Transfer Learning

Submitted as final project report for Practical Topics in Machine Learning, BIU, 2021

1 Introduction

Transfer Learning is a method in machine learning where a learner developed in one domain is used to improve a learner from a different domain. It is done by transferring the information learned in the first domain, to be used as the starting point for the new domain.

In some scenarios is it hard to find a good set of training data to be used for training the learner, and therefore there is a need for a different method. Transfer Learning allows us to use information gathered in a related domain, and to use it to improve our results on the new domain. Furthermore, it can help save training time, by using a pre-trained model that has already run a sufficient amount of time and epochs, and therefore allows rapid progress in the learning stage of the model.

We wanted to test the productivity of using transfer learning on medical images. We are interested in seeing if transfer learning can be used to achieve better classification on a medical diagnosis, given training on a different medical problem, As opposed to training our model directly to our target data. We also thought it could be interesting to see if the technology used on the images has an effect on the success of transfer learning. If we choose two diseases that use the same kind of technology (i.e. X-ray) or different kinds of technology, will we get different results? In [8] it is said that the transfer learning should be used when the training data and target data exist in the same high-level common domain. Is it enough that the data are all images? Or does the kind of image matter - will we get better results if our training data and target data both use Histopathology Images? Or does the number of images used for training the model play a more significant role in the process?

In our project we chose to use the VGG neural network, specifically VGG11.

1.1 Related Works

For the definition of transfer learning we used [8]. For implementing VGG and transfer learning, we referenced [1] and [7].

1.2 Datasets

For our training and target datasets we tried 2 different datasets that use Histopathology imaging, colon cancer [2] and breast cancer [3], and also [4] for pneumonia and [5] for lung opacity, Both being x-ray images. We also used ImageNet [6] indirectly as our training dataset, when testing VGG that was pre-trained.

In order to use the data, we needed to split it to 2 folders, one of which was labeled 0, and the other one was labeled 1. For splitting a dataset to train and test, we created 2 folders - train, test, and had in each the 2 folders mentioned above.

2 Solution

2.1 General approach

We tried three different approaches for training our datasets using VGG:

1. using VGG with weights already pre-trained on ImageNet. We than did transfer learning to our target dataset.
2. using VGG with weights initialized randomly, and training our model from scratch.
3. training one dataset using VGG and using transfer learning to train a target dataset.

2.2 Design

We used VGG11 as we achieved good results and the model has relatively fewer layers than other versions of VGG, so it took less time than other VGG models. We also used data augmentation in all our approaches, as it helps train our data in the best way, and helped achieve better results.

There were also different issues that needed to be addressed in each approach that we used:

First Approach. For this approach we needed to match the size of our images and do normalization, as the pre-trained classifier layer expects certain features to appear in certain places within the flattened 512x7x7 output of the features layer after the adaptive average pooling. Using a different image size than was used to pre-train the model causes features sent to the classifier to be in different places than expected, and thus leads to poor performance when using the pre-trained model.

In ImageNet the final layer has a 1000 dimensional output, but as we need binary classification, We changed the last layer specifically by indexing into the classifier layer of the pre-trained model. Our dataset only has 2 classes, so we adjusted the last layer to have 1 dimensional output for binary classification.

Second Approach. In one of the datasets tested [2] The original size of the images was 768 x 768 pixels. VGG has many layers and training such large images takes a lot of time and memory, so we changed the images to a smaller size while making sure the accuracy stayed high.

also in this case we needed to adjust the output to be 1 dimensional, the same way that we did it in the first approach.

Third Approach. We needed to build the VGG model so that we could train it from scratch and also use the intermediate layer for transfer learning. And so we implemented the VGG11 model ourselves, but made sure that the layers would be the same as in the previous models mentioned.

Furthermore, both the training dataset and the target dataset need to have the same size. The breast-cancer [3] dataset is only 50, but the colon [2] is much bigger, so we transformed both to size 100, so we don't loose to much information in shrinking [2].

3 Tests

Learning Rate

We tried 3 options:

- 0.001
- 5e-4
- 1e-7

We referenced [7] to chose the learning rates that we tested. 1e-7 and 5e-4 had better results, but the differences were actually very small. At the end we chose 5e-4.

Epochs

We tried 5, 8, 10 epochs. After that there wasn't much of an improvement between epochs. We adjusted the number of epochs per dataset, in order to avoid over-fitting.

Approaches

There were different variables to test in each approach:

First Approach. We tried freezing the values, i.e. Instead of training all of the parameters we have loaded from a pre-trained model, we instead only learn some of them and leave some "frozen" at their pre-trained values. As our model will then have less trainable parameters it will usually train faster. We got a lower accuracy so for our final results we chose not to do so.

As mentioned previously, we needed to change the last layer to receive binary classification. We tried going from 4096 features straight to 1 and from 4096 features to 10 features to 1. We got similar results but the second option was slightly better, so we chose that one.

Second Approach. The original size of the colon [2] was very large and took an unusable amount of time, and too much memory for us to use, and so transformed the size to 100.

Also here we tested the drop from 4096 features to 1 - directly and going from 4096 to 10 to 1. The results and conclusion are the same as in the previous case.

Third Approach. In this approach we tried using the model of pytorch, as we used in the previous approaches, and adding layers to that to achieve transfer learning, but it was cumbersome and complicated so we decided to implement VGG ourselves and that way return from the function also a layer in the middle. Just as we did in the homework assignment during the semester.

Furthermore, as mentioned, we tested 4 different datasets for the training and target [2][3][5][4].

Also here we needed to end with binary classification, so we tested the aforementioned options and also a slightly milder transfer; We tested going from an output of 32 features to 16, 10, 1. This gave us the best results.

4 Results

4.1 Third Approach

The results of our tests on different datasets are in the table below.

training/target	eval	test
training-breast cancer	86.15%	
target-colon		98.54%
training-breast cancer	90.03%	
target-lung opacity		91.23%
training-breast cancer	90.00%	
target-pneumonia		89.00%
training-lung opacity	88.23%	
target-breast cancer		87.95%
training-colon	99.45%	
target-breast cancer		87.95%
training-pneumonia	88.99%	
target-breast cancer		88.39%

Although we can see that in the first case the eval on the training dataset is the lowest, the results after transfer learning are by far the best. Therefore, we decided to use colon as our target dataset on our first approach, although we tested both colon and breast cancer on our second approach.

On our chosen datasets we also checked the time taken:

training/target	time per epoch
training	15 minutes
target	2 minutes

4.2 Second Approach

As, mentioned, we tested both breast cancer and colon.

dataset	eval	test	time per epoch
breast cancer	85.95%	87.90%	15 minutes
colon	97.81%	98.96%	2 minutes

We would like to mention the the breast cancer dataset has many more images than the colon, and that explains the discrepancy in the amount of time the epochs took on each dataset.

4.3 First Approach

In this section we only tested on the colon dataset, however, we tried freezing the values to see the effects. With frozen values we got an eval of 94.96%, however the *train* was 78.87%. As mentioned, we decided not to freeze the values.

without freezing the values, we got an eval of 99.90%, and each epoch took about 4 minutes.

eval	time per epoch
99.90%	4 minutes

We would like to note that in this approach, the size of each image from the colon dataset was 224, but in the second approach we used a starting size of 100, and so the training was faster in that section.

5 Discussion

After all our tests we conclude that the best way to train our dataset is using the pre-trained VGG model. Although the results were very similar in all approaches, it seems that the most useful characteristic was the richness of the ImageNet dataset, with the large amount of images that it provided.

However, it is interesting to note that looking only at our results using the third approach, our best result was using breast cancer for training and colon as our target dataset. In that section we tested breast cancer as our training dataset for each of our 3 other datasets, since breast cancer was a relatively larger dataset, and we hoped would help us achieve good results after the transfer. As mentioned, using colon as our target dataset gave us by far the best results, so perhaps similar imaging technologies do have an advantage.

However, given the success in training colon in our second approach, without transfer learning, and given that our success in training the breast cancer dataset was not so high in all our attempts, it is more likely that colon is a relatively easy dataset to train and get good results on, and that in this case transfer learning is simply not necessary.

6 Code

We used a colab notebook. You can open the notebook here.

References

[1] Will Koehrsen. Transfer Learning with Convolutional Neural Networks in PyTorch . https://towardsdatascience.com/transfer-learning-with-convolutional-neural-networks-in-pytorch-dd09190245ce, 2018. [Online; accessed 13-September-2021].

[2] Larxel. Lung and Colon Cancer Histopathological Images. https://www.kaggle.com/andrewmvd/lung-and-colon-cancer-histopathological-images, 2020. [Online; accessed 13-September-2021].

[3] Paul Mooney. Breast Histopathology Images. https://www.kaggle.com/paultimothymooney/breast-histopathology-images, 2017. [Online; accessed 13-September-2021].

[4] Paul Mooney. Chest X-Ray Images (Pneumonia). https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia, 2018. [Online; accessed 13-September-2021].

[5] Tawsifur Rahman. COVID-19 Radiography Database. https://www.kaggle.com/tawsifurrahman/covid19-radiography-database, 2021. [Online; accessed 13-September-2021].

[6] Princeton University Stanford Vision Lab, Stanford University. ImageNet. https://www.image-net.org/index.php, 2021. [Online; accessed 13-September-2021].

[7] Ben Trevett. PyTorch Image Classification : 4-VGG. https://github.com/bentrevett/pytorch-image-classification/blob/master/4_vgg.ipynb, 2020. [Online; accessed 13-September-2021].

[8] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.