Visual Concept Learning for Hand Written Letter Recognition

Gita Rayung Andadari

gitarayung.andadari@studenti.unipd.it

1. Introduction

Supervised learning has been proven to give a lot of benefit in machine learning world. However, this model needs a lot of data with appropriate label which may not be available for certain casses. Not because it is labor intensive (real human has to assign real label), the amount of data that needs to be labeled increased exponentially especially in the era of big data. For this reason, unsupervised learning becoming more favorable.

In this project, several model will be introduced to overcome hand written recognition task. EMNIST handwritten dataset will be used to train and test the models. Deep Belief Network (DBN) with several layers is being trained. Each DBN model then paired with linear readout. To observe DBN effectiveness, Feed Foward Neural Network (FFNN) and Perceptron model is also being trained and tested. Furthermore, model robustness against noise will also be explored. There are two kind of noise that will be introduce, gaussian noise and adversarial attack.

2. Datasets

In this project, 145,600 samples of handwritten letters (a-z) from EMNIST dataset is being used. Each sample is converted to a 28x28 pixel image format. Furthermore, each class of the 26 classes has the same amount of samples, that is 5600 samples. All samples not necessarily have the same orientation, letters might be rotated, flipped horizontally and vertically. For training and testing purposes, this datasets is then divided into 2 sets :

- Training set consists of 124,800 handwritten letters sample
- Test set consists of 20,800 handwritten letters sample

To reduce the computational cost, each sample is being grayscaled. Additionally, the label of each sample is being transformed to one hot encoding respective to the order of letter. For example, label ‘a’ is equal to label 0 and label ‘z’ is eqal to label 25. Picture 1 shows the example of the handwritten letter of each class.

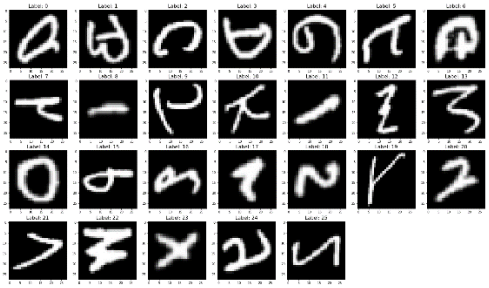


Figure 1. Handwritten Letter Sample of each class

3. Method

3.1 Model Architecture

Restricted Boltzmann Machines (RBM) are stochastic and generative neural networks capable of learning internal representations. RBM composed of multiple layers of latent variables, with connections between the layers but not between units within each layer [1]. RBM can be stacked to compose a model in hope to produce a better accuracy. This kind of model is called Deep Believe Network (DBN). In this project, 3 different layers of DBN will be trained and evaluated for its effectiveness based on the model accuracy. Each RBM layer has 500 units. However, DBN does not necessarily produce an output layer. Therefore, a perceptron will be taylored to each DBN model to produce final prediction.

To learn the effectiveness of DBN, another perceptron will be trained and tested using raw data (not taylored to any DBN model). Furthermore, we want to compare a network that is trained end-to-end to solve a classification task with a simple classifier that solves the same task using representations of input data learned in an unsupervised way. Therefore, a Feed Forward Neural Network with the same structure as the RBM (i.e. one input layer with 784 neurons and one hidden layer 500 neurons) is also being trained with training test and tested on the sets of data that the model has not seen before, the test set.

3.2 Robustness to Noise

Noise in a picture is inevitable in the real dataset. Therefore, a good recognition model should have a good robustness to noise. To check this characteristic, some noise will be injected to the datasets to see how the model perform. There are two kind of noise that will be observed within this project:

- Random Gaussian Noise

This type of noise will mostly affect the background of the image. Picture 2 shows the example of Random Gaussian Noise.

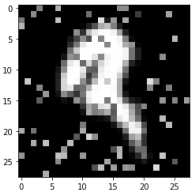


Figure 2. Random Gaussian Noise on Letter "a"

- Adversarial Attack

Unlike the Random Gaussian Noise, Adversarial attack mostly affect the edge of the shape hence makes the data looks very similar to other letter. This action will confuse the model hence causing missclassification. Picture 3 shows the example of Adversarial Attack.

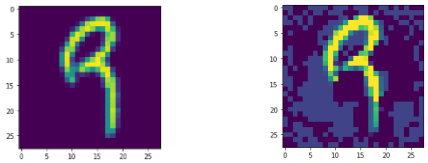


Figure 3. Adversarial Noise on letter "a"

To make adversarial attacks, some modification needs to be done to the input image so that the model cannot correctly classify it anymore. This means that the loss for that specific input has to increase. The loss is usually a function of the input, the model's parameters and the outputs $J(w, x, y)$. The adversarial sample can be written as equation 1.

$$\tilde{x} = x + \epsilon \cdot sign(\nabla_x J(w, x, y)) \qquad (1)$$

3.3 Resisting to Adversarial Attack

Theoritically, any noise introduced will alter model capability to produce a correct prediction. Contrary to popular belief, DBN has data reconstruction capability that allow itself to be more robust to the attack.

To evaluate DBN robustness against adversarial samples, several attack with varried intensities range from 0 to 0.01 was conducted. Then, its accuracy is calculated and plotted to observe the behavior.

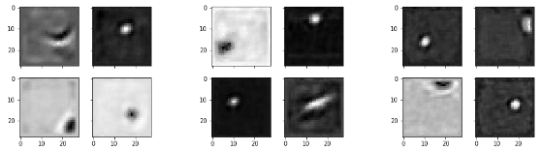
4. Experiments

4.1 Layer Receptive Field in RBM

In every RBM layer, each neurons represent specific characteristic of the input. Figure 4 shows the sample of the receptive field from each RBM layer. The white and black color in the graph represent a strong connection on particular feature in the letters. On the other hand, color gray represent weaker connection. It can be seen that as the layer goes deeper, the neuron has stronger connection implied by the darkens of color (less gray).

The weights in the second and third hidden layer have dimensionality (500,500). To overcome the dimensionality issue, each vectors needs to be projected in a space of dimensionality 784

(28x28). In mathematic terms, this is merely a matrix multiplication of the weights in first,

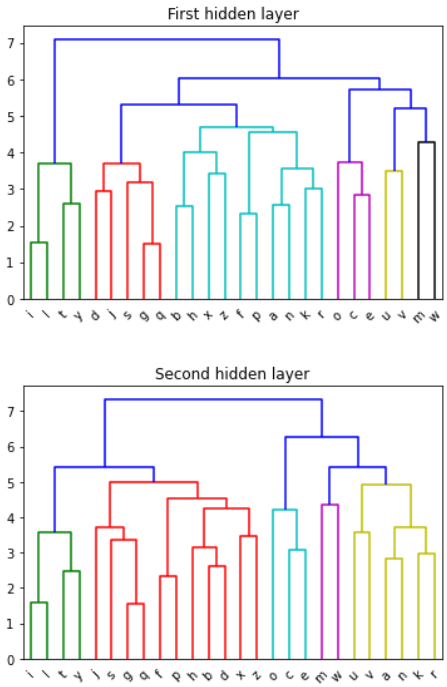


Layer 1 Layer 2 Layer 3

Figure 4. Sample of Receptive Field from each Layer

4.2 Clustering Internal Representation

To observed model abbility to capture similar visual features, mean hidden representation needs to be calculated corresponding to each class. This is done by doing hierarchical clustering of the representations and visually inspecting the result. The similarity between classes then can be concluded by comparing the closeness of mean hidden representation of each class. Figure 5 shows the dendrogram of the class based on each layer. By a visual inspection of the plot, it can be seen that handwritten letters which have similar shape are close together. For example, letter “i” and “l” are next to each other. As the layer grow, the model can capture better letter feature hence put the letter on the better grouping system.



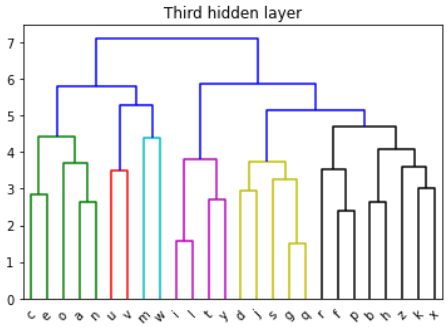


Figure 5. Dendrogram plot of the hierarchical clustering for 1, 2, and 3 layers respectively

4.2 Linear Read Out Performance

Three different linear read out are attached to each layer of DBN. The purpose of this experiment is to determine the effectiveness of each layer. Figure 6 shows the accuracy across various DBN layer. It can be concluded that more layer does improve the accuracy of the model. Especially, in this project a lot of classes are being considered. The model needs to find detail representation that can distinguished one class to another. The feature relationship in the form of weight from each layer is one of the way to boost model abbility.

4.3 Comparing DBN, Raw Perceptron, and FFNN

Figure 6 shows the model accuracy across different models. It can be seen that model that use DBN performs better than the one which does not. The difference between DBN 3 layers and raw perceptron is as high as 10%. Another comparison, given the similarity of model architecture, FFNN performs way worse than the DBN model. The ability of DBN in finding similarity between inputs really increase the model ability to deliver correct prediction.

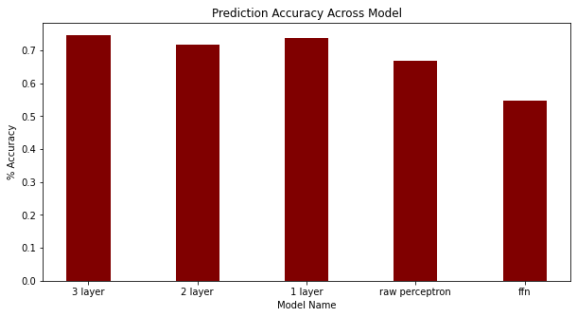


Figure 6. Prediction Accuracy Across Models

4.6 Robustness to Noise

Gaussian noise is being introduced to the test set to see model abbility dealing with noise. Figure 7 shows model accuracy on the noisy test set. It can be seen that DBN 3 layers works best on less noisy set. However, as greater noise is being introduced, the DBN performance decrease drastically. On the other hand, FFNN and Raw Perceptron are more resistance to the noise. At 0.6 level, FFNN delivered almost three times better than DBN 3 layer performance.

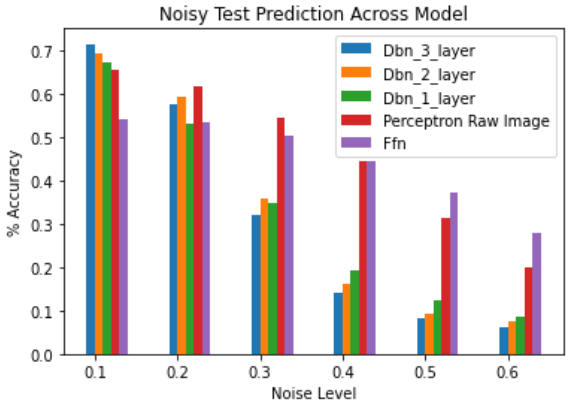


Figure 7. Noisy Test Prediction Across Model

4.7 Resisting to Adversarial Attack

Data reconstruction abbility is activated to undergo this experiment. Figure 8 on the right shows the result of reconstructed image. With just one step, the picture looks similar to original image. Therefore, this way model can give prediction better.

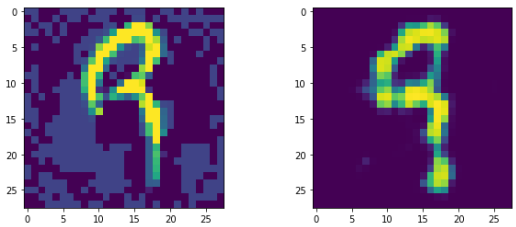


Figure 8. Left: Perturbed Image. Right: Reconstruction Image in 1 Step

4.8 Effect of the Parameter ϵ

Sensitivity test on epsilon parameter was conducted across model. The epsilon used in this experiment is [0, .0001, .0005, .001, .005, 0.01]. Figure 9 shows the sensitivity test result across various model. It can be said that DBN reconstruction steps does help the model to deliver better prediction in the end. The straight line for DBN is happening because the discrepancy among epislon ranges is small. DBN is not too sensitive with small change of epsilon. Whereas, even though FFNN and Raw Perceptron are more robust to Gaussian noise, adversarial attack affect FFNN and Raw Perceptron model so much. With small epsilon, it decreases the model capability by much.

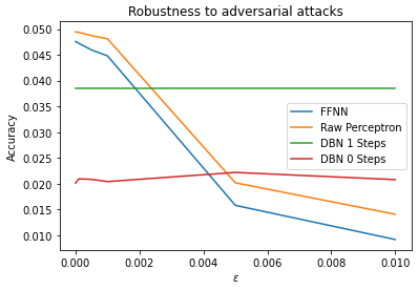


Figure 9. Epsilon effect on Model Accuracy

4.9 Error Analysis

The models missclassified some of the letters. Figure 10 shows the sample of misclassified letters. The training and test data consisted of data with various orientation. Looking at figure 10, it is predicted as letter “c” when it is actually a letter “i”. At a glance, it does look like a letter c because of the curve in the body. If this is written in the middle of some word, human can easily miss interpret this letters as c.

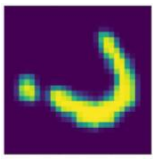


Figure 10. Sample of misclassified letter. Predicted ‘c’, actual ‘i’

Furthermore, figure 11 shows the confusion matrix from 3 layers DBN model. It helped us to identify which classes that contribute to most miss-classification. The model is very sensitive to letter “s”. A lot of instances marked as “s” even though they are not. One of possible explanations

for the missclassification is because the dataset consist of upper-case and lower-case letter mixed together in the same class. Some of the letters may have similar shape for upper and lower case, such as “o” and “O”. However, “g” and “G” are very different. “G” is closer to shape “c” other than its lower case mode. For future improvement in model accuracy, dataset can be divided into more classes so the model can have a better feature representation within class.

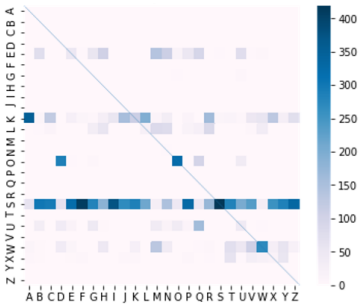


Figure 11. Confusion Matrix. X axis denoted the actual representation, y axis denoted the predicted representation

5. Conclusion

Hierarchical Generative Model such as Deep Belief Network is being used for the handwritten letters recognition. The RBM layers proven to be improving the overal DBN + Linear Read Out performance. However, this model originally does not perform well if the test image has noise. Feed forward neural network and Raw Perceptron are more robust to gausian noise naturally. However, once the model attacked with adversarial noise, FFNN and Raw perceptron accuracy drop significantly. DBN has generative abbility that can recreate image with noise to be more clear wich helps the model deliver better prediction. This is one of the way to overcome the adversarial noise with DBN.

6. References

[1] Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters. Retrieved from http://arxiv.org/abs/1702.05373

[2] Inkawhich, N. Adversarial Example Generation. 2021. https://pytorch.org/tutorials/beginner/fgsm_tutorial.html

[3] Sharma, A. Restricted Boltzman Machines – Simplified. 2018. https://towardsdatascience.com/restricted-boltzmann-machines-simplified-eab1e5878976