Transfer Learning & Semi-Supervised Learning Methods

Hugo TEIXEIRA DE CASTRO Younes OUMHAMED
Jon H. SIGURDSSON

May 2023

Abstract

This report explores the application of transfer learning and semi-supervised learning techniques to improve the performance of a pretrained convolutional model on the Oxford-IIIT Pet dataset. The first part focuses on fine-tuning a Resnet50 model for binary and multi-class classification tasks. Various fine-tuning steps, such as updating specific layers, batch normalization, data augmentation, and learning rate schedule, are applied and evaluated. The results show that these modifications lead to significant improvements in accuracy, with the best-performing model achieving 95% accuracy.

The second part of the project investigates two semi-supervised learning methods: Consistency Regularization and Pseudo-Labeling. These methods are implemented and evaluated on the PET dataset with varying percentages of labeled data (1%, 10%, and 50%). The results demonstrate that both methods contribute to higher accuracy compared to the fully supervised baseline model. Consistency Regularization, specifically the Mean Teacher approach, helps improve model consistency and prevent overfitting.

Overall, this report provides insights into the effectiveness of transfer learning and semi-supervised learning techniques in improving the performance of convolutional models on challenging datasets like Oxford-IIIT Pet.

1 Introduction

The goal of this project was to explore the possibilities of transfer learning by fine-tuning a pre-trained convolution model and making it perform on a different dataset. The idea is that, during training, we only update the parameters of a few specific layers, whereas the rest of the pre-trained weights remain frozen. According to many papers [1], this approach yields better results than training a model from scratch and also reduces the training cost.

In the first part of the project, we decided to apply this method to the Oxford PET dataset, with both a binary classification (distinguishing between images of cats and dogs) and a multi-classification problem (identifying the breed). To do this, we used Resnet50 as our base and applied several fine-tuning steps to improve performance.

The second part was dedicated to semi-supervised learning. We wanted to see what techniques existed to train a model with missing labels, and which one was most efficient. We only implemented 2 different methods, Consistency Regularization, and Pseudo-Labeling, and ran them on the PET dataset which we removed more and more labels.

2 Related Work

Transfer Learning is a widely used method in Deep Learning applications. In a context where finding appropriate and sufficiently large datasets is becoming harder and harder, transfer learning appeared as a convenient solution [2][12].In many cases, this approach actually yields better results than fine-tuning a model from scratch[1]. The most difficult part is to properly fine-tune the model to make it perform on a specific dataset. Many strategies, such as selecting the appropriate layers/blocks to fine-tune, have been explored [7].

The area of semi-supervised learning is also very promising, especially since it can be difficult to find labeled examples in specific domains. There exists a huge variety of options, but we only focused on two of them: Consistency Regularization and Pseudo Labeling. The former tries to minimize the difference between the initial predictions and slightly augmented ones. Different approaches (Π model [9], Temporal Ensembling [4], Mean Teacher[10]) apply different noise transformations and optimizations.

The latter goes through multiple training phases. First, we train only on the labeled datasets and use the resulting model to predict labels for the missing ones. We then integrate the labels we predicted into the training set and repeat until all data has been labeled [6].

3 Data

For all our experiments, we used the Oxford-IIIT Pet dataset, which is a collection of images representing cats and dogs. It is provided by Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar (available at `https://www.robots.ox.ac.uk/~vgg/data/pets/`). It contains 2371 images for cats and 4978 for dogs. It has labels for the species (cat or dog), but also for each different breed (37 in total, with around 200 images for each). The dataset can thus be used both for binary and multi-classification. Each image has a different size and scale, so as preprocessing we resized all of them to 224*224 px and also normalized the RGB values. We split the training and validation set in an 80/20 ratio, and as there are almost twice as many dogs as cats, we distributed them evenly by first splitting separately the cats and dogs 80/20 and then concatenating them. The Oxford-IIIT dataset is widely used in the literature, and the best results go as high as 96% [8].

4 Methods

4.1 Fine-tuning Resnet

In the first part of the project, we tried to add some fine-tuning to the pre-trained Resnet model to increase the validation accuracy on the PET dataset. For the binary classification, we already went beyond 99% without any modification. For the following, we will thus only focus on the multi-classification problem.

We started with pre-trained Resnet50, to which we replace the last layer with a 224*224 FC. We performed training and updated only the parameters of that last layer, which would correspond to the minimum amount of fine-tuning. After this, we separately applied different fine-tuning steps, which we ultimately combined to try and improve the performance. More precisely, we performed the following experiments:

- Fine-tuning the parameters of the last one, two, and three layers of Resnet50 during training.
- Fine-tuning the parameters of all the batch normalization
- Augmenting the initial dataset with image transformations (flips, perspective, crops, Gaussian blurs)
- Using higher learning rates for the last layer (0.002 for the last but 0.001 for the others) as well as a Linear Scheduler starting from 100 to 30% of the initial LR.

We then compared the validation accuracy obtained after each of these methods with the base model to see how much it improved performance.

4.2 Consistency Regularization

There are many different approaches to consistency regularization. Their common goal is, however, to make the network more invariant to small changes whether it is within the network, e.g. dropout, or data, e.g. data augmentation. Model predictions should be consistent. To compare these methods, we trained the model on the PET dataset again but this time changing the percentage of the labeled dataset (3 experiments: 50%, 10 %, and 1%) and then measuring the validation accuracy. We do this for the 2 semi-supervised techniques we implemented (cf next sections) as well as for the fully fine-tuned model we built for the basic assignment to serve as the baseline.

4.2.1 Mean teacher

The first method used in this assignment is called *mean teacher* [11] and was proposed as an improvement of the previously introduced *temporal ensembling* [5]. The mean teacher method works with two different models, a student, and a teacher. The student model weights are updated identically to the ones in the basic assignment network but based on a loss function that contains supervised and unsupervised loss terms. The supervised loss term is a classification loss function identical to the one previously used.

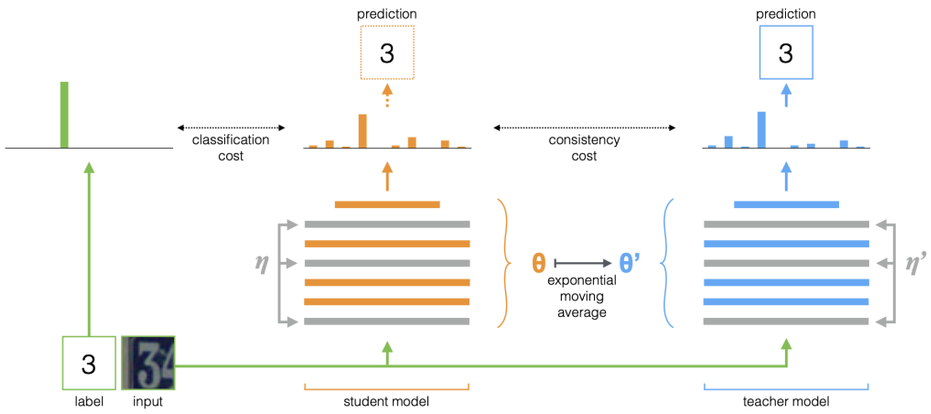


Figure 1: Training process for student and teacher models [11]

The unsupervised loss term is, however, a consistency loss function that computes the MSE between the student and teacher softmax outputs. The inputs for the student and teacher are introduced with different Gaussian blurs. By feeding the student and teacher models slightly different inputs and penalizing based on the difference in outputs, the student model is 'encouraged' to be consistent in its predictions and prevented from overfitting to the labeled data. In the case of an unlabelled data sample, the loss function only contains the consistency loss. The process of training is explained in figure **??**.

The loss function is only used to update the student model weights whereas the teacher weights are an exponential moving average of the student weights (equation **??**).

$$\theta_t^{'} = \alpha\theta_{t-1}^{'} + (1-\alpha)\theta_t \tag{1}$$

where $\theta_t^{'}$ are the teacher weights at time step t and θ_t are the student model weights at time step t. The hyperparameter α was set to 0.99 during the first 10 epochs, then raised to 0.999, as suggested by the original paper.

4.2.2 Pseudo-Labeling

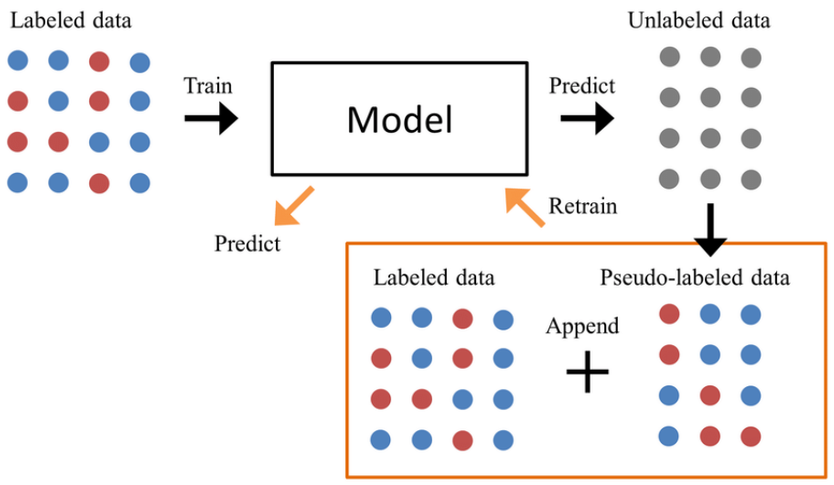


Figure 2: Training process for Pseudo-Labels [3]

To cope with fewer labeled training examples, we opted for a second approach called Pseudo-Labeling. This naive approach first proposed by Lee in 2013 [6] consists of training the network with labeled and unlabeled examples simultaneously in each batch. The loss function is therefore equal to:

$$L = \frac{1}{n}\sum_{m=1}^{n}\sum_{i=1}^{37} L(y_i^m + f_i^m) + \alpha(t)\ \frac{1}{n'}\sum_{m=1}^{n'}\sum_{i=1}^{37} L(y_i'^m + f_i'^m) \tag{2}$$

Where f_i^m is the output units of m's sample in labeled data, y_i^m is the label of that,$f_i'^m$ unlabeled data, $y_i'^m$ is the pseudo-label of that for unlabeled data. The hyperparameter

$\alpha(t)$ is a coefficient to control the contribution of the unlabeled examples in our model. According to [6], this coefficient is defined as follow:

$$\alpha(t) = \begin{cases} 0 & \text{if } t < 100 \\ \frac{t-100}{600-100} * 3 & \text{if } 100 < t < 600 \\ 3 & \text{if } t > 60 \end{cases} \tag{3}$$

In simpler words, the loss function could be seen as an addition of the loss in labeled data and the loss in the unlabeled data with a weight to control the contribution of unlabeled data to the overall loss.

$$Loss\ per\ batch = Labeled\ loss + Weight * Unlabeled\ loss$$

5 Results

5.1 Basic assignment

The main part of the basic assignment was concentrated on exploring solutions in an attempt to further increase the accuracy of the multi-class classification problem. The different improvements were applied separately and compared to the original version of our network for comparison. Finally, all the modifications were combined.

Down below you can find the evolution of the validation accuracy over the 15 epochs. Each curve represents a specific fine-tuning method.

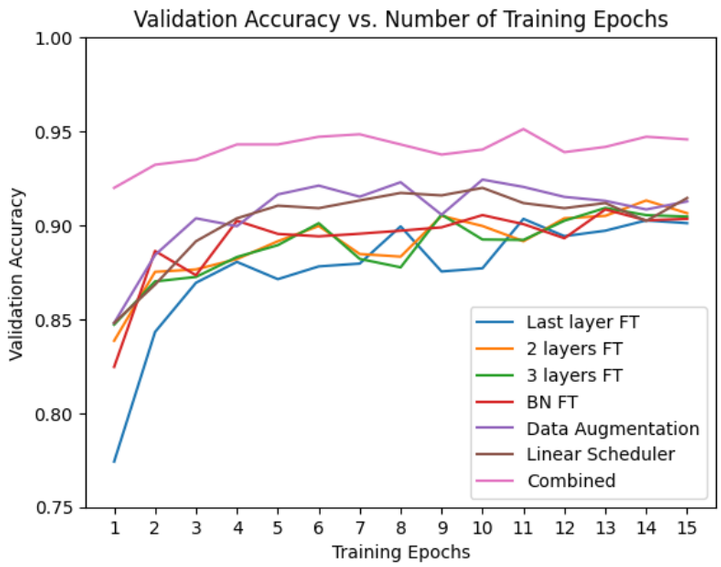


Figure 3: Evolution of the validation accuracy for each fine-tuning procedure

We also show the best performance obtained in each configuration:

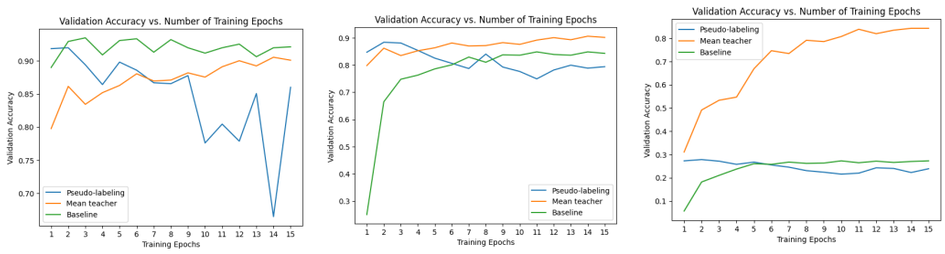
1 layer FT	2 layers FT	3 layers FT	BN FT	Data Augmentation	Scheduler	All combined
0.903	0.913	0.909	0.914	0.924	0.908	0.951

Table 1: Best validation accuracies for the different fine-tuning procedures

We observe that no matter which fine-tuning method we choose, we ultimately perform better than the base model. All the contributions, though slight, add up and allow us to go from 90% to 95% accuracy, which is significant. Overall, the biggest improvements were brought by data augmentation and by learning rate schedulers. We also notice that there is little difference between fine-tuning 2 or 3 layers, which shows that there is a limit to the number of layers to train to get better performance (this is why we only fine-tuned 2 layers in the end). In the extreme case where we train the model from scratch, we know that the accuracy drops significantly.

5.2 Semi-supervised learning

The different characteristics of the two semi-supervised learning methods mentioned previously were then explored and compared to the final network and learning methods from the basic assignment. As a clear evaluation of the impact of these semi-supervised learning methods, the network from the basic assignment was fed the same number of labeled samples as the semi-supervised networks whereas those networks were additionally fed unlabelled data samples.



(a) 50% labels (b) 10% labels (c) 1% labels

Figure 4: Evolution of the validation accuracy for the different Semi-supervised learning implementations

	50%	10%	1%
Baseline	93.49%	84.80%	27.27%
Pseudo labeling	92.00%	88.33%	27.82%
Mean teacher	90.56%	87.23%	42.06%

Table 2: Best validation accuracies with different percentages of labeled data

We can see that for a relatively high percentage of labeled data (50 or 10%), our baseline performs as well or sometimes even better than the models that implement semi-supervised learning. It can be explained by the fact that the model is already pre-trained, so it's less sensitive to drops in the amount of input data. Besides, we already greatly augmented the PET dataset during the fine-tuning phase.

However, in a case with very few labeled examples (1%), the base model cannot maintain the same performance, whereas our Mean teacher model manages to remain above 40% accuracy, making it a significant improvement (we had a slight bug in the previous implementation resulting in an ∼80% accuracy as can be seen in figure 4 (mentioned in the presentation). The revised learning curve can be seen in figure 5). The fact that the loss function can be applied regardless of the labels definitely helps the training phase. We can conclude that semi-supervised learning is perhaps unnecessary when there is a large number of labels available, but that it can make the model much more robust in a context with very few annotated data samples.

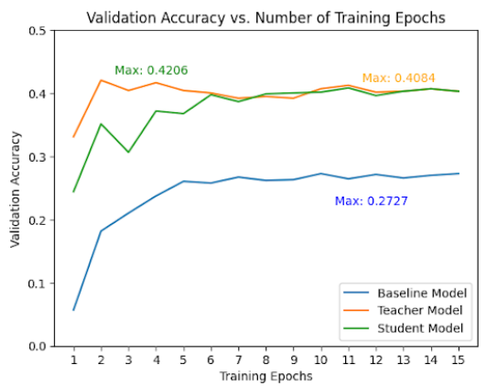


Figure 5: The revised learning curve for 1% of labels with mean teacher

When it comes to Pseudo-Labeling, an important finding is that the accuracy tends to decrease towards the end of the training process. This decline can be explained by the

amplification of the weights that determine the impact of unlabeled examples. Since the initial accuracy is quite low, it is likely that a significant portion of the pseudo-labels is incorrect. Consequently, training the model using these erroneous examples with false labels leads to even poorer accuracy compared to training without pseudo-labeling.

Moreover, it is important to highlight that Pseudo-Labeling does not outperform the baseline model when 50% of the data is labeled. This is not surprising since having half of the data labeled is usually sufficient to achieve high accuracy with the baseline model. However, Pseudo-Labeling demonstrates significant improvements in cases where the labeled data is extremely limited. For instance, when only 10% or 1% of the data is labeled, the test accuracy of Pseudo-Labeling surpasses that of the baseline model. This indicates that Pseudo-Labeling effectively enhances model performance when dealing with very few labeled examples.

6 Conclusion

In this project, we explored transfer learning by fine-tuning a pre-trained convolutional model on the Oxford-IIIT Pet dataset. We achieved significant improvements in accuracy by employing various fine-tuning techniques, such as updating the last layers, batch normalization, data augmentation, and learning rate scheduling.

We also investigated semi-supervised learning methods, including Consistency Regularization and Pseudo-Labeling. These approaches improved the model's performance when labeled data was limited. Both demonstrated potential in leveraging unlabeled data for better results. Fine-tuning techniques and semi-supervised learning methods offer promising avenues for achieving high accuracy with limited labeled data. Further research can explore different fine-tuning strategies, architectures, and semi-supervised techniques for real-world applications.

References

[1] Aidan Boyd, Adam Czajka, and Kevin Bowyer. Deep learning-based feature extraction in iris recognition: Use existing models, fine-tune or train from scratch? In *2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–9, 2019.

[2] Benjamin Q Huynh, Hui Li, and Maryellen L Giger. Digital mammographic tumor classification using transfer learning from deep convolutional neural networks. *Journal of Medical Imaging*, 3(3):034501–034501, 2016.

[3] Sung Kim, Young Lee, Bayu Adhi Tama, and Seungchul Lee. Reliability-enhanced camera lens module classification using semi-supervised regression method. *Applied Sciences*, 10:3832, 05 2020.

[4] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *CoRR*, abs/1610.02242, 2016.

[5] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning, 2017.

[6] Dong-Hyun Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

[7] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015.

[8] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[9] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *CoRR*, abs/1606.04586, 2016.

[10] Antti Tarvainen and Harri Valpola. Weight-averaged consistency targets improve semi-supervised deep learning results. *CoRR*, abs/1703.01780, 2017.

[11] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, 2018.

[12] Grega Vrbancic, Milan Zorman, and Vili Podgorelec. Transfer learning tuning utilizing grey wolf optimizer for identification of brain hemorrhage from head ct images. In *StuCoSReC: proceedings of the 2019 6th student computer science research conference*, pages 61–66, 2019.

7 Appendix

Here is the link to the Google Colab we used for the project: `https://colab.research.google.com/drive/1av-HOgdaAnG1KaE1ezb3ggFoIMRAu2wd#scrollTo=8X8mLOfXseb5`