Transfer Learning Based Image Classification

A.Syed
University of Oulu
abdurrehman.syed@student.oulu.fi
2308546

Abstract

Transfer learning, an efficient approach which transfers knowledge of one model to another in order to increase its generalization and performance. As machine learning evolves, transfer learning stays at the forefront for both efficient model training and fine-tuning pre-existing solutions. This project report highlights the application of transfer learning techniques and their scalability, utilizing ResNet18 and MobileNetV2 architectures for single-shot image classification. The miniImageNet dataset is used for pre-training these models followed by EuroSAT-RGB and Kaggle's Scene-Classification dataset to access accuracy and model performance. Finally, key findings are plotted, emphasizing the impact of transfer learning on smaller training sets.

1. Introduction

Modern machine learning tasks, especially deep learning revolve around the working of human brain, its ability to find patterns, links and adapt accordingly. It consists of complex network of layers, each of which containing an enormous amount of parameters forming a neural network. These networks take information as their input, perform a function and gives an output based on what relationships they have learned. For this reason they require an extensive amount of data and pre-processing efforts for creating, training, testing, and validating efficient models. This process is not only time-consuming but also computationally intensive, requiring numerous iterations to fine-tune it. However, there are certain limitations to these networks, firstly, they require a lot of data to train and learn patterns, with smaller dataset the network would struggle to learn and too much data of the same type may result in overfitting of the model. To address these challenges transfer learning is used, it is an approach which uses industry standard models trained and fine-tuned on vast datasets and are reused on newer data to make efficient predictions as well as handling problems like overfitting and underfitting in traditional neural networks. Additionally, transfer models resolve the issue of low training samples since the model already knows patterns and relationships even with smaller size of data it can make accurate predictions and adapt quickly.

In this paper two transfer models are used ResNet18 (Residual Network) comprising of 18 layers is a convolutional neural network used for image classification. The second model used is MobileNetV2 consisting of 53 layers, similar to ResNet it is also a convolutional classifer. These pre-trained networks are retrained on a single shot dataset called miniImageNet after which these newly learned connections are evaluated on two datasets: EuroSAT-RGB and Kaggle's scene classification dataset.

The motivation behind this report is to understand how transfer-learning can be helpful in giving promising results for general purpose tasks such as image-classification and its effectiveness in single-shot analysis when the training data is negligible; A common challenge in real-world applications. Understanding how these models can answer the mentioned challenges could pave the way for more efficient, accurate, and feasible architectures.

2. Approach

This section explores the methodology of the project in detail.

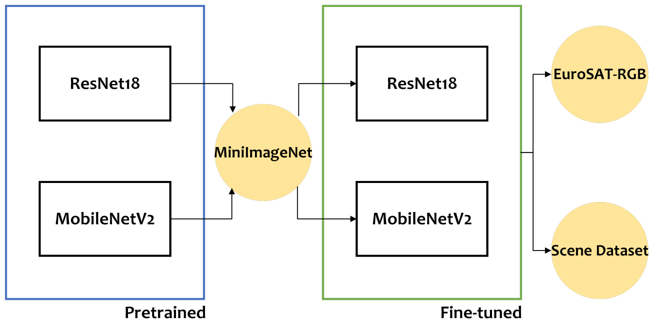

Figure 1. Transfer Learning Approach

2.1. Data Preprocessing

Here you can start to introduce the approach in detail. Initially 60 percent of the miniImageNet dataset was installed and divided into three section train, test and valid, with training section getting 75 percent of the images, valid getting 10 percent and test getting 15 percent. The split was made using random module and a seed value was fixed to keep the models and training consistent. After splitting into lists they're passed on to a CustomDataset class which deconstructs data into labels and images and then applies the transformations to them. Following this step the datasets are then passed on to a DataLoader class which converts them into batches and shuffles the training samples prior to training.

2.1.1 Pre-trained Transformations

Transformations are augmented methods that are applied on images before they're fed into the neural networks. These transformations will help the model converge faster and prevent overfitting. Following are the pre-trained transformations applied on the networks.

Resize: The initial shape of the images is 64 by 64 since the transfer models we are using work on 224 by 224 the images are resized.

RandomHorizontalFlip: This augmentation flips the image so it learns the patterns from a different angle, the probability used for flipping horizontally is 50 percent.

ToTensor: This augmentation after applying the prior changes converts the image to a tensor representation.

Normalize: To make the training computationally easier and faster Normalization is used with constant mean and standard deviation on which the model works best. Finally getting the model ready for the next step.

2.2. Training Pretrained Models

Firstly, the necessary transfer models were loaded (ResNet18 and MobileNetV2), their final layer was accessed and replaced with our custom layer since we will be using 64 classes as output from the final fully connected layer. Following that, the CrossEntropyLoss function was initialized since the problem is of non-binary classification and SGD optimizer is used with a learning rate of 0.001 and regularizations to stop the model from overfitting and converging quicker.

Two methods are created for training and evaluating the neural network which runs for variable epochs learning and calculating loss after each iteration. Similarly, validation and testing accuracies are also calculated to understand the convergence and overall accuracy of the model. No layers were froze from the model so that the model learns everything of the new dataset, assigning new weights and biases.

2.3. Evaluating the Fine-tuned Models

After training the models and updating their weights, now they're tested upon new datasets. EuroSAT and Kaggle's Scene Classification. Since the goal is to test how accurate the models are using Single-shot analysis, the new datasets are divided in a 75, 25 ratio with 75 being the training percentage and 25 is testing. However, the training samples are kept at almost negligible with EuroSAT being 75 images - 25 images while Scene classification dataset having an 80-20 split, 30 images in training and 120 in testing. The images go through the previous steps of preprocessing and some new transformations are applied in this fine-tuned model.

For this model all the layers were frozen except the last fully connected layer which was modified to have 10 and 6 classes(output neurons) for each respective dataset.

The model is trained again but this time only the last layer is trainable and later predictions and evaluations were noted.

2.3.1 Fine-tuned Transformations

Resize and CenterCrop: In this augmentation rather than resizing straight to 224 by 224 a different approach was used, the resize size was set to 256 by 256 and then CenterCrop augmentation was applied which crops the region from the center of the image.

Random Rotation: The images were randomly rotated to 10 and 11 degrees which showed the most accuracy.

Gaussian Blur: Gaussian Blur with a Kernel-size of 3 was added to reduce overfitting and model's dependability on certain regions, acting like a dropout region.

Color Jitter: Only the contrast was changed to 0.2 while saturation, hue and brightness are not changed.

RandomVerticalFlip: These datasets also had a vertical flip added (50 percent probability) since flipping these images would not decrease their accuracy or model's ability to learn. However, since our miniImageNet dataset was based on animals it would've not made sense to apply this transformation.

RandomPosterize: It applies posterization, simplifying the colors of the images with a bit-size of 4 and a probability of 20 percent. Bit-size represents how many levels do we need to retain.

Following are the datasets used and experimented conducted with different training setups along with their analysis.

Multiple experiments were tried and tested, some of which are mentioned in table 1. It represents the training of the resnet model with specified changes and augmentations performed along with the accuracies. After trying out different batch sizes, adding more layers, dropouts, trying different optimizers like Adam, Adagrad, RMSprop it turned out that Adagrad did well however, SGD was still slightly better. Each batch and learning rate had a unique result but the model was overfitting or underperforming in the most of them. Before finally I found a good match of the model which was neither overfitting nor under-performing and the stats were: batch-size 64, SGD, 0.001 lr, 0.9 momentum, CrossEntropyLoss as the lossfunction, tried NLLLoss but didnt work really well.

Now for MobileNetV2, I again tried different parameters mentioned in table 2. A number of variations similar to ResNet were applied and again most were overfitting while underperforming until I found the correct match which were 0.001 lr, 0.001 weightDecay and 0.87 momentum, turns out 0.9 momentum was giving a drastically bad result, SGD optimizer with CrossEntropyLoss my training acc was 90.8 test 85 and valid at 85.8.

2.4. Datasets

MiniImageNet: A subset of ImageNet, designed specifically for the purpose of single-shot learning. It comprises 60,000 images equally distributed among 100 classes. Used for the classification task in this project.

EuroSAT-RGB: A dataset composed of satellite imagery; shots taken by Sentinel-2. It contains 27,000 labeled images of 10 unique classes. Used as an evaluator for the pretrained model.

Kaggle's Scene-Classification Dataset: A dataset consisting of more than 24,000 labeled images of scenes spanning across six classes such as mountains, roads, forest, etc. It is also used as an evaluator for the models.

2.5. Example of Table

3. Conclusion

This report shows how transfer learning can made such a huge impact when performing single shot tasks as well as counter overfitting. It is an easy approach to implement and use while saving time and execution, leading the foundation for future ideal models.

Changes	Avg Valid Acc	Avg Test Acc
No augment, no batch, lr 0.0001	69.2	68
HorizontalFlip, batch 8, lr 0.0001	70.2	69.2
HorizontalFlip, batch 64 lr 0.001	84.6	84

Table 1. ResNet-18 training table

Changes	Avg Valid Acc	Avg Test Acc
batch 64, lr 0.001 decay 0.0001 momentum 0.9	68	69.2
batch 32 lr 0.01 momentum 0.9	70.2	69.2
batch 32 lr 0.001 decay 0.001 momentum 0.87	70.2	69.2

Table 2. MobileNetV2 training table

Changes	Avg Valid Acc	Avg Test Acc
batch 64, lr 0.001 decay 0.0001 momentum 0.9	68	69.2
batch 32 lr 0.01 momentum 0.9	70.2	69.2
batch 32 lr 0.001 decay 0.001 momentum 0.87	70.2	69.2

Table 3. MobileNetV2 training table

Average Testing Accuracy for every executions
82.3
80
81
80
85

Table 4. MobileNetV2 Fine Tuned table

References

Average Testing Accuracy for every executions
86.6
87.2
83.3
92.0
81.333

Table 5. MobileNetV2 Fine Tuned table