Neural Networks for Machine Learning Project

Zane Bayer
Dept. of Applied Science, College of William & Mary

ACM Reference Format:
Zane Bayer. 2022. Neural Networks for Machine Learning Project. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

The following is a brief summary of the methods and results of the semester project for the course 'Neural Networks for Machine Learning' (CSCI520, Spring2022). Both the methods and results sections are broken into three subsections: Convolutional Neural Networks for Image analysis, Transformers for Text Analysis, and Variational Autoencoder for Image Analysis. In the methods section, the statistical methods will be detailed as well as the training procedures for the various models. The results section will detail the findings of the statistical analyses and neural network models for each subsection. The statistical analyses relied heavily on visualizations of the data and relevant statistics. Only some of these figures could be included in this paper, it's encouraged that the reader refer to the python notebook for the full set of figures.

2 Methods

In general, the statistical methods applied for exploratory data analysis were chosen in an attempt to gain a fuller understanding of the data sets numerically and visually. The statistical methods detailed below are standard procedures in both image analysis and natural language processing, respectively. The neural network models were either provided in source references or built using the TensorFlow and Keras Libraries.

2.1 Convolutional Neural Networks for Image analysis

The source for this section was He et al.'s 'Deep Residual Learning for Image Recognition' (2015). The primary objectives of this section were to do an exploratory data analysis of one the data sets used in the He's publication and to reproduce some of his results with residual neural networks(ResNets). A statistical analysis of the CIFAR-10 image data set was performed. The data set was obtained from the Keras library's collection of data sets. CIFAR-10 is a standard benchmark data set for image analysis. The statistical analysis began by collecting simple metrics from the data set such as number of samples, class distribution, and target labels. To gain a further understanding of the images within each class, analysis was performed on the mean images of each class. The mean image for each class is the image produced by first dividing the images by their respective class, then computing the mean of those images within each class. For the CIFAR-10 data set, this procedure produced ten (32,32,3) images. The intention of creating these mean images is to view the common traits, such as colors and shapes, which are common within a given class. For a better visualizations of the shapes and pixel intensities, further manipulation of the mean images was performed to create heat maps. These heat maps were creating computing the magnitude of the pixel intensities by the following formula: $\sqrt{R(x,y)^2 + G(x,y)^2 + B(x,y)^2}$ where RGB refer to three color image channels. This further reduced the ten mean images of size (32,32,3) to ten images of size (32,32). Finally, principal component analysis(PCA) was performed on the images do reduce the dimensions for visualizations and a scree plot was created to gauge the effectiveness of PCA for the data set. Two ResNet models from the Keras library with 50 and 101 layers were used for training. Unfortunately, the ImageNet 2012 data set used by He et al. is not easily available through TensorFlow. Further, the size of the data set(14 million images) poses difficulties in terms of data management and computational resources for training. For these reasons, the two ResNet models were trained on the CIFAR-100 data set which is readily available through the Keras library. Care was taken in performing training in a similar manner to the source paper with the Adam optimizer, momentum, and learning rate decay.

2.2 Transformers for Text Analysis

The source paper of this section was Vaswani et al.'s 'Attention is All You Need' (2017). The primary objectives of this section were to do a statistical analysis of one of the natural language processing data sets available through the Keras library and to train a transformer model on one of these data sets for classification. Statistical analysis was performed on the IMDB data set available through the Keras library. The IMDB data set is a natural language processing data set which consists of movie reviews of varying length with

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

binary targets representing either a positive or negative sentiment. Basic statistics such dimensions, class distribution, review length, and word frequency were computed. PCA component was then performed by augmenting the data set. To place the vectors in the same dimensional vector space (i.e. make the vectors equal length), the maximum length movie review was found and all other samples were zero-padded to this length. PCA was then performed for visualization purposes and a scree plot was generated to gauge the effectiveness of PCA. The provided source implementation of the Transformer was trained on the Reuters data set. The Reuters data set is a collection of excerpts from Reuters publications and the task associated with the data set is identifying subject matter from the excerpts. There are 45 possible classification categories. For training, the vocabulary size was reduced to the 20,000 most frequently used words and the inputs were cut to a maximum length of 1000 words. Training followed the same procedure as used in the provided implementation except the model was allowed to train for five epochs rather than two.

2.3 Variational Autoencoders for Image Analysis

The source paper of this section was Higgins et al.'s 'β-VAE: Learning Basic visual concepts with a Constrained Variational Framework'. The primary goals of this section was to implement a provided PyTorch VAE model, translate the model to TensorFlow, and to experiment with some parameters using the TensorFlow model. The provided PyTorch model was implemented successfully and the reader is referred to the python notebook for the results of this model. The source implementation was translated to the TensorFlow framework while also making use of the Keras library. The TensorFlow model was trained on the MNIST data set of handwritten digits. To analyze the model, a matrix of the latent-space digits was plotted along with a scatter plot of the classes in the latent space. Experimentation was done with TensorFlow model by varying the optimizer and β value.

3 Results

3.1 Convolutional Neural Networks for Image Analysis

The statistical analysis revealed that the CIFAR-10 data set consisted of 60,000 images and targets. The 10 possible classes are: 'Airplane', 'Automobile', 'Bird', 'Cat', 'Deer', 'Dog', 'Frog', 'Horse', 'Ship', 'Truck'. The images are evenly distributed with 6000 images per class. Figure 1 displays a comparison of the mean images and heat maps for the 'Airplane' and 'Automobile' classes. Refer to the notebook for a comparison of all the mean images and heat maps. Across all classes, it's clear that the colors in the mean images are quite similar, however the shapes which present themselves in the heat maps vary across classes. These shapes can give us insights into the filtering layers which the convolutional network

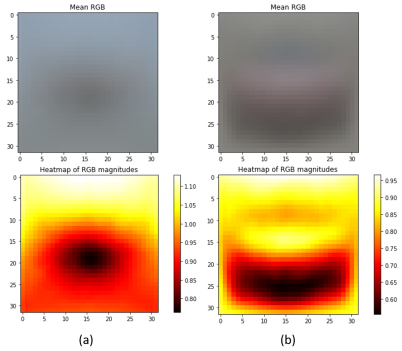

Figure 1. (a) Mean image and heat map for class 0 ('Airplanes') and (b) Mean image and heat map for class 1 ('Automobiles').

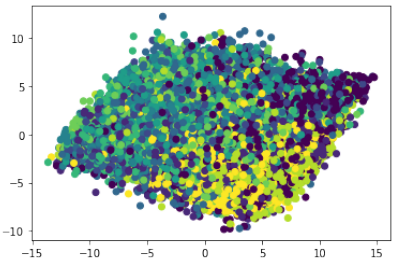

Figure 2. Scatter plot of principal component 1 vs. principal component 2 for CIFAR-10, color denotes class.

develops to classify images. It is shapes such as these which the network learns to extract from images and use to discern between classes.

Figure 2 is a scatter plot of the first and second principal components for each image and a scree plot for the principal components. It's clear that for the CIFAR-10 data set, two principal components is insufficient to visually discriminate between the ten classes. We can gain further insight by refer referring to the scree plot in figure 3. The scree plot compares the proportion of variance which can be accounted for by each principal component. The blue curve represents the proportion of variance accounted for a principal component while the orange curve displays the cumulative proportion of variance. A good rule of thumb is that the set of principal components should cumulatively account for at least 85% of the observed variance. For the CIFAR-10 data set, approximately 40 principal components are needed to reach this threshold. This provides some explanation as to why the first two components were insufficient for plotting distinct classes.

Figure 3 Displays the Training results of training the ResNet models on the CIFAR-100 data set over 30 epochs. As can be seen the models achieved only modest results. The ResNet-50 model achieved 49.69% accuracy on the training data, however the validation accuracy stalled at 33%.

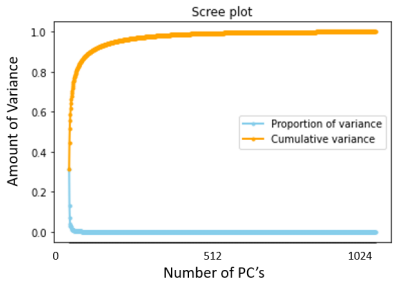

Figure 3. Scree plot for the principal components of the CIFAR-10 data set. As expected, the first two components account for the largest amount of variance compared the other components.

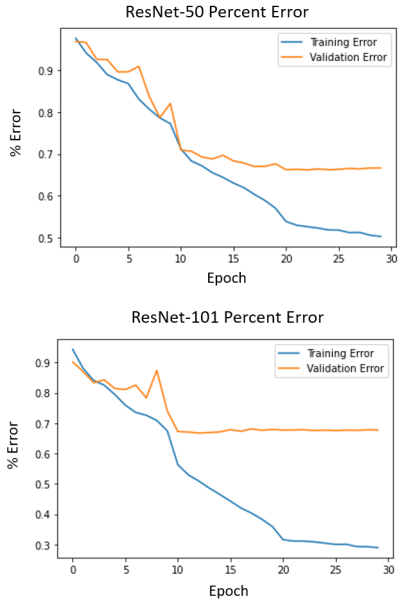

Figure 4. (Top) Training and validation error percentages for ResNet-50 model. (Bottom) Training and validation error percentages for ResNet-101 model.

This suggests that the model was likely starting to overfit the training in the last several epochs. The performance of the ResNet-101 model followed the same trends achieving a training accuracy of 71% but only a validation accuracy of 32%.

Obviously, a comparison of the ResNet models presented here and those presented He et al's paper is difficult. Those in the paper were trained on a different, much larger data set over a number of epochs on the order of 1e4. It truly seems apples to orange to compare the two sets of results. Although, I think by any measure, it can be agreed that the two ResNet models presented here failed to achieve anything beyond mediocre performance on the CIFAR-100 data set.

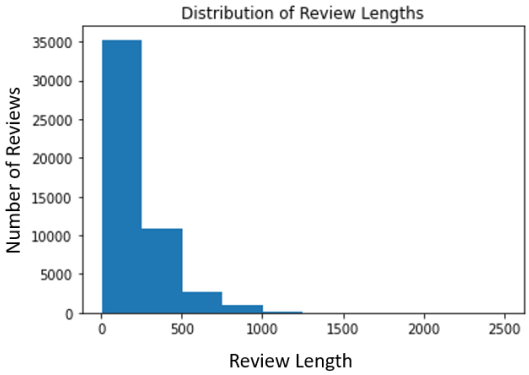

Figure 5. Distribution of movie review lengths in the IMDB data set. Mean value of 235 with standard deviation of 173.

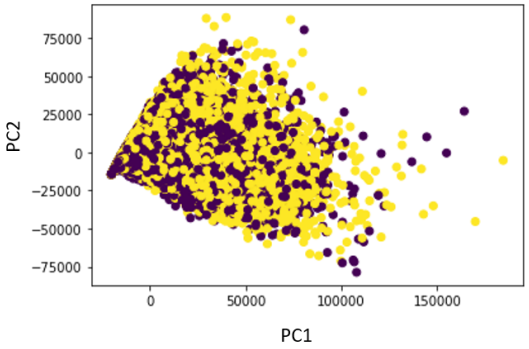

Figure 6. Scatter plot of principal component 1 vs. principal component 2 for IMDB data set, color denotes class.

Perhaps, a different training regime or optimizer could be explored to achieve more impressive results.

3.2 Transformers for Text Analysis

The transformer model provided in the resources was successfully implemented within the notebook and trained on the IMDB data set. The statistical analysis of this data set revealed that it consisted of 50,000 vectors of varying length with 50,000 binary targets. Within the data set, there were 88,585 unique words present and the class distribution was even across the two classes. The most common words were largely what one would expect: 'the', 'and', 'a', 'of', 'to' and other words of the like. Refer to the notebook for a longer list of the most frequently used words. Figure 5 displays a histogram of the distribution of the movie review lengths. As can be seen, most of the reviews tend to be short, however there is a large standard deviation in the review length.

To perform PCA on the data set, the reviews needed to be set to a fixed length. This was achieved by finding the

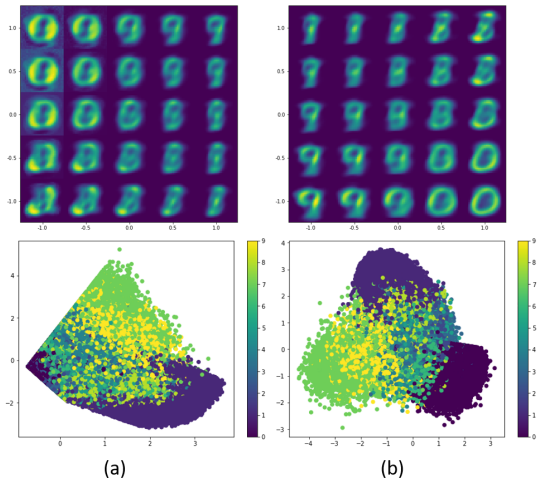

Figure 7. Latent digit representation and latent space classes for (a) Baseline TensorFlow VAE with Adam optimizer and $\beta = 0.5$ and (b) TensorFlow VAE with RMSProp optimizer and $\beta = 0.5$.

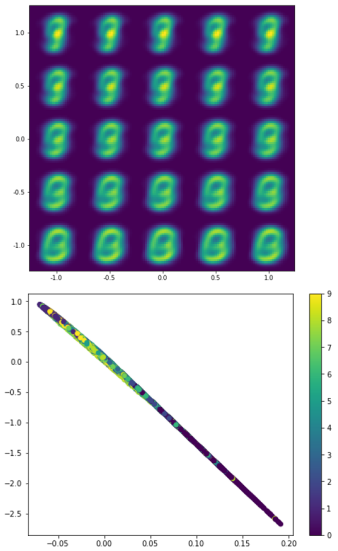

Figure 8. Latent digit representation and latent space classes for VAE with $\beta = 10$ and Adam optimizer.

maximum length review(2494 words) and zero-padding the other reviews to that length. After doing so, PCA was applied to the data set and a scatter plot of the data set using the first two principal components was generated (see Figure 6). Clearly, as was the case in the convolutional network section, two components is insufficient for visually discriminating between the data.

To gain further insight, a scree plot was generated for the principal components just as in the previous section. Refer for the notebook for the plot itself. it revealed that around 350 principal components are needed to account for 85% of the variance. While this is far less than the nearly 2500 components originally present, it is far too many to make useful visualizations. The provided transformer was trained and validated on the Reuters data set as described in the relevant methods section. The model achieved 82% training accuracy and 77% validation accuracy over five epochs. It's likely that these results could be easily improved. this is lower than the accuracy achieved by the model on the IMDB data set in only two epochs, however the Reuters data set is considerably larger and has far more target classes in comparison.

3.3 Variational Autoencoders for Image Analysis

The provided PyTorch implementation of the β-VAE as successfully migrated to the notebook environment. The PyTorch model provided some baseline results for the how the model's outputs change throughout training. Refer to the notebook for these baseline results. The model was translated into a TensorFlow and Keras model which matches the architecture of the PyTorch model. Figure 7 provides a comparison between the latent spaces of the baseline TensorFlow VAE model and a model which was optimized using RMSprop rather than Adam. If the reader were to compare the baseline PyTorch results with the corresponding TensorFlow results, it's clear that the TensorFlow is providing poorer results. The reason for this discrepancy is currently unknown. Despite this discrepancy, the TensorFlow model still provides an architecture to test differing optimizers and β values. comparing the two sets of results in figure 7, it appears that the model optimized with RMSprop achieves a slightly better separation of classes in the latent space. Finally, the model was trained with variety of β values. In the notebook, results can be seen for values of $[0.25, 1, 10]$. In general, it seems that as the value of β increases, the classes become more constricted in the latent space. This can be seen in Figure 8 where $\beta = 10$. The latent space classes have constricted to a single line. If we consider that β is a regularization parameter, it's clear that the model is being over-regularized with $\beta = 10$.

4 Conclusions

Presented above were the methods and results of the three sections comprising the neural network project. First, statistical analysis was performed on the CIFAR-10 data set followed by training two ResNet models on the CIFAR-100 data set. Second, statistical analysis was performed on the IMDB data set from the Keras library and then a transformer was used for a classification task on the Reuters data set. Finally, A TensorFlow VAE was developed and tuned using differing optimizers and β values.