On Finding the Best Strategy for Limited Dataset Deep Learning

Neelabh Pareek[#1], Nicholas Christman[*2]

[#]*Department of Computer Science, Columbia University in the City of New York, New York, NY 10027*

[1]np2647@columbia.edu

[2]nc2677@columbia.edu

***Abstract*— Many companies and machine learning practitioners do not have access to extremely large datasets. This constraint makes it very difficult to achieve model effectiveness. The research covered herein provides a perspective on three strategies that have been known to excel on jobs that are constrained by limited data -- namely, data augmentation, transfer learning, and one-shot learning. Each method is compared to the performance of a baseline ResNet-18 model that is trained using the entire dataset. In the end, we show that there is no "one size fits all" approach to choosing a model or process for limited dataset deep learning.**

***Keywords*— deep learning, limited dataset, data augmentation, transfer learning, one-shot learning.**

I. Introduction

It would seem that in the past couple of decades, industries have generated more data than they know how to use, consuming a superfluous amount or resources to acquire data that *might* be important. At the turn of the century as data storage technologies continued to evolve, companies now had the ability to store all this seemingly valuable data but the technology to process these large amounts of data had not yet matured [9]. As we are all aware, however, the re-invention of artificial intelligence (AI), machine learning (ML), and deep learning (DL), have significantly improved a company's ability to leverage *big data* and make more informed business decisions -- a data-driven approach to solve complex problems using the large amounts of data that had been collected since the turn of the century [9].

The objective of this work, on the contrary, is to provide a perspective into the complexities of dealing with small datasets while leveraging the resources developed to process big data. The impact of different novel learning strategies utilized for limited dataset applications will be surveyed. Specifically, we plan to explore the following hypotheses for how stakeholders might overcome challenges with limited data:

(i) using data augmentation (pre-processing) to extend a smaller dataset,
(ii) applying transfer learning using a base model, and
(iii) exploring one-shot learning for difference detection (limited use-cases).

For all hypotheses, a larger dataset will be sampled as evenly as possible to create a limited set with the same training/test data split. This makes it possible to have a base model trained as a control variable during the experiments, providing ground-truth results to compare performance improvements and degradations.

A. Problem Motivation

As alluded to above, many companies and ML practitioners may not have access to extremely large datasets -- for example, a startup company in a niche or new industry may only have a very limited and likely biased dataset from a beta launch. Where state-of-the-art neural networks generally form deep, complex networks, we know that to effectively train these networks generally requires a large amount of data [2]. It is thus easy to recognize that businesses wishing to leverage ML technology with limited data will have difficulty achieving model effectiveness. The question that motivates this research is simply, what are some ways for a company to overcome this constraint?

Conducting the experiments outlined above will provide insight into the different techniques and

their effectiveness at addressing the concerns of using ML on limited datasets.

II. Background

Despite the notion of "big data", many companies are limited in their ability to produce reliable model results and performance due to the lack of quality training data. As such, this area of limited data and training with limited labelled data is well researched.

For data augmentation we looked at two background research. Oh et al. conducted a data augmentation experiment on their extremely limited dataset implementation for classification of chest x-rays images. Their idea was to take their small dataset and subsample 100 images from each individual image yielding an augmented dataset that was 100 times the size of their original dataset [1]. Peng et al. were able to implement a second network to perform the complicated augmentation of facial alignment [2].

For transfer learning, we looked at research from Pan et al.. They conducted a survey of transfer learning techniques and their effectiveness [4]. We also looked at published tutorials as guides of how to implement successful transfer learning networks. [3].

One shot learning is a technique that is specifically useful in limited training examples. A training dataset is provided to a pre-trained network with only one sample per class. Siamese networks are very useful because they can be trained on an original dataset such as Omniglot and then use one shot learning to update the weights such that performance in the MNIST dataset is achieved [4][5][8]. We looked into a number of other resources that worked through the implementation for Siamese networks in different frameworks [6][7].

III. Technical Challenges

The challenges of this project were primarily encompassed with curating a dataset and creating an architecture that would facilitate limited data experiments. In scientific experiments, there exists a control experiment that is executed to understand the impact of making a change. In our experiments, we needed a similar control. This control would be the basis that each experiment would be compared against.

Another challenge was the decision of which dataset to use. There could only be one dataset chosen because this would need to be used for all experiments. Though many datasets exist, not all are appropriate for the types of experiments we wanted to execute. We chose an MNIST dataset because it is applicable to most model types. Though it is a great dataset for the reason mentioned above, after conducting the experiments we realized that the most models obtain near perfect accuracy for the MNIST dataset. This meant that the control experiment resulted in near perfect accuracy which made it difficult to understand the true improvements seen from the experiments. A possible next step would be to redo these experiments with a different image classification dataset that yields lower accuracies for the control.

IV. Approach

A. Theory

The approach can be broken down into the control and the different experiments we conducted. The control test was to train the ResNet-18 model using the full MNIST dataset. All experiment metrics were compared back to this control. We would expect that limited data experiments would not perform as well as the control because of the dataset size. However, we are interested in achieving performance as close as possible to the control.

The first experiment was to conduct a survey of different models on a limited dataset. The dataset is a 1% subset of MNIST that contains evenly distributed classes. Four different models were trained: ResNet-18, ResNet-50, VGG-11, VGG-16.

The second experiment was to implement different data augmentation transformations: Random Crop, Horizontal Flip, Random Rotate, Random Erase. These four transformed limited MNIST datasets were used to train a ResNet-18 model. A fifth trained model was created which combined all four of the above transformations into one dataset. The purpose of this was to validate our belief that one cannot blindly select transformations to apply to a dataset. The selection of data

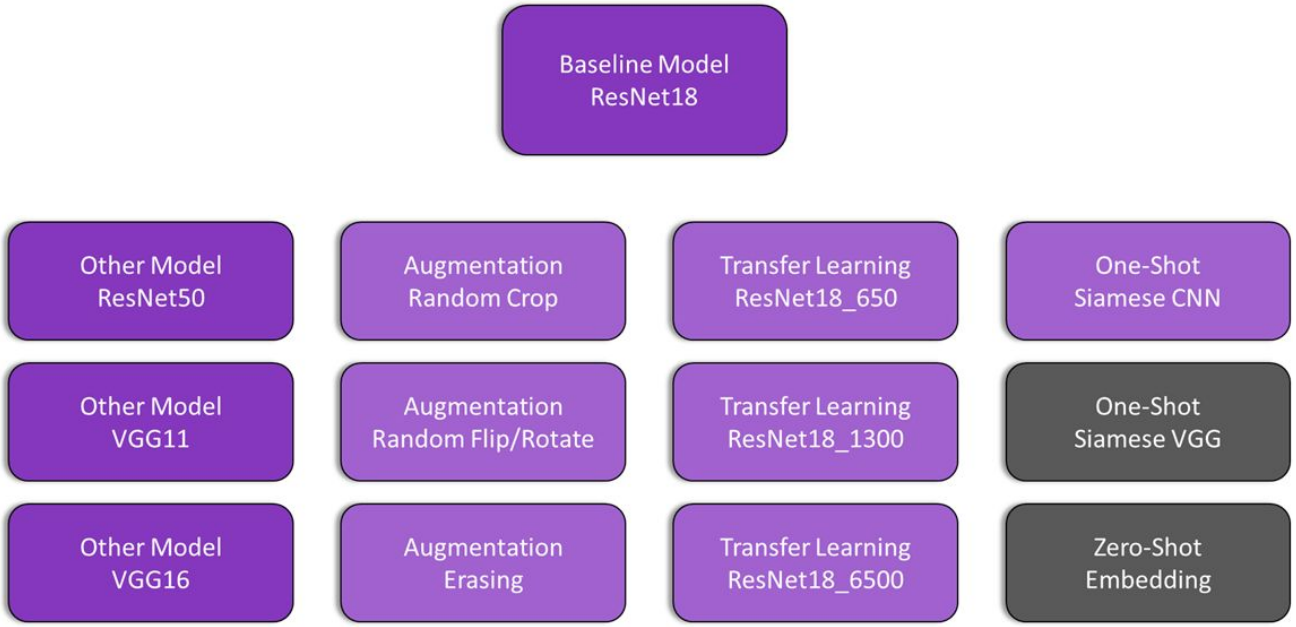


Fig. 1 Hierarchical architecture showing the flow of the solution. At the top is the baseline model (a ResNet-18 trained on the full dataset. The baseline results were used as "ground truth," indicating the absolute maximum performance achievable on the MNIST dataset as a whole. To the left (in dark purple) are a small colleton of other well known models that were also trained on the full dataset -- these additional models were included as an additional reference point, with the intention to further illustrate the complexities of limited dataset training. From left to right are each of the tasks required to achieve the project goals: data augmentation, transfer learning, and one-shot learning. Note: zero-shot learning was included for completeness, but is outside the scope if this project.

augmentation techniques needs to be carefully chosen or the model will perform poorly. In the case of blindly selecting all transformations together, the model performed poorly as expected.

The third experiment was to use a pretrained ResNet-18 model and measure its effectiveness on the stock limited dataset used in the first experiment. The model was pre-trained using ImageNet. This experiment would mainly be compared with the results of the limited dataset ResNet-18 training conducted in the first experiment. A secondary comparison would be made to the control.

The fourth experiment was to implement a Siamese network and understand the effectiveness of one shot learning compared to the other experiments.

B. Architecture

The project was implemented using the Google Cloud Platform - AI Platform offering. Jupyter Notebooks were used to implement each experiment. The hardware used was the N1-standard-8 machine. which contains 8vCPUs and 30GM RAM. To keep training time to a minimum, a Nvidia V100 GPU was used.

The software architecture was based around the PyTorch implementation. For this CUDA 10.1, Python 3.7.8, PyTorch 1.7.1 and Torchvision 0.8.2 were used.

V. Solution & Implementation

This section provides more details regarding the approach taken to achieve our goal. First, the solution architecture developed for this project is illustrated in Figure 1. Observe that we selected the ResNet-18 as our baseline model -- this choice was in part due to the positive reputation of the ResNet structure in addition to the lower complexity of the ResNet-18 structure (making it a good choice for the low number of classes in the chosen MNIST dataset).

A. MNIST Dataset

For these experiments we chose to leverage the MNIST database of handwritten digits (0-9), a

dataset consisting of a 60,000 training image set and a 10,000 test image set. It is worth noting that MNIST is a subset of images available from NIST, where the images have been size-normalized and centered in a fixed-size image. To accomplish our objective, we would need to further subsample the whole dataset. To generate the limited dataset of N training samples, we extended the PyTorch Torchvision MNIST dataset class to evenly subsample the whole dataset. This allowed us full control over the class-distribution of images that were used for each of the tasks defined below. For convenience, we will refer to the entire MNIST dataset as "full dataset" and the subsampled MNIST dataset as "limited dataset" in this document.

B. Baseline Performance

A limited dataset was used to train the baseline models. The size of the dataset was: Training - 650, Validation - 107, Prediction - 6. Thus near 1% of the MNIST dataset is used to emulate the limited dataset constraint. Four baseline models were used to understand baseline performance: ResNet-18, ResNet-50, VGG-11, and VGG-16. Each model was trained with the same limited dataset constraint. None of the models were fine tuned.

C. Data Augmentation

Data augmentation was conducted using the four transforms mentioned earlier: random rotate, random crop, random erase, and horizontal flip. Only one transformation was applied to a dataset at a time. None of the transformations were fine tuned to achieve optimal performance. This experiment was to understand which of these transformations was viable while keeping all other variables constant.

Random crop was performed on a 15x15 area with no padding. This is a fairly large area of each image and could be the cause for the random crop augmentation technique to perform worse than other transformations. Fine tuning would be needed to understand more.

Horizontal flip was performed on half of the dataset. The results were fair and the expectation is the performance would improve as the dataset size increases.

Random erase was performed similarly to horizontal flip in that it was done to half the dataset. Random erase is similar to random crop however it performed much better. This may have been because erase acts as adding block noise rather than removing information from the image.

Random rotate was performed between -90 and +90 degrees. This was done to prevent a full rotation which would make a 6 look like a 9. However this did not prevent a 6 from being rotated -90 and a 9 being rotated +90. Even with this shortcoming, this still resulted in a better performing augmentation technique.

D. Transfer Learning

To accomplish the transfer learning task, we leverage the PyTorch ResNet-18 pretrained model. The objective of using this model with pretrained weights was to observe any performance benefits over the baseline ResNet-18 trained on the limited dataset. Because the ResNet-18 pretrained model was trained using ImageNet data (a completely different type of data compared to MNIST), we do not anticipate a performance increase over the baseline model; however, by performing this experiment we should still be able to collect valuable data in support of our objective.

E. One-Shot Learning

One-shot learning is a slightly different approach than augmentation and transfer learning; however, the goal of one-shot learning is similar in that it aims to detect the presence of a known category in an image that contains clutter (i.e., other uninteresting objects) [5]. Where the MNIST dataset consists of images with very low clutter, it is unclear how one-shot learning performance will be affected (e.g., our model will likely be overfit). The underlying objective of one-shot learning, however, is to provide reliable model performance with only a few labeled images [5]. Therefore, the one-shot learning method was implemented using the limited MNIST dataset with the understanding that it may not perform well.

The implementation was carried out by first exploring an existing one-shot learning network -- that is, the Siamese CNN. Moreover, to first gain domain knowledge we loosely followed online

tutorials that developed networks for the Omniglot dataset (a well-known dataset commonly used to train one-shot models) [6][7][8]. This domain knowledge was then transferred to our application of using the limited dataset. To accomplish this, we again extended the Torchvision MNIST dataset to generate a subset of positive and negative image pairs. That is, we generated a binary classification dataset of positive image pairs (i.e., both images belong to the same class) labeled one (1) and negative image pairs (i.e., both images belong to different classes) labeled as zero (0). With this extension to the Torchvision MNIST dataset, we were able to utilize all of the PyTorch functionality (transforms, dataloaders, etc.).

VI. Experimental Results

This section discusses the results of the control and experiments.

A. Network training results

In the table below is a summary of the training and validation results collected during these experiments.

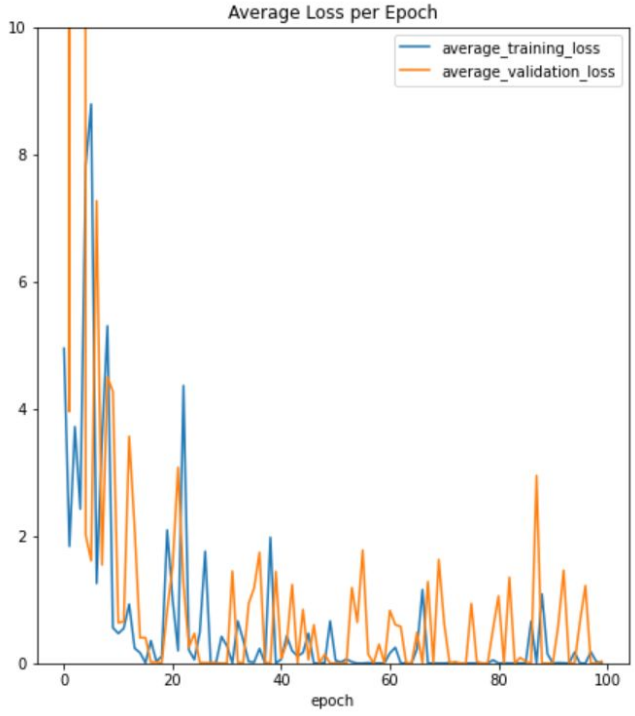
TABLE I
Showing the training/validation results for each model.

Training Results			
Model	**Final Accuracy (%)**	**Final Loss**	**Total Time (sec)**
Resnet18 ***Full***	100.0 / 99.4	0.0 / 0.035	2,924.8
Resnet18 ***Limited***	100.0 / 94.2	0.002 / 0.365	68.7
Resnet50 ***Limited***	100.0 / 91.3	0.002 / 0.400	69.3
VGG-11* ***Limited***	9.9 / 10.7	nan / nan	102.6
VGG-16* ***Limited***	9.9 / 10.7	nan / nan	108.8
Augmentation** ***Limited***	95.7 / 94.4	0.169 / 0.210	72.3
Transfer Learn. ***Limited***	97.5 / 87.4	0.069 / 0.443	72.7
One-Shot*** ***Limited***	100.0 / 96.9	0.133 / 0.559	102.6

* VGG-11 and -16 did not perform well on this dataset.
** Showing the best performing augmentation, random rotate.
*** One-shot was trained using Omniglot because the MNIST dataset was causing the network to diverge but the issue was never resolved. This data is included here only for completeness.

B. Network validation results

The table below summarizes the validation results for each of the limited data training approaches.



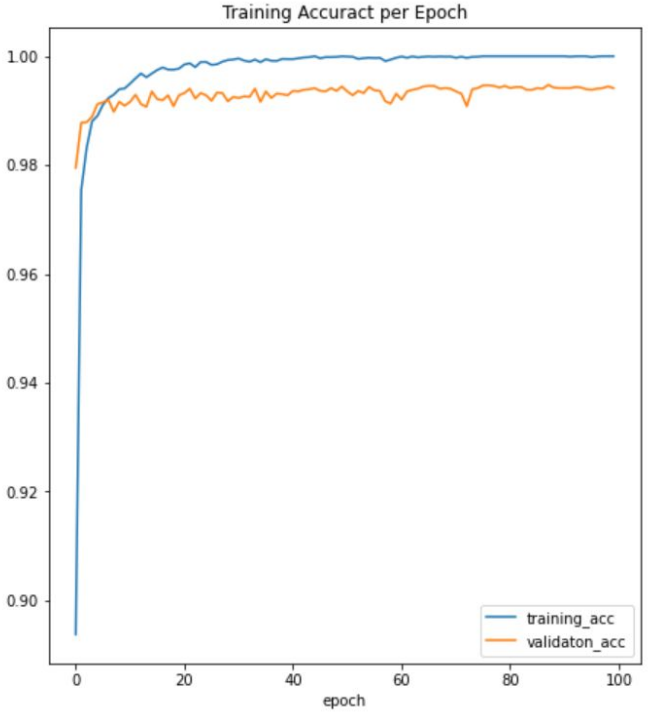


Fig. 2 Show resulting performance plts from training the ResNet-18 on a limited MNIST dataset with only 650 images.

VII. Conclusion

The goal of the investigations was to understand the different techniques that could be used to enhance the trained model efficacy for limited dataset applications. Through research and our own implementation, we saw that there is no set solution for achieving the best result for all limited datasets. Techniques such as random rotate work well for this dataset and random crop do not. For another dataset it may be the opposite. The same applies for models when comparing ResNet-18 to VGG. Next steps for this project would be to expand on these results. Some expansions would be to use other data augmentation techniques, implement Zero-Shot Learning, fine tune each individual approach, try different types of datasets (RGB images, audio, RF, etc.).

References

[1] Oh, Yujin, et al. "Deep Learning COVID-19 Features on CXR Using Limited Training Data Sets." IEEE Transactions on Medical Imaging, vol. 39, no. 8, Aug. 2020, pp. 2688–700. DOI.org (Crossref), doi:10.1109/TMI.2020.2993291.

[2] Peng, Xi, et al. "Learning Face Recognition from Limited Training Data Using Deep Neural Networks." 2016 23rd International Conference on Pattern Recognition (ICPR), IEEE, 2016, pp. 1442–47. DOI.org (Crossref), doi:10.1109/ICPR.2016.7899840.

[3] Transfer Learning for Computer Vision Tutorial — PyTorch Tutorials 1.7.1 Documentation. https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html. Accessed 04 Dec. 2020

[4] Pan, Sinno Jialin, and Qiang Yang. "A Survey on Transfer Learning." IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, Oct. 2010, pp. 1345–59. DOI.org (Crossref), doi:10.1109/TKDE.2009.191.

[5] Li Fei-Fei, et al. "One-Shot Learning of Object Categories." IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 4, Apr. 2006, pp. 594–611. DOI.org (Crossref), doi:10.1109/TPAMI.2006.79.

[6] Lamba, Harshall. "One Shot Learning with Siamese Networks Using Keras." Medium, 17 Feb. 2019, https://towardsdatascience.com/one-shot-learning-with-siamese-networks-using-keras-17f34e75bb3d.

[7] Cheng, Ta-Ying. "Building a One-Shot Learning Network with PyTorch." Medium, 31 May 2020, https://towardsdatascience.com/building-a-one-shot-learning-network-with-pytorch-d1c3a5fafa4a.

[8] Holländer, Branislav. "Siamese Networks: Algorithm, Applications And PyTorch Implementation." Medium, 24 Sept. 2018, https://becominghuman.ai/siamese-networks-algorithm-applications-and-pytorch-implementation-4ffa3304c18.

[9] Taylor-Sakyi, Kevin. "Big Data: Understanding Big Data." ArXiv:1601.04602 [Cs], Jan. 2016. arXiv.org, http://arxiv.org/abs/1601.04602