] http://host.robots.ox.ac.uk:8080/anonymous/NAAVTI.html.

Table 2
Average classification accuracies (in %) of L^2, L^2-*SP* and L^2-*SP-Fisher* on 5 different runs. The source database is Places 365 for MIT Indoors 67 and ImageNet for Caltech 256, Stanford Dogs and Foods. References of the state of the art are taken from [14], except for Foods-101 where it is from [33]. For Dogs, there is no reference that is based on the test set used here to avoid the overlap with the ImageNet training set. Enhanced variants respecting the aspect ratio and using the 10-crop test are marked with a star (*). Results with the highest accuracy in each part are highlighted in bold.

	Caltech-30	Caltech-60	Indoors	Dogs	Foods
L^2	81.5 ± 0.2	85.3 ± 0.2	79.6 ± 0.5	66.3 ± 0.2	84.6 ± 0.1
L^2-*SP*	**83.5 ± 0.1**	**86.4 ± 0.2**	**84.2 ± 0.3**	**74.9 ± 0.2**	**85.4 ± 0.3**
L^2-*SP-Fisher*	83.3 ± 0.1	86.0 ± 0.1	84.0 ± 0.4	74.4 ± 0.1	85.1 ± 0.1
L^{2*}	82.7 ± 0.2	86.5 ± 0.4	80.7 ± 0.9	67.7 ± 0.3	86.7 ± 0.2
L^2-*SP**	**84.9 ± 0.1**	**87.9 ± 0.2**	**85.2 ± 0.3**	**77.1 ± 0.2**	**87.1 ± 0.1**
L^2-*SP-Fisher**	84.8 ± 0.1	**87.9 ± 0.1**Â	**85.2 ± 0.1**	76.9 ± 0.1	87.0 ± 0.1
Reference	83.8 ± 0.5	89.1 ± 0.2	85.8	–	90.3

Table 3
Training and test details for segmentation on Cityscapes. Abbreviations used in this table: lr - learning rate; poly lr - polynomial learning rate policy; bs - batch size; bn - batch normalization; rdm scale - random scale; ms test - multi-scale test.

		FCN	ResNet	DeepLab	PSPNet
training	lr policy	fixed lr	poly lr		
	bs × h × w	2 × 800 × 800			8 × 836 × 836
	bn stats	frozen but trained β and γ			all training
	rdm scale	no			[0.5, 2.0]
test	ms test	no			yes
	image size	whole image			836 × 836 crops

Table 4
Mean Intersection over Union scores (in %) on Cityscapes validation set. Note that the initial model for DeepLab-COCO is pre-trained on ImageNet and then on Microsoft COCO, others are only pre-trained on ImageNet, and that PSPNet-extra uses 20,000 extra coarsely labeled images of Cityscapes for training whereas PSPNet only uses the training set (finely labeled images). Results with higher mIoU scores are highlighted in bold.

Method	L^2	L^2-*SP*
FCN	66.9	**67.9**
ResNet-101	68.1	**68.7**
DeepLab	68.6	**70.4**
DeepLab-COCO	72.0	**73.2**
PSPNet	78.2	**79.4**
PSPNet-extra	80.9	**81.2**

Table 5
Classification accuracy drops (in %, the lower, the better) on the source tasks due to fine-tuning based on L^2, L^2-*SP* and L^2-*SP-Fisher* regularizers. The source database is Places 365 for MIT Indoors 67 and ImageNet for Caltech 256, Stanford Dogs and Foods 101. The classification accuracies of the pre-trained models are 54.7% and 76.7% on Places 365 and ImageNet respectively. Results with the lowest drops are highlighted in bold.

	L^2	L^2-*SP*	L^2-*SP-Fisher*
MIT Indoors 67	24.1	5.3	**4.9**
Caltech 256–30	15.4	4.2	**3.6**
Caltech 256–60	16.9	3.6	**3.2**
Stanford Dogs 120	14.1	4.7	**4.2**
Foods 101	68.6	64.5	**53.2**

We now look at the diagonal elements of the Fisher information matrix, still computed on ResNet-101 from training inputs of ImageNet.

Their distributions across layers, displayed in Fig. 5, show that the network is more sensitive to the parameters of the first layers, with a high disparity within these layers, and are then steady with most values within one order of magnitude. As a result, L^2-*SP-Fisher* is very similar to L^2-*SP*, except for being more conservative on the first layers. This observation explains the small differences between L^2-*SP* and L^2-*SP-Fisher* that are observed in our transfer learning setups.

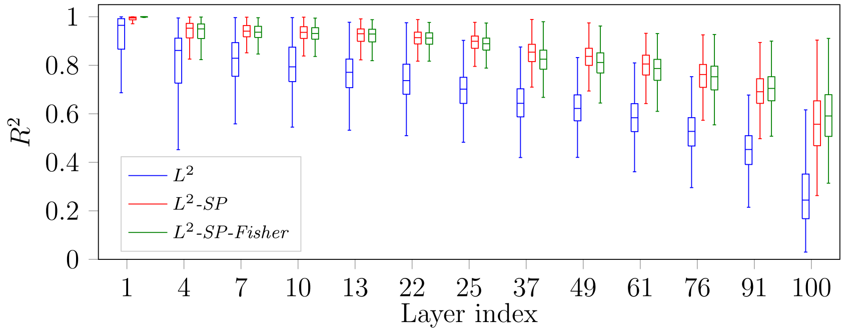

Fig. 4. R^2 coefficients of determination with L^2 and L^2-*SP* regularizations for Stanford Dogs 120. Each boxplot summarizes the distribution of the R^2 coefficients of the activations after fine-tuning with respect to the activations of the pre-trained network, for all the units in one layer. ResNet-101 begins with one convolutional layer, then stacks 3-layer blocks. We display here only the R^2 at the first layer and at the outputs of some 3-layer blocks.

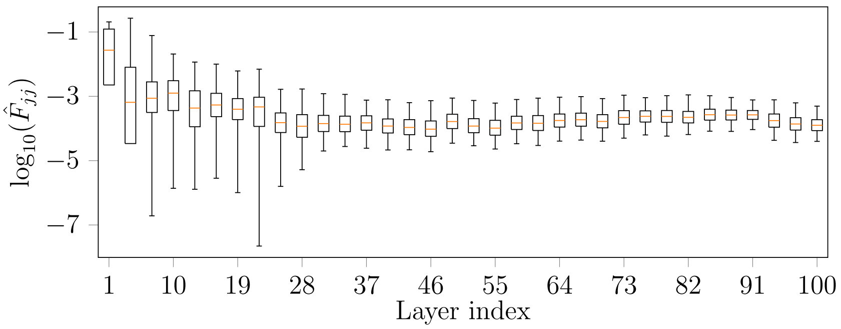

Fig. 5. Boxplots of the diagonal elements of the Fisher information matrix (log-scale) computed on the training set of ImageNet using the pre-trained model. We display here these elements at the first layer and then at the last layer of all 3-layer blocks of ResNet-101.

5.4. Computational efficiency

The *-SP* penalties introduce no extra parameters, and they only increase slightly the computational burden. L^2*-SP* increases the number of floating point operations required for a learning step of ResNet-101 by less than 1%. Hence, at a negligible computational cost, we can obtain significant improvements in classification accuracy, and no additional cost is experienced at test time.

5.5. Theoretical insights

5.5.1. Effect of L^2-SP

Analytical results are very difficult to obtain 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 [see, e.g. 17, 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.

A similar analysis can be used for L^2*-SP* regularization. Let J be the unregularized objective function and $J^{SP}(\boldsymbol{w}) = J(\boldsymbol{w}) + \frac{\alpha}{2}\left\|\boldsymbol{w} - \boldsymbol{w}^0\right\|_2^2$ be the regularized objective function. Let $\widehat{\boldsymbol{w}} = \text{argmin}_{\boldsymbol{w}} J(\boldsymbol{w})$ and $\widehat{\boldsymbol{w}}^{\text{SP}} = \text{argmin}_{\boldsymbol{w}} J^{SP}(\boldsymbol{w})$ be their respective minima. The quadratic approximation of $J(\widehat{\boldsymbol{w}})$ gives

$$\mathbf{H}(\widehat{\boldsymbol{w}}^{\text{SP}} - \widehat{\boldsymbol{w}}) + \alpha(\widehat{\boldsymbol{w}}^{\text{SP}} - \boldsymbol{w}^0) = 0, \tag{7}$$

where $\mathbf{H}$ is the Hessian matrix of J w.r.t. $\boldsymbol{w}$, evaluated at $\widehat{\boldsymbol{w}}$. Since $\mathbf{H}$ is symmetric and positive semidefinite, it can be decomposed as $\mathbf{H} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^{\mathsf{T}}$. Applying the decomposition to Eq. (7), we obtain the following relationship between $\widehat{\boldsymbol{w}}^{\text{SP}}$ and $\widehat{\boldsymbol{w}}$:

$$\mathbf{P}^{\mathsf{T}}\widehat{\boldsymbol{w}}^{\text{SP}} = (\boldsymbol{\Lambda} + \alpha\mathbf{I})^{-1}\boldsymbol{\Lambda}\mathbf{P}^{\mathsf{T}}\widehat{\boldsymbol{w}} + \alpha(\boldsymbol{\Lambda} + \alpha\mathbf{I})^{-1}\mathbf{P}^{\mathsf{T}}\boldsymbol{w}^0. \tag{8}$$

This equation shows that, in the direction defined by the i-th eigenvector of $\mathbf{H}$, $\widehat{\boldsymbol{w}}^{\text{SP}}$ is a convex combination of the projections of $\widehat{\boldsymbol{w}}$ and $\boldsymbol{w}^0$ on that direction. Indeed noting λ_i the eigenvalue corresponding to the i-th eigenvector, the terms of the convex combination are $\frac{\lambda_i}{\lambda_i+\alpha}$ and $\frac{\alpha}{\lambda_i+\alpha}$.

This contrasts with L^2 that leads to a trade-off between the optimum of the unregularized objective function and the origin. Clearly, searching for a solution in the vicinity of the pre-trained parameters 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.

5.5.2. Bias-variance analysis

We propose here a simple bias-variance analysis for the case of linear regression, for which this analysis is tractable. Consider the squared loss function $J(\boldsymbol{w}) = \frac{1}{2}\|\mathbf{X}\boldsymbol{w} - \boldsymbol{y}\|^2$, where $\boldsymbol{y} \in \mathbb{R}^n$ is a vector of continuous responses, and $\mathbf{X} \in \mathbb{R}^{n\times p}$ is the matrix of predictor variables. We use the standard assumptions of the fixed design case, that is: (i) $\boldsymbol{y}$ is the realization of a random variable $\mathbf{Y}$ such that $\mathbb{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{w}^*$, $\mathbb{V}[\mathbf{Y}] = \sigma^2\mathbf{I}_n$, and $\boldsymbol{w}^*$ is the vector of true parameters; (ii) the design is fixed and orthonormal, that is, $\mathbf{X}^T\mathbf{X} = \mathbf{I}_p$. We also assume that the reference we use for L^2*-SP, i.e.* $\boldsymbol{w}^0$, is not far away from $\boldsymbol{w}^*$, since it is the minimizer of the unregularized objective function on a large data set: $\boldsymbol{w}^0 = \boldsymbol{w}^* + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon}$, the difference between the two parameters, is supposed to be relatively small, *i.e.* $\|\boldsymbol{\varepsilon}\| \ll \|\boldsymbol{w}^*\|$.

We consider the three estimates $\widehat{\boldsymbol{w}} = \text{argmin}_{\boldsymbol{w}} J(\boldsymbol{w})$, $\widehat{\boldsymbol{w}}^{\text{L}^2} = \text{argmin}_{\boldsymbol{w}} J(\boldsymbol{w}) + \frac{\alpha}{2}\|\boldsymbol{w}\|_2^2$ and $\widehat{\boldsymbol{w}}^{\text{SP}} = \text{argmin}_{\boldsymbol{w}} J(\boldsymbol{w}) + \frac{\alpha}{2}\left\|\boldsymbol{w} - \boldsymbol{w}^0\right\|_2^2$. Their closed-form formulations are respectively:

$$\begin{cases} \widehat{\boldsymbol{w}} = \mathbf{X}^{\mathsf{T}}\boldsymbol{y} \\ \widehat{\boldsymbol{w}}^{\text{L}^2} = \frac{1}{1+\alpha}\mathbf{X}^{\mathsf{T}}\boldsymbol{y} \\ \widehat{\boldsymbol{w}}^{\text{SP}} = \frac{1}{1+\alpha}\mathbf{X}^{\mathsf{T}}\boldsymbol{y} + \frac{\alpha}{1+\alpha}\boldsymbol{w}^0 \end{cases} \tag{9}$$

So that their expectations and variances are:

$$\begin{cases} \mathbb{E}[\widehat{\boldsymbol{w}}] = \boldsymbol{w}^* \\ \mathbb{E}[\widehat{\boldsymbol{w}}^{\text{L}^2}] = \frac{1}{1+\alpha}\boldsymbol{w}^* \\ \mathbb{E}[\widehat{\boldsymbol{w}}^{\text{SP}}] = \frac{1}{1+\alpha}\boldsymbol{w}^* + \frac{\alpha}{1+\alpha}\boldsymbol{w}^0 \\ \qquad\quad = \boldsymbol{w}^* + \frac{\alpha}{1+\alpha}\boldsymbol{\varepsilon} \end{cases} \tag{10}$$

$$\begin{cases} \mathbb{V}[\widehat{\boldsymbol{w}}] = \sigma^2\mathbf{I}_p \\ \mathbb{V}[\widehat{\boldsymbol{w}}^{\text{L}^2}] = \left(\frac{\sigma}{1+\alpha}\right)^2\mathbf{I}_p \\ \mathbb{V}[\widehat{\boldsymbol{w}}^{\text{SP}}] = \left(\frac{\sigma}{1+\alpha}\right)^2\mathbf{I}_p \end{cases} \tag{11}$$

These expressions show that, without any regularization, the least squared estimate $\widehat{\boldsymbol{w}}$ is unbiased, but with the largest variance. With the L^2 regularizer, the variance is decreased by a factor of $1/(1+\alpha)^2$ but the squared bias is $\|\boldsymbol{w}^*\|^2\alpha^2/(1+\alpha)^2$. The L^2*-SP* regularizer benefits from the same decrease of variance and suffers from the smaller squared bias $\|\boldsymbol{\varepsilon}\|^2\alpha^2/(1+\alpha)^2$. It is thus always a better option than L^2 (provided the assumption $\|\boldsymbol{\varepsilon}\| \ll \|\boldsymbol{w}^*\|$ holds), and it is the best option regarding squared error when $\|\boldsymbol{\varepsilon}\|\alpha < \sigma$, which is likely when the sample size on the source task is much larger than the sample size on the target task.

5.5.3. Shrinkage estimation

Using L^2*-SP* instead of L^2 can also be motivated by an analogy with shrinkage estimation [see e.g. 25, 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 parameters” is not readily 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 more relevant reference than the single parameter value provided by the pre-trained network.

6. Conclusion

We described and tested simple regularization techniques for transfer learning with convolutional networks. They all encode an explicit bias towards the solution learned on the source task, resulting in a trade-off between the solution to the target task and the pre-trained parameter that is coherent with the original motivation for fine-tuning. All the regularizers evaluated here have been already used for other purposes or in other contexts, but we demonstrated their relevance for inductive transfer learning with deep convolutional networks.

We show that a simple L^2 penalty using the starting point as a reference, L^2-*SP*, is useful, even if early stopping is used. This penalty is much more effective than the standard L^2 penalty that is commonly used in fine-tuning. It is also more effective and simpler to implement than the strategy consisting in freezing the first layers of a network. We provide theoretical hints and strong experimental evidence showing that L^2-*SP* retains the memory of the features learned on the source database. We thus believe that this simple L^2-*SP* scheme should be considered as the standard baseline in inductive transfer learning, and that future improvements of transfer learning should rely on this baseline.

Besides, we tested the effect of more elaborate penalties, based on L^1 norm, Group-L^1 norm, or Fisher information. None of the L^1 or Group-L^1 options seem to be valuable in the context of inductive transfer learning that we considered here, and using the Fisher information with L^2-*SP*, though being better at preserving the memory of the source task, does not improve accuracy on the target task. Different approaches, which implement an implicit bias at the functional level, alike [27,32], remain to be tested: being based on a different principle, their value should be assessed in the framework of inductive transfer learning.

Acknowledgments

This work was carried out with the supports of the China Scholarship Council and of a PEPS grant through the DESSTOPT project jointly managed by the National Institute of Mathematical Sciences and their Interactions (INSMI) and the Institute of Information Science and their Interactions (INS2I) of the CNRS, France. It was carried out in the framework of SIVALab, a joint laboratory between Renault and Heudiasyc (UTC/CNRS). We acknowledge the support of NVIDIA Corporation with the donation of GPUs used for this research.

References

[1] 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, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015, Software available from tensorflow.org.

[2] M.J. Afridi, A. Ross, E.M. Shapiro, On automated source selection for transfer learning in convolutional neural networks, Pattern Recognit. 73 (2018) 65–75.

[3] Y. Aytar, A. Zisserman, Tabula RASA: Model transfer for object category detection, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2011, pp. 2252–2259.

[4] L. Bossard, M. Guillaumin, L. Van Gool, Food-101–mining discriminative components with random forests, in: Proceedings of the European Conference on Computer Vision (ECCV), 2014, pp. 446–461.

[5] C. Chelba, A. Acero, Adaptation of maximum entropy capitalizer: little data can help a lot, Comput. Speech Lang. 20 (4) (2006) 382–399.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille, Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE Trans. Pattern Anal. Mach. Intell. 40 (4) (2018) 834–848.

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The Cityscapes dataset for semantic urban scene understanding, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3213–3223.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: a large-scale hierarchical image database, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 248–255.

[9] Z. Ding, M. Shao, Y. Fu, Incomplete multisource transfer learning, IEEE Trans. Neural Netw. Learn. Syst. 29 (2) (2018) 310–323.

[10] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: a deep convolutional activation feature for generic visual recognition, in: Proceedings of the International Conference on Machine Learning (ICML), 2014, pp. 647–655.

[11] J. Duchi, Y. Singer, Efficient online and batch learning using forward backward splitting, J. Mach. Learn. Res. 10 (2009) 2899–2934.

[12] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (VOC) challenge, Int. J. Comput. Vis. 88 (2) (2010) 303–338.

[13] L. Fei-Fei, R. Fergus, P. Perona, One-shot learning of object categories, IEEE Trans. Pattern Anal. Mach. Intell. 28 (4) (2006) 594–611.

[14] W. Ge, Y. Yu, Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 10–19.

[15] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 580–587.

[16] B. Gong, Y. Shi, F. Sha, K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 2066–2073.

[17] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, Adaptive Computation and Machine Learning, MIT Press, 2017.

[18] G. Griffin, A. Holub, P. Perona, Caltech-256 object category dataset, Technical Report, California Institute of Technology, 2007.

[19] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, J. Malik, Semantic contours from inverse detectors, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2011, pp. 991–998.

[20] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.

[21] J.-T. Huang, J. Li, D. Yu, L. Deng, Y. Gong, Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 7304–7308.

[22] A. Khosla, N. Jayadevaprakash, B. Yao, F.-F. Li, Novel dataset for fine-grained image categorization: Stanford dogs, in: Proceedings of the CVPR Workshop on Fine-Grained Visual Categorization (FGVC), 2011.

[23] 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, Proc. Natl. Acad. Sci. 114 (13) (2017) 3521–3526.

[24] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Proceedings of the Advances in Neural Information Processing Systems (NIPS), 2012, pp. 1097–1105.

[25] E.L. Lehmann, G. Casella, Theory of Point Estimation, second ed., Springer, 1998.

[26] X. Li, Y. Grandvalet, F. Davoine, Explicit inductive bias for transfer learning with convolutional networks, in: Proceedings of the International Conference on Machine Learning (ICML), 2018, pp. 2830–2839.

[27] Z. Li, D. Hoiem, Learning without forgetting, IEEE Trans. Pattern Anal. Mach. Intell. (2017), doi:10.1109/TPAMI.2017.2773081.

[28] H. Liao, Speaker adaptation of context dependent deep neural networks, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2013, pp. 7947–7951.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft COCO: common objects in context, in: Proceedings of the European Conference on Computer Vision (ECCV), 2014, pp. 740–755. Zurich

[30] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3431–3440.

[31] M. Long, Y. Cao, J. Wang, M. Jordan, Learning transferable features with deep adaptation networks, in: Proceedings of the International Conference on Machine Learning (ICML), 2015, pp. 97–105.

[32] M. Long, H. Zhu, J. Wang, M.I. Jordan, Unsupervised domain adaptation with residual transfer networks, in: Proceedings of the Advances in Neural Information Processing Systems (NIPS), 2016, pp. 136–144.

[33] N. Martinel, G.L. Foresti, C. Micheloni, Wide-slice residual networks for food recognition, in: Proceedings of the Winter Conference on Applications of Computer Vision (WACV), 2018, pp. 567–576.

[34] T. Ochiai, S. Matsuda, X. Lu, C. Hori, S. Katagiri, Speaker adaptive training using deep neural networks, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2014, pp. 6349–6353.

[35] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1717–1724.

[36] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.

[37] A. Pentina, C.H. Lampert, Lifelong learning with non-iid tasks, in: Proceedings of the Advances in Neural Information Processing Systems (NIPS), 2015, pp. 1540–1548.
[38] A. Quattoni, A. Torralba, Recognizing indoor scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 413–420.
[39] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788.
[40] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, in: Proceedings of the Advances in Neural Information Processing Systems (NIPS), 2015, pp. 91–99.
[41] A. Rozantsev, M. Salzmann, P. Fua, Beyond sharing weights for deep domain adaptation, IEEE Trans. Pattern Anal. Mach. Intell. 41 (4) (2019) 801–814.
[42] M. Shao, D. Kit, Y. Fu, Generalized transfer subspace learning through low-rank constraint, Int. J. Comput. Vis. 109 (1–2) (2014) 74–93.
[43] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Proceedings of the International Conference on Learning Representations (ICLR), 2015.
[44] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2818–2826.
[45] S. Thrun, T.M. Mitchell, Lifelong robot learning, Robot. Auton. Syst. 15 (1–2) (1995) 25–46.
[46] T. Tommasi, F. Orabona, B. Caputo, Learning categories from few examples with multi model knowledge transfer, IEEE Trans. Pattern Anal. Mach. Intell. 36 (5) (2014) 928–941.
[47] J. Yang, R. Yan, A.G. Hauptmann, Adapting SVM classifiers to data with shifted distributions, in: Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW), 2007, pp. 69–76.
[48] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? in: Proceedings of the Advances in Neural Information Processing Systems (NIPS), 2014, pp. 3320–3328.
[49] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: Proceedings of the European Conference on Computer Vision (ECCV), 2014, pp. 818–833.
[50] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2881–2890.
[51] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, A. Torralba, Places: a 10 million image database for scene recognition, IEEE Trans. Pattern Anal. Mach. Intell. 40 (6) (2018) 1452–1464.

Xuhong Li is currently a Ph.D. candidate at Université de technologie de Compiègne (UTC), Heudiasyc Laboratory, Compiègne, France. He received the B.S. and M.S. degrees in computer science at Beihang University, Beijing, China. His main research interests include transfer learning, deep learning and computer vision.

Yves Grandvalet is a full researcher at CNRS since 1996. He holds a senior researcher position at Compiègne since 2009, and was previously a visiting researcher at the University of Montréal and at the Idiap research institute, in 2002–2003 and 2006–2008 respectively. He is the author/coauthor of over 100 reviewed papers and book chapters. His current interests include feature selection with sparse methods, model selection, learning from small samples, performance evaluation and fairness.

Franck Davoine received his Ph.D. degree from Grenoble INP, France, in 1995. He was appointed as an Associate Professor at Université de technologie de Compiègne (UTC), Heudiasyc Laboratory, Compiègne, France, in 1997, and as a Researcher at CNRS in 2002. From 2007 to 2014, he was on leave at LIAMA Sino-European Laboratory, Beijing, China, as a PI of a project with CNRS and Peking University, on multisensor-based perception and reasoning for intelligent vehicles. Since 2015, he is back at Université de Technologie de Compiègne, as a PI of the Challenge-Team, within the Laboratory of Excellence MS2T of UTC, focusing on collaborative vehicle perception and urban scene understanding for autonomous driving.