Transfer Learning for Remote Sensing: Enhancing EuroSAT Classification with Limited Data

Mitun Paul
University of Oulu
Oulu, Finland
mpaul23@student.oulu.fi

Akash Shingha Bappy
University of Oulu
Oulu, Finland
abappy23@student.oulu.fi

H M Tanvir Shuvo
University of Oulu
Oulu, Finland
tshuvo23@student.oulu.fi

Abstract

The report focuses on enhancing deep learning models using transfer learning for the EuroSAT data set. It delves into the methodology involving pre-training, fine-tuning, and deployment phases. The study highlights the advantages of transfer learning in scenarios with limited data. It conducts a comparative analysis of different models and hyper-parameters to optimize performance, demonstrating the effectiveness of transfer learning in improving model accuracy for specific tasks with small data-sets.

1. Introduction

Deep learning has emerged as the preferred option for numerous applications. Deep learning is a type of algorithm for machine learning designed to learn at multiple levels, each of which relates to a distinct degree of abstraction [3]. Although strong, deep learning has its own limitations. Both Shalev-Shwartz (2017) [12] and Abbe (2018) [1] illustrate circumstances in which fundamental deep-learning approaches fall short, especially when confronted with adversarial attacks and while learning particular efficiently learnable functions. These attacks exploit vulnerabilities in the training phase, making it difficult to defend against them [10]. Waldrop (2019) [15] emphasizes these limits by discussing deep learning's vulnerability to spoofing and inefficiency.

Transfer learning solves some of the problems deep learning has when data is scarce. Transfer learning has emerged as a pivotal paradigm in the rapidly evolving landscape of machine learning, allowing models to leverage knowledge acquired in one task for improved performance in a different, yet related, domain [9], illustrated on figure 1. Transfer learning is a beneficial strategy when dealing involving scenarios with small sample sizes because it allows knowledge to be repurposed from one domain to another, minimizing the requirement for huge amounts of training data [16] [17].This is especially beneficial in deep learning models, where millions of parameters necessitate a large amount of data [17]. Transfer learning can also be used in network modeling, where it beats single-task learning when it comes to identifying networks for related tasks [8]. A hybrid multi-task/transfer learning model has been suggested in the medical industry to improve classification with limited sample sets, displaying higher performance over existing methods [11]. In short, compared to traditional deep learning, transfer learning can handle discrepancies between domains and distributions, making it more versatile [4].

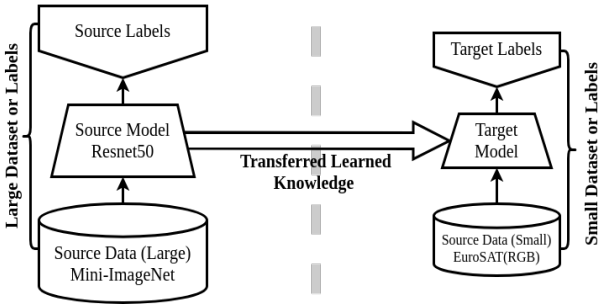

Figure 1. Illustration of Transfer Learning Model

This report mainly focuses on three parts. Namely:

- Pre-Training: This is the initial phase which involves training a model on a large data set from a source domain. The model learns general features and patterns from this diverse data set.

- Fine-tuning: In this phase, pre-trained model is adapted to the specific target domain or task by fine-tuning on a data-set which is smaller related to the new task. Lower layers that are responsible for general features are often frozen to preserve the learned representations, while higher layers are adjusted to task-specific data.

- Deployment: After fine-tuning, the model is ready for inference on new, unseen data in the target domain. The adapted model leverages both general knowledge from the source domain and specific insights gained during fine-tuning to make predictions or classifications in the target domain.

Finally, the contribution of each author is given below:

- First author was responsible for pre-training phase where he with the help of other authors developed the code and pre-trained models using the Mini-ImageNet data set. First author also helped in the report drafting process.
- The Second author was responsible for the Fine-tuning phase, where he, with the help of other authors fine, fine-tuned the model hyper-parameters. The second author also helped in the report drafting process.
- The Third author was responsible for the Deployment phase, where he used the EuroSat data set for the final transfer learning phase. The third author was also responsible for the report drafting process and the final Latex finalization of the report.

2. Approach

2.1. Data set Description

The Mini-ImageNet data set, which was developed by Vinyals et al. in 2016 [14], is a subset of the larger ImageNet data set designed specifically for few-shot learning evaluation. It consists of 60,000 color images with a resolution of 84x84 pixels. These images are divided into three splits: 64 training classes, 16 validation classes, and 20 testing classes, containing 600 examples for each class.

The EuroSat data set, developed by Helber et al. 2018-2019 [6] [7], serves as a benchmark for land use as well as land cover classification. It is based on Sentinel-2 satellite images, covering 13 spectral bands. The data set contains 27,000 labeled and geo-referenced images distributed across 10 different classes. EuroSat offers two versions: one containing nothing but the optical frequency bands, i.e., R, G, and B and encoded as JPEG images (RGB), and another that includes all 13 bands in their original value range.

2.2. Data Preparation

For transfer learning the first step is always setting up the dataset. The datasets we are using for this paper are Mini-ImageNet and EuroSat-RGB. For initial training, we used the Mini-ImageNet dataset for pretraining the proposed model. The original dataset had a total of 100 classes with 600 images each of 84x84 resolution and was split to 64, 16, and 20 for training, validation, and testing [14]. However, to make the process easier, we have skipped the validation and test sub-dataset and instead focused on only the train data and split it into three more sub-datasets for training, validation, and testing, with a split ratio of 50%, 25%, and 25% respectively.

For the next task, which is training a smaller dataset with a pre-trained model, we used EuroSAT(RGB) dataset, which has 10 classes and 27,000 images in total [6] [7]. But we have used only 5 classes randomly and 20 images from each class, which is 100 images in total. Among these 25 images were used for training, and the rest 75 samples were used for testing.

2.3. Data Preprocessing

We preprocessed images with various techniques before feeding them into the network and training. We anticipate that the model will perform effectively in classes with more examples since it will be able to learn to map features for labeling with greater efficiency. We have employed augmentation during training to compensate for the limited number of training samples. The purpose of data augmentation is to raise the amount of training images artificially that our model receives by transforming the images randomly. Such as, we can randomly rotate, flip or trim the images vertically or horizontally. In this way, the model becomes more immune to input data changes.

Figure 2. Illustration of data Augmentation

2.4. Model Description

ResNet-50 is a variant of the Residual Network (ResNet) architecture [5] which has 50 layers deep and is a convolutional neural network (CNN). It was designed to solve the problem of vanishing gradients and to improve training performance. The key innovation in ResNet50 is the use of residual blocks. These blocks allow the model to skip over some layers during training, which prevents the gradient from becoming too small (vanishing gradient problem). This architecture enables the model to be deeper without

suffering from performance degradation, which is a common problem in deep neural networks. ResNet50 is widely used in image recognition tasks and has shown excellent performance in large-scale image classification challenges.

VGG16 [13] is a model from the VGG (Visual Geometry Group, University of Oxford) series. It consists of 16 layers, of which 13 are convolutional layers and 3 are fully connected layers at the end. The model is characterized by its use of 3x3 convolutional filters throughout the network and increased depth compared to its predecessors. VGG16 became popular not only for its simplicity but also for its performance in image recognition tasks.

2.5. Model Selection

When the data was ready our next target was to select a good model which is suitable for training. As we are using pytorch we got a lot of options for models. Initially we choose ResNet18 [5], Resnet50 [5], VGG16 [13] and gave some trials which will be mentioned in later part.However, we focused on Resnet50 [5] as our main model to train Mini-ImageNet dataset as it worked well and had a balance between training performance and speed.

2.6. Hyper Parameters Configuration

The performance and accuracy of a model in deep learning depends on the hyper-parameters such as learning rate, batch size, epochs, optimizer, weight decay, dropout rate etc. However, there could be a lot of possible combinations with these variables which makes the process very complicated. To resolve this issue we used Optuna [2] which is an open source automatic hyper-parameter optimization framework. It evaluates the training with different variations of parameters given in a range to get the best output.

2.7. Training Mini-ImageNet

The most crucial and time-consuming step is where the data set is trained. During the training, the main function iterates over numerous epochs, and on each epoch, we iterate through the train Data-Loader. The Data Loader processes one batch of data and targets, which is passed through the model. After each batch of training, the loss is calculated. The gradients of the loss are back-propagated, and the parameters are updated with the optimizer. First, we ran 100 trials with 20 epochs each to find the potential hyper-parameters configurations. This was done several times to get a better result. After a while, when we found a threshold, we stopped the auto-tuning and then continued the training process but instead manually with the chosen hyper-parameters. However, this time we did it with 50 epochs so that a better result can be obtained. The output of this training is shown in Table 1 in the result section. It should be noted that during the training, the pre-trained mode of the model was set to "False." Turning off pre-trained mode for the first model training in transfer learning allows the model to learn from scratch using the specific data set, preventing it from relying too heavily on patterns learned from a different domain. This helps the model adapt better to the nuances of the new task and data set.

2.8. Saving the Model

After the training process, a version of the model is saved keeping its architecture, weights and bias where,

$$Output = Activation(Weight \times Input + Bias) \quad (1)$$

This will make it possible to train the next data set with a greater performance which was the main objective of this report

2.9. Training EuroSAT

The following task was training the EuroSAT data set with the pre-trained model.Here, only 5 classes were randomly chosen, trained for several, and their score were averaged.We also utilized automatic hyper-parameter tuning to get the best configurations. However, unlike the previous step we turned the pretrained mode to "True" and provided a pretrained path to apply the transfer learning. Also, we trained the dataset without transfer learning so that we could observe and compare the differences. The result of the training is shown in **Figure 02**

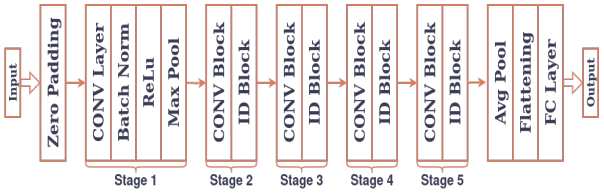

Figure 3. Illustration of the Transfer Learning Model layers

3. Results and Discussion

We have tried different models such as ResNet 50, ResNet 18, VGG on Mini ImageNet data set with default model initialization without any pre-trained initialization. We have tried to search for the best hyper-parameters with Optuna with 20 epochs. From the trials, we got the best values, which were used to to train our model for 50 epochs. We collected Accuracy on the test data set of those models given and hyper-parameters related to those models. We found ResNet50 gives above 71% accuracy on test data set which is better among other models we tried. The results can be observed on the table 1.

Figure 4 illustrates the hyper-parameter optimization process for each data set. As we implemented optuna library for the tuning we did not have to change the hyper-parameters values manually but rather provided a range.

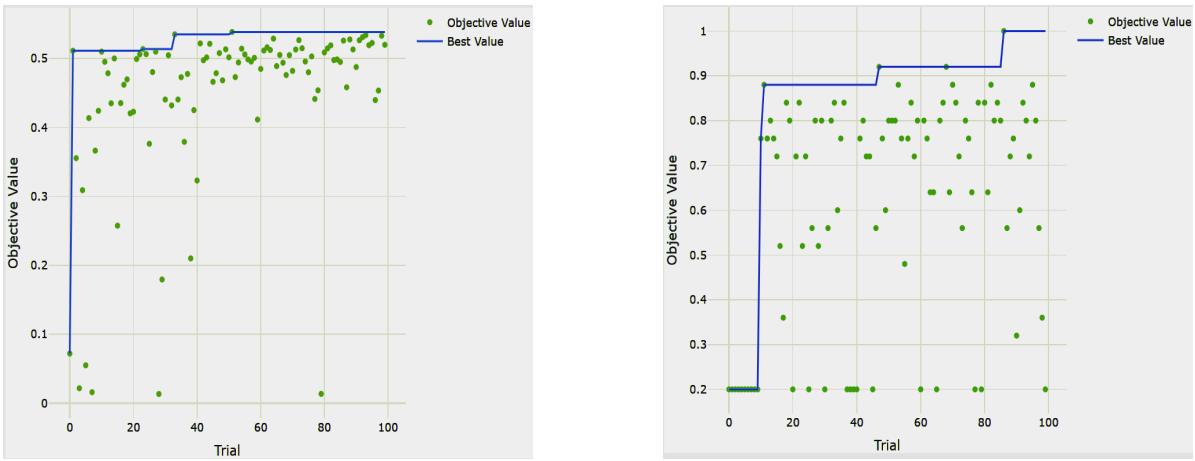

Figure 4. Figure 4 illustrates automatic hyper-parameter optimization using Optuna for Mini-ImageNet (left) and EuroSAT (right) data set for ResNet50

Model	Optimizer	Batch Size	Learning Rate	Dropout Rate	Weight Decay	Momentum	Accuracy (Test)
ResNet50	RMSProp	32	4.24E-05	0.43548	8.51E-05	0.48492	0.717502
ResNet50	RMSProp	16	4.24E-05	0.43548	8.51E-05	0.48491	0.711615
ResNet18	RMSProp	128	4.66E-05	0.48	2.01E-05	0.93876	0.712906
ResNet18	RMSProp	32	7.22E-05	0.30413	1.40E-05	0.69849	0.712792
ResNet18	RMSProp	128	1.60E-04	0.26769	3.85E-05	0.75918	0.714502
ResNet18	RMSProp	32	4.97E-05	0.35106	2.15E-05	0.51789	0.702708
ResNet18	Adam	128	2.17E-04	0.36554	4.38E-05	0.76311	0.684479
ResNet18	SGD	128	0.67401	0.08793	0.45189	2.47E-05	0.028660
VGG	RMSProp	32	4.15E-05	0.36617	1.59E-05	0.44255	0.702969
VGG	RMSProp	32	1.10E-05	0.30451	8.89E-05	0.86951	0.694063

Table 1. Comparison of different training approaches for different models and hyper-parameters for 50 epochs

For example, the learning rate was between 0.1 to 0.00001; batch size was set to 8-128; adam, sgd , rmsprop as optimizer; dropout rate of 0.2 to 0.5; weight decay from 1e-3 to 1e-5 and momentum of 0.1-1.0. For each trial a different configuration within this range was set and a total of 100 trials were tested for each model. Although there were more images of tuning for other models, we included only ResNet50 as it was our best-performed model. In the images, the green dots represent the objective values which are the validation accuracy for 10 epochs in this case after each trial and the blue line represents the optimization curve of the tuning process. After the trials we got the best hyper-parameters as shown in table 2.

From Figure 5 it can be clearly observe a difference of the training and validation accuracy of the EuroSAT Data set. To start with, in the left figure while training the data sat with transfer learning the training and validation accuracy are progressive and have a good learning curve. On the contrary, in the other figure the accuracies are very poor.

Hyperparameters	Mini-ImageNet	EuroSat (RGB)
Learning Rate	4.2412e-6	2.4895e-05
Optimizer	RMSProp	Adam
Batch Size	64	8
Dropout Rate	0.43548	0.40102
Weight Decay	8.5096e-6	4.9412e-05
Momentum	0.48491	0.55699

Table 2. Best hyper-parameter configuration

This is because, as the data set was not large enough and only 25 images were included for training while 75 for testing. Hence, the training was over-fitted after some epochs and validation accuracy did not cross even 35%. However, with transfer learning it went above 70% which justifies our main objective. Therefore, we can say that transfer learning makes a great improvement of the performance of the model when it is applied on a smaller data set.

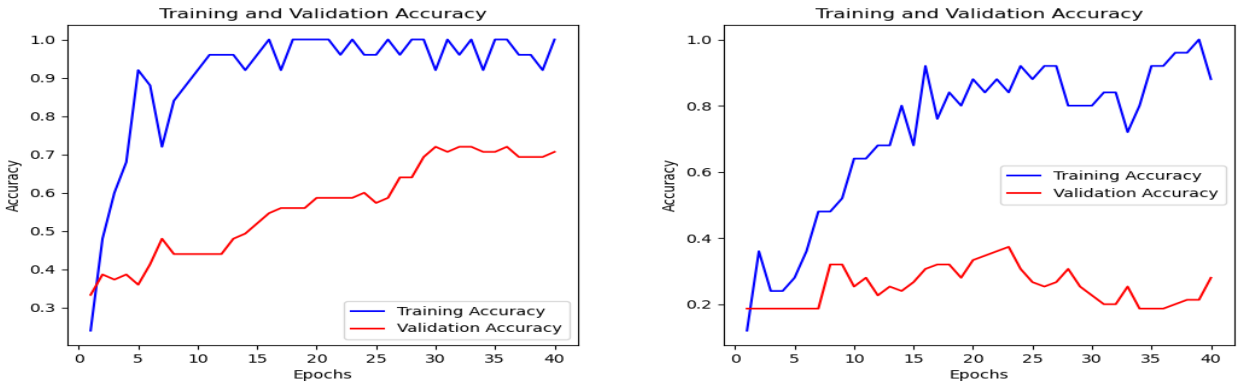

Figure 5. illustrates comparison between with(left) and without(right) transfer learning for EuroSAT dataset

4. Conclusion

In this paper, we have upheld the comprehensive analogy of our experiment on deep transfer learning. In short, we pre-trained a ResNet50 model on Mini-ImageNet and later deployed it on EuroSat(RGB). We have tried different approaches such as data augmentation, model variations, auto hyper-parameter tuning etc. All of these have played a great role in achieving our primary objective which was increasing the accuracy of the model while training on the latter data set. However, There are several additional strategies which can improve the performance of the model even more. For example, pre-training with the Vision Transfer Model could give a better score as it is also very suitable for transfer learning. Also, methods like "learning rate scheduler" and "early stopping" might also provide a good result as they can optimize and reduce loss in epochs dynamically. Running further trials with auto optimization can also increase the chance to find even better parameters which can increase the performance of the model.

References

[1] Emmanuel Abbe and Colin Sandon. Provable limitations of deep learning. *arXiv preprint arXiv:1812.06369*, 2018. 1

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019. 3

[3] C. L. Philip Chen. Deep learning for pattern learning and recognition. In *2015 IEEE 10th Jubilee International Symposium on Applied Computational Intelligence and Informatics*, pages 17–17, 2015. 1

[4] Abolfazl Farahani, Behrouz Pourshojae, Khaled Rasheed, and Hamid R. Arabnia. A concise review of transfer learning. In *2020 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 344–351, 2020. 1

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2, 3

[6] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. In *IGARSS 2018-2018 IEEE international geoscience and remote sensing symposium*, pages 204–207. IEEE, 2018. 2

[7] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019. 2

[8] Shuai Huang, Jing Li, Kewei Chen, Teresa Wu, Jieping Ye, Xia Wu, and Li Yao. A transfer learning approach for network modeling. *IIE transactions*, 44(11):915–931, 2012. 1

[9] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. 1

[10] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroSP)*, pages 372–387, 2016. 1

[11] Budhaditya Saha, Sunil Gupta, Dinh Phung, and Svetha Venkatesh. Multiple task transfer learning with small sample sizes. *Knowledge and information systems*, 46:315–342, 2016. 1

[12] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3067–3075. PMLR, 06–11 Aug 2017. 1

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[14] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016. 2

[15] M. Mitchell Waldrop. What are the limits of deep learning? *Proceedings of the National Academy of Sciences*, 116(4):1074–1077, 2019. 1

[16] Xin Zheng, Luyue Lin, Shouzhi Liang, Bo Rao, and Ruidian Zhan. A transfer learning method for deep networks with small sample sizes. In *Journal of Physics: Conference Series*, volume 1631, page 012072. IOP Publishing, 2020. 1

[17] Wenbo Zhu, Birgit Braun, Leo H. Chiang, and Jose A. Romagnoli. Investigation of transfer learning for image classification and impact on training sample size. *Chemometrics and Intelligent Laboratory Systems*, 211:104269, 2021. 1