Explicit Inductive Bias for Transfer Learning with Convolutional Networks

Xuhong LI [1] **Yves GRANDVALET** [1] **Franck DAVOINE** [1]

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, and we eventually recommend a simple L^2 penalty with the pre-trained model being a reference as the baseline of penalty for transfer learning tasks.

1. Introduction

It is now well known that modern convolutional neural networks (e.g. Krizhevsky et al. 2012, Simonyan & Zisserman 2015, He et al. 2016, Szegedy et al. 2016) can achieve remarkable performance on large-scale image databases, e.g. ImageNet (Deng et al. 2009) and Places 365 (Zhou et al. 2017), 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 (Yosinski et al. 2014, Simonyan & Zisserman 2015).

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.

In order to help preserve the knowledge embedded in the initial network, we consider a series of other parameter regularization methods 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. We show that it has noticeable effects in inductive transfer learning scenarios.

This paper copes with the inconsistency that still prevails in transfer learning scenarios, where the model is initialized with some parameters, while the abuse of L^2 regularization encourages departing from these initial values. We thus 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. This type of penalty will be designated with *SP* to recall that they encourage similarity with the *starting point* of the fine-tuning process. We evaluate 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 parameters. Fisher information is also taken into account when we test L^2-*SP* and Group-Lasso-*SP* approaches. 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 solving transfer learning tasks and benchmarking new algorithms.

[1]Sorbonne universités, Université de technologie de Compiègne, CNRS, Heudiasyc, UMR 7253, Compiègne, France. Correspondence to: Xuhong LI <xuhong.li@hds.utc.fr>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

2. Related Work

In this section, we recall the approaches to inductive transfer learning in convolutional networks. We focus on approaches that also encourage similarity (of features or parameters) on different models. Our proposal departs either by the goal pursued or by the type of model used.

2.1. Shrinking Toward Chosen Parameters

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. Lehmann & Casella 1998, chapter 5), where it can be related to empirical Bayes arguments. In transfer learning, it has been used in maximum entropy models (Chelba & Acero 2006) or SVM (Yang et al. 2007, Aytar & Zisserman 2011, Tommasi et al. 2014). These approaches were shown to outperform standard L^2 regularization with limited labeled data in the target task (Aytar & Zisserman 2011, Tommasi et al. 2014).

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 usually 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 parameters having a very different role.

2.2. Transfer Learning for Deep Networks

Regarding transfer learning, we follow here the nomenclature of Pan & Yang (2010), 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 & 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. Note that, though we refer here to problems that were formalized or popularized after (Pan & Yang 2010), such as lifelong learning, Pan & Yang's typology remains valid.

2.2.1. Representation Transfer

Donahue et al. (2014) repurposed features extracted from different layers of the pre-trained AlexNet of Krizhevsky et al. (2012) and plugged them into an SVM or a logistic regression classifier. This approach outperformed the state of the art of that time on the Caltech-101 database (Fei-Fei et al. 2006). Later, Yosinski et al. (2014) showed that fine-tuning the whole AlexNet resulted in better performance than using the network as a static feature extractor. Fine-tuning pre-trained VGG (Simonyan & Zisserman 2015) on the image classification task of VOC-2012 (Everingham et al. 2010) and Caltech 256 (Griffin et al. 2007) achieved the best results of that time.

Ge & Yu (2017) 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 jointly fine-tuned a pre-trained convolutional network. Besides image classification, many procedures for object detection (Girshick et al. 2014, Redmon et al. 2016, Ren et al. 2015) and image segmentation (Long et al. 2015a, Chen et al. 2017, Zhao et al. 2017) 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. (2014) 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 (Zeiler & Fergus 2014). Overall, the learned representations can be conveyed to related but different domains and the parameters in the network are reusable for different tasks.

2.2.2. Regularizers in Related Learning Setups

In lifelong learning (Thrun & Mitchell 1995, Pentina & Lampert 2015), where 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 a good performance on all tasks, Li & Hoiem (2017) proposed to use the outputs of the target examples, computed by the original network on the source task, to de-

fine 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 they did not obtain interesting results.

Kirkpatrick et al. (2017) 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 parameter space that 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.

In domain adaptation (Long et al. 2015b), where the target domain differs from the source domain whereas 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. Rozantsev et al. (2016) proposed a parameter regularization scheme for encouraging the similarity of the representations of the source and the target domains. Their regularizer encourages similar source and target parameters, up to a linear transformation. Still in domain adaptation, besides vision, encouraging similar parameters in deep networks has been proposed in speaker adaptation problems (Liao 2013, Ochiai et al. 2014) and neural machine translation (Barone et al. 2017), where it proved to be helpful.

The L^2-*SP* regularizer was used independently by Grachten & Chacón (2017) for transfer in vision application, but where they used a random reinitialization of parameters. For convex optimization problems, this is equivalent to fine-tuning with L^2-*SP*, but we are obviously not in that situation. Grachten & Chacón (2017) conclude that their strategy behaves similarly to learning from scratch. We will show that using the starting point as an initialization of the fine-tuning process *and* as the reference in the regularizer improves results consistently upon the standard fine-tuning process.

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 $\tilde{J}$ 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 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:

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

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:

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

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 $\boldsymbol{w}_S$, the other one for the novel part, denoted $\boldsymbol{w}_{\bar{S}}$.

The compound penalty then becomes:

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

L^2-SP-Fisher Elastic weight consolidation (Kirkpatrick et al. 2017) 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 $\boldsymbol{w}^0_{\mathcal{S}}$. More precisely, it relies on the diagonal of the Fisher information matrix, resulting in the following penalty:

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

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 $\boldsymbol{x}$ 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*:

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

The usual L^1 penalty encourages sparsity; here, by using $\boldsymbol{w}^0_{\mathcal{S}}$ 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 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:

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

where $\boldsymbol{w}^0_{\mathcal{G}_0} = \boldsymbol{w}^0_{\bar{\mathcal{S}}} \stackrel{\triangle}{=} \mathbf{0}$, 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:

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

4. Experiments

We evaluate the aforementioned parameter regularizers on several pairs of source and target tasks. We use ResNet (He et al. 2016) as our base network, since it has proven its wide applicability on transfer learning tasks. Conventionally, if the target task is also a classification task, the training process starts by replacing the last layer with a new one, randomly generated, whose size depends on the number of classes in the target task.

4.1. 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 (Deng et al. 2009) for generic object recognition and Places 365 (Zhou et al. 2017) for scene classification. Likewise, we have three different databases related to three target problems: Caltech 256 (Griffin et al. 2007) contains different objects for generic object recognition; MIT Indoors 67 (Quattoni & Torralba 2009) consists of 67 indoor scene categories; Stanford Dogs 120 (Khosla et al. 2011) contains images of 120 breeds of dogs; Each target database is split into training and testing sets following the suggestion of their creators (see Table 1 for details). 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 test.

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
MIT Indoors 67	scene classification	80	20	67
Stanford Dogs 120	specific object recog.	100	~ 72	120
Caltech 256 – 30	generic object recog.	30	20	257
Caltech 256 – 60	generic object recog.	60	20	257

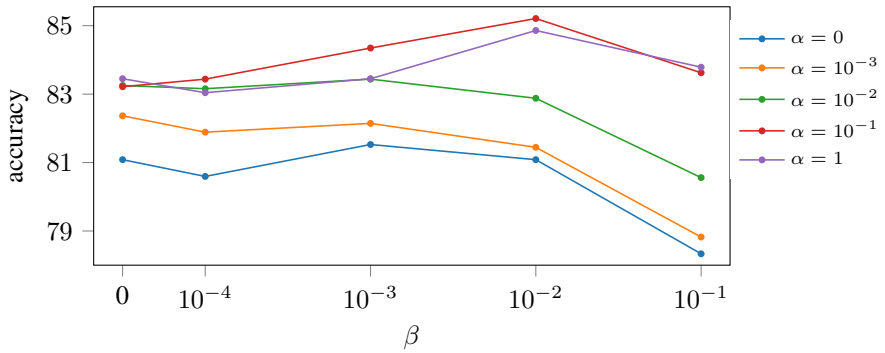

Figure 1. Classification accuracy (in %) on Stanford Dogs 120 for L^2-*SP*, according to the two regularization hyperparameters α and β respectively applied to the layers inherited from the source task and the last classification layer (see Equation 3).

4.2. 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 searching 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. Figure 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 (Abadi et al. 2015).

4.3. Results

4.3.1. 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. 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. We can observe that L^2-*SP* and L^2-*SP-Fisher* always improve over L^2, and that when less training data are available for the target problem, the improvement of L^2-*SP* and L^2-*SP-Fisher* compared to L^2 are more important. Meanwhile, no large difference is observed between L^2-*SP* and L^2-*SP-Fisher*.

We can boost the performance and outperform the state of the art (Ge & Yu 2017) in some cases by exploiting more training techniques and post-processing methods, which are described in the supplementary material.

4.3.2. Behavior Across Penalties, Source and Target Databases

A comprehensive view of our experimental results is given in Figure 2. Each plot corresponds to one of the four target databases listed in Table 1. The light red points mark the accuracies of transfer learning when using Places 365 as the source database, whereas the dark blue points correspond to the results obtained with ImageNet. As expected, the results

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 Stanford Dogs 120 and Caltech 256.

	MIT Indoors 67	Stanford Dogs 120	Caltech 256 – 30	Caltech 256 – 60
L^2	79.6±0.5	81.4±0.2	81.5±0.2	85.3±0.2
L^2-*SP*	**84.2±0.3**	**85.1±0.2**	**83.5±0.1**	**86.4±0.2**
L^2-*SP-Fisher*	84.0±0.4	**85.1±0.2**	83.3±0.1	86.0±0.1

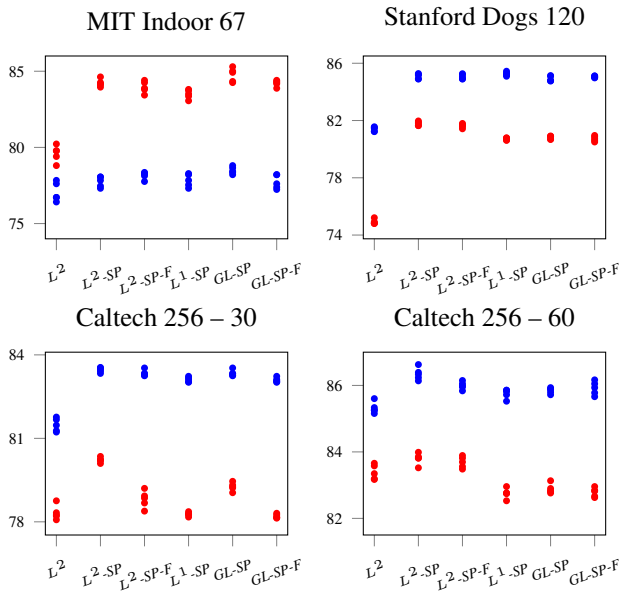

Figure 2. Classification accuracies (in %) of the tested fine-tuning approaches on the four target databases, using ImageNet (dark blue dots) or Places 365 (light red dots) as source databases. MIT Indoor 67 is more similar to Places 365 than to ImageNet; Stanford Dogs 120 and Caltech 256 are more similar to ImageNet than to Places 365.

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. It is however interesting to note that the trends are similar 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.

This benefit 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 – 60 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 of Duchi & Singer (2009), 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.

Table 3. Classification accuracy drops (in %) 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 Stanford Dogs 120 and Caltech 256. The classification accuracies of the pre-trained models are 54.7% and 76.7% on Places 365 and ImageNet respectively.

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

Finally, the variants using the Fisher information matrix behave like the simpler variants using a Euclidean metric on parameters. We believe that this is due to the fact that, contrary to lifelong learning, our objective does not favor solutions that retain accuracy on the source task. The metric defined by the Fisher information matrix may thus be less relevant for our actual objective that only relates to the target task. Table 3 confirms that L^2-*SP-Fisher* is indeed a better approach in the situation of lifelong learning, where accuracies on the source tasks matter. It 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 smaller for L^2-*SP-Fisher* than for L^2-*SP*. In comparison, L^2 fine-tuning results in catastrophic forgetting: the performance on the source task is considerably affected by fine-tuning.

4.3.3. Fine-Tuning *vs.* Freezing the Network

Freezing the first layers of a network during transfer learning is another way to ensure a very strong inductive bias,

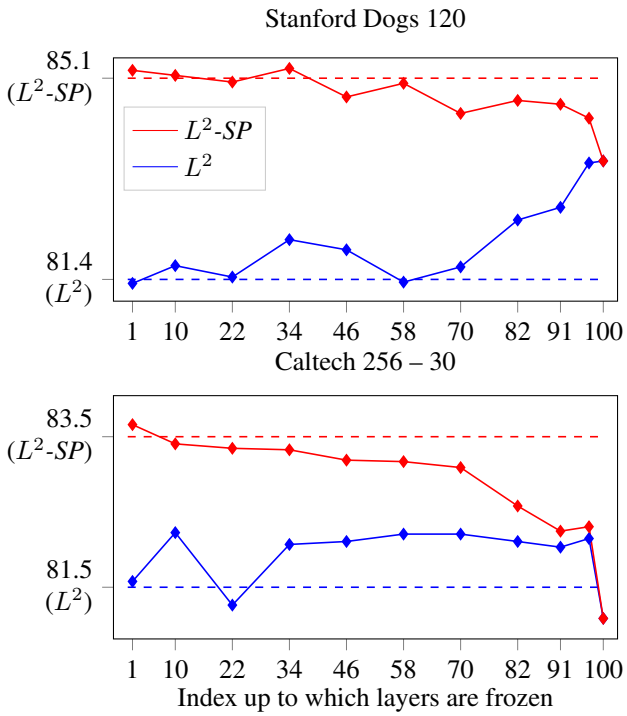

Figure 3. Classification accuracies (in %) of fine-tuning with L^2 and L^2-*SP* on Stanford Dogs 120 (top) and Caltech 256–30 (bottom) when freezing the first layers of ResNet-101. The dashed lines represent the accuracies reported in Table 2, where no layers are frozen. ResNet-101 begins with one convolutional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

letting less degrees of freedom to transfer learning. Figure 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 a rather inefficient strategy 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*.

4.4. Analysis and Discussion

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 helps in lifelong learning, but there is no significant difference between the two options. 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 analysis and the discussion would also apply to L^2-*SP-Fisher*.

4.4.1. 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.

4.4.2. Theoretical Insights

Analytical results are very difficult to obtain in the deep learning framework. Under some (highly) simplifying assumptions, we show in supplementary material that the optimum of the regularized objective function with L^2-*SP* is a compromise between the optimum of the unregularized objective function and the pre-trained parameter vector, 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 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.

Using L^2-*SP* instead of L^2 can also be motivated by an analogy with shrinkage estimation (see e.g. Lehmann & Casella 1998, 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.

4.4.3. 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, and 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

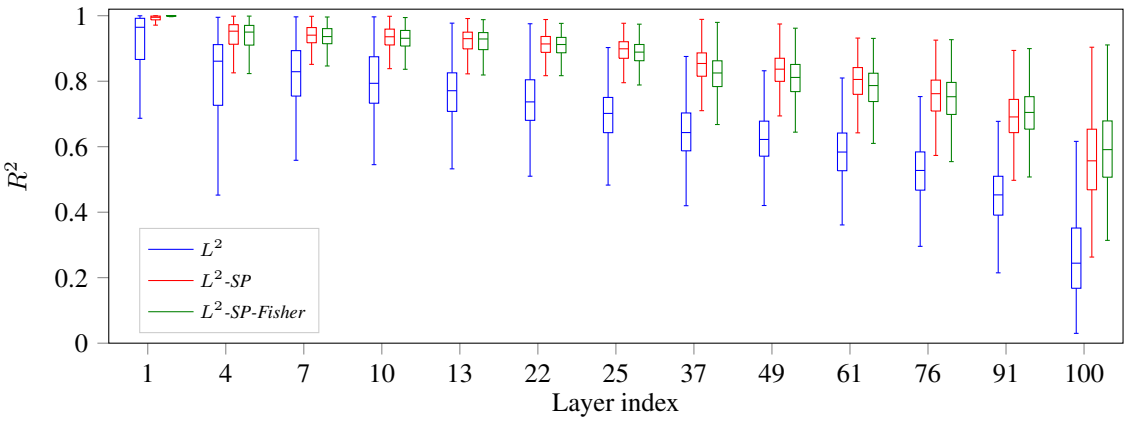

Figure 4. R^2 coefficients of determination with L^2 and L^2-*SP* regularizations for Stanford Dogs 120. Each boxplot summarizes the distribution of the R^2 coefficients of the activations after fine-tuning with respect to the activations of the pre-trained network, for all the units in one layer. ResNet-101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

activations throughout the network is displayed in Figure 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 details how the roles of the network units is 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.

5. Conclusion

We described and tested simple regularization techniques for inductive transfer learning. They all encode an explicit bias towards the solution learned on the source task, resulting in a compromise with 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 or Group-L^1 norms, or based on 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* does not improve accuracy on the target task. Different approaches, which implement an implicit bias at the functional level, alike Li & Hoiem (2017), 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. We acknowledge the support of NVIDIA Corporation with the donation of GPUs used for this research.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M.,

Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.

Aytar, Y. and Zisserman, A. Tabula rasa: Model transfer for object category detection. In *ICCV*, 2011.

Barone, A. V. M., Haddow, B., Germann, U., and Sennrich, R. Regularization techniques for fine-tuning in neural machine translation. *arXiv preprint arXiv:1707.09920*, 2017.

Chelba, C. and Acero, A. Adaptation of maximum entropy capitalizer: Little data can help a lot. *Computer Speech & Language*, 20(4):382–399, 2006.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

Duchi, J. and Singer, Y. Efficient online and batch learning using forward backward splitting. *J. Mach. Learn. Res.*, 10:2899–2934, 2009.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88 (2):303–338, 2010.

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

Ge, W. and Yu, Y. Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. In *CVPR*, 2017.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

Grachten, M. and Chacón, C. E. C. Strategies for conceptual change in convolutional neural networks. *arXiv preprint arXiv:1711.01634*, 2017.

Griffin, G., Holub, A., and Perona, P. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Khosla, A., Jayadevaprakash, N., Yao, B., and Li, F.-F. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, 2011.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. U.S.A.*, 2017.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

Lehmann, E. L. and Casella, G. *Theory of point estimation*. Springer, 2 edition, 1998.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017. doi: 10.1109/TPAMI.2017.2773081.

Liao, H. Speaker adaptation of context dependent deep neural networks. In *ICASSP*, 2013.

Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015a.

Long, M., Cao, Y., Wang, J., and Jordan, M. Learning transferable features with deep adaptation networks. In *ICML*, 2015b.

Ochiai, T., Matsuda, S., Lu, X., Hori, C., and Katagiri, S. Speaker adaptive training using deep neural networks. In *ICASSP*, 2014.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE T. Knowl. Data En.*, 22(10):1345–1359, 2010.

Pentina, A. and Lampert, C. H. Lifelong learning with non-iid tasks. In *NIPS*, 2015.

Quattoni, A. and Torralba, A. Recognizing indoor scenes. In *CVPR*, 2009.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

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

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

Thrun, S. and Mitchell, T. M. Lifelong robot learning. *Robot. Auton. Syst.*, 15(1-2):25–46, 1995.

Tommasi, T., Orabona, F., and Caputo, B. Learning categories from few examples with multi model knowledge transfer. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2014.

Yang, J., Yan, R., and Hauptmann, A. G. Adapting svm classifiers to data with shifted distributions. In *ICDM Workshops*, 2007.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? In *NIPS*, 2014.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. Pyramid scene parsing network. In *CVPR*, 2017.

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A. Places: A 10 million image database for scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017.