Transfer Learning of Latin and Greek Characters in Deep Neural Networks using Unsupervised Feature Learning

Submitted in Partial Fulfillment
of the Requirements for SYDE 675

Dylan Drover, MASc. Candidate

Faculty of Engineering
Department of Systems Design Engineering

April 25, 2015.
Course Instructor: Professor Daniel Stashuk

Abstract

The following paper attempts to demonstrate and test the effectiveness of "transfer learning" in artificial neural networks. The theory, that is associated to human cognition, is that once a skill is learned, that knowledge facilitates the learning of similar skills. Transfer of knowledge is attempted by using unsupervised neural networks to extract features from two different alphabet characters sets. These features are then used for supervised training of a classification network. This is done both between the two character data sets as well as extended to different sized input images of the characters sets. In this case the features are scaled to achieve transfer learning for different sized inputs. Results show using this technique shows equal results to non transfer learning and in some cases might improve performance. Training time advantages are also inherent in this technique as it suggests only one set of features may need to be learned to be extended to multiple data sets.

Contents

1 Introduction

Artificial neural networks (ANN) have had a resurgence in the popularity in recent years. This can, in part, be attributed to both improvements in computational capability (parallel computation) as well as new learning methods and network architectures. The ancestor of the modern neural network was the perceptron, created by Rosenblatt in 1957[1]. Perceptrons and thus connectionist models fell out of vogue with artificial intelligence researchers after their limitations were analysed by Minsky *et al* [2]. It was not until the 80's and 90's that new algorithms and architectures renewed interest [3, 4, 5]. These advancements in hardware and software have led to ANNs providing some of the largest successes and best results in the field of pattern recognition and machine learning. Newer architectures such as Hopfield Networks and Restricted Boltzmann Machines have the ability to do rich learning without the need for labelled data sets. These new unsupervised methods take advantage of what are called "generative models" and "feature extraction".

Problems still exist for neural networks, however. One major hurdle is that the aforementioned unsupervised deep neural networks need very large data sets as well as ample time to be trained. Another problem is that these neural nets tend to have a myopic view of the data they are modelling, that is to say, it is difficult for them to generalize well.

Some of these problems may be solved by emulating the brain's ability to apply previously taught knowledge to unknown problems. This is what will be referred to as "transfer learning" and it is something that occurs naturally in human cognition. It can be described as the transferral of knowledge from one field or skill set to another related but different problem. A method for transfer learning will be explored for this project. It will be applied to learning one alphabet and then using the learned features to improve learning for another language's alphabet.

2 | Background Review

Generative Neural Networks

The overarching idea for a generative neural network (GNN) is to recreate the input vector (of dimension n) that is passed into the GNN through a fewer number of neurons (m) such that $m << n$. The neurons at the input are labelled as V and are called visible units whereas the other neurons are appropriately called hidden neurons, labelled H. The GNN that will be used is called the Restricted Boltzmann Machine (RBM) [6]. This neural network works as a bipartite graph (as seen in Figure 2.1) that is given a training example which it attempts to reproduce. The weights are symmetrical, that is, information can be passed from the input layer to the hidden layer and vice versa.

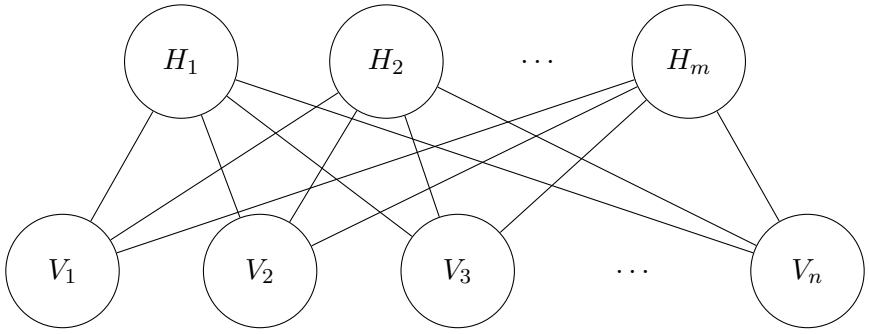

Figure 2.1: Visual layer and hidden layer of a RBM

The hidden layer of neurons take a real valued input $[0, 1]$ and will return a binary output, 1 or 0. The learning algorithm employs stochastic learning for new weights. This randomness helps remedy the problem of getting trapped in what are called local minima. RBMs are also energy based neural networks. This means that their optimal state can be achieved through minimizing the network's energy function (as seen in Equation 1, where v is a given visible vector, h is a hidden

neuron vector, b and a are biases and w is a weight vector) [7].

$$E(v,h) = -\sum_i a_i v_i - \sum b_i h_i - \sum \sum v_i w_{i,j} h_j \tag{1}$$

This energy function helps define the probability of a input vector based on the states of the hidden neurons as seen in Equation 2. These RBM act as generative feature extractors. That is that, they will attempt to duplicate the input provided to them through the use of extracted features. They do this through their learning method which attempts to maximize the probability of the input vector that is represented through the firing of the hidden layer.

$$P(v) = \sum_h P(v,h) = \sum_h \frac{e^{-E(v,h)}}{\sum_v e^{-E(v,h)}} \tag{2}$$

Due to the structure of the Restricted Boltzmann Machine, given a configuration of hidden units, the probability of a given input vector v can be found with the following equation.

$$P(v|h) = \prod_{i=1}^{m} P(v_i|h) \tag{3}$$

To maximize the likelihood of all the input vectors, a method called contrastive divergence (CD) [8] is used to train the weights. The details of CD are well beyond the scope of the project. The important part to grasp is that the weights are updated by improving the approximation of the input vector x by the firing of the hidden neuron pattern y. The weights will be altered in proportion to the expected value ($\langle \cdot \rangle$) of the product of the vectors minus what the hidden unit activations "think" the product of the vectors should be. What is interesting is that the learning equations end up looking like theoretical models of how the human brain might work; this is called Hebbian Learning [9]. Where neurons that have causal relationships will grow stronger connections. To prevent this from blowing up in the RBM, there is a negative component added as seen in Equation 4.

$$\Delta w_{ij} \propto \langle y_i x_j \rangle_{data} - \langle y_i x_j \rangle_{model} \tag{4}$$

Deep Neural Networks and Pre-Training

Because the Restricted Boltzmann Machine, with $m << n$, creates a smaller dimensional representation of the input data, they act as excellent feature extractors. This can allow for the stacking of these RBMs into layers of feature extractors. That is, the input layer creates a feature representation of the data, then the layer above can take these extracted features and create further abstractions and so on.

State of the Art

Currently taking the abilities of one ANN and using it to speed up the learning of another ANN that is trying to solve a similar problem is a growing field in machine intelligence [10, 11]. This ability to use previously mastered skills or knowledge of features in similar data sets will allow us to improve learning and generalization. There is already precedent for learning one alphabet and using that knowledge to improve the learning of a new character set [12]. Ciresan, Meier and Schmidhuber have had success in transferring knowledge from one character set to another (English and Chinese) [13].

3 | Methodology

Data

For the successful transfer of pattern recognition ability from one trained neural network to another, there needs to be some relation between the two problems' data sets. This project uses capital letters of the Greek and Latin alphabets. The basis for choosing these two sets of characters was their observational similarity. Both share similar general shapes and number of high level features (number of lines making up the letters *etc*). To assure that the two data sets did share adequate characteristics to merit transfer learning, a distinct set of 10 characters was taken from each alphabet and using a structural similarity index (SSIM) each character was compared to each other letter in its own alphabet set and then to the other alphabet. The SSIM measures image similarity and ranges from -1 to 1, with 1 being complete similarity. The results are shown in Table 3.1.

Compared Alphabet	μ SSIM	σ SSIM
Latin-Latin	0.3478	0.2503
Greek-Greek	0.2961	0.2736
Latin-Greek	0.2692	0.1378

Table 3.1: Inter-character similarity using SSIM

As can be seen, the correlation between characters within their own alphabets is comparable to the similarities between the two alphabets. This shows that there is some feature similarity between the two. Figure 3.1 shows the two sets of characters from each alphabet. Each character is treated either as a 70×70 pixel image or a scaled down 24×24 pixel image.

Figure 3.1: Greek and Latin character data sets, respectively

Feature Extraction and Transferral of Features

Two experimental set ups are used for transfer learning, both using a single hidden later Restricted Boltzmann Machine. The first configuration uses input images that are 24×24 pixels. The value of a pixel is a real value between 0 and 1. Images are reshaped into an input vector with a length of 576 values (24^2). Therefore there are 576 visible neurons (input neurons). The hidden layer is composed of 36 neurons with sigmoid activation functions and binary outputs. 36 was heuristically the chosen as it limited training time, while providing adequate accuracy. The number of hidden and visible neurons also satisfies the rough metric for feature extraction which is $m << n$ as noted above.

Each dataset for training, Greek and Latin, was 10000 characters, chosen randomly from the original 10. This data was then split 90% for training and 10% for testing. The RBM was trained using a batch size of 100 samples and this was done over 10 epochs (training was done 10 times). After training was done with either the Greek or Latin data set, the RBM was saved and used as the initial weights for the feed forward neural network which is discussed in the Classification section.

Scaling and Transferral of Extracted Features

The second experimental set up was created to examine the effects of transferring the features learned from the small images (24×24) to larger images (70×70). The RBMs were set up similarly to the previous section, however the RBMs were only trained on the Greek alphabet. A RBM was trained for the 24×24 then scaled up to work for inputs of 70×70. Each of the 36 neurons had 576 or 24^2 weights, one from each input pixel. These weights were arranged in a 24×24 square matrix and scaled by the appropriate factor and then reshaped into weight vectors for each neuron. There

now existed 36 neurons with 4900 weights, corresponding to each pixel in a 70×70 pixel image.

Classification

The classification of each character was performed by associating each image of a letter to a unique vector that was associated with that character. This goal was achieved using a feed forward neural network (FFNN) using sigmoid neurons. This neural network was trained using the well known supervised back-propagation algorithm. Each training image (character) had an associated classification vector that corresponded to its place in the original sequence seen in Figure 3.1.

The FFNN has an architecture of $576 \rightarrow 36 \rightarrow 10$ or $4900 \rightarrow 36 \rightarrow 10$. The architecture can be seen in Figure 3.2. The input layer corresponds to the number of pixels, the hidden layer corresponds to the number of hidden neurons from the RBM and the output layer corresponds to the number of classes that exist for the problem (10 characters). The advantage this FFNN has in comparison to a traditional FFNN is that the weights from the RBM now become the weights in between the input layer and the hidden layer in the FFNN. This allows for much faster learning and this is where the transfer learning actually takes place. The RBM weights trained on the Greek or Latin alphabets will be used to do classification on the other alphabet. The weights will be taken from the RBM from the other alphabet and the FFNN will then be trained using these weights to do classification.

With this system, one set of features can be learned and then a FFNN can be created for either the Latin or the Greek alphabets and have good error rates. This limits training time, as well as demonstrates the proof of concept of transfer learning.

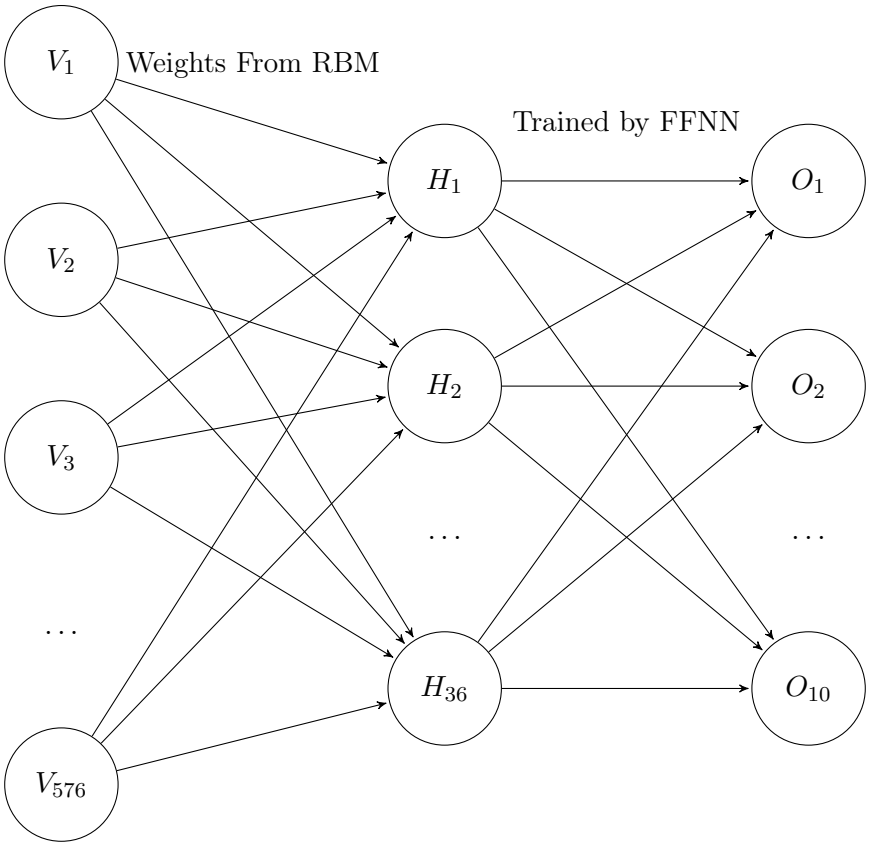

Figure 3.2: Architecture of Feed Forward Neural Network for Classification

4 | Results

Intra-Alphabet Transfer Learning

In the following section the results of the transfer learning will be discussed. Figures 4.1 and 4.2 each show two examples of separate RBMs that were trained to recreate their respective character sets. Its is very apparent from Figure 4.1 that these are the weights used to classify the Greek letter. Each separate square in Figure 4.1 and 4.2 corresponds to a weight matrix of the connections of the 576 (24×24) input neurons to each of the 36 hidden neurons. A lighter colour means a stronger response from the neuron for that part of the input vector. The weights for of the 36 neurons have actually taken on the appearance of the characters they are supposed to generate. These weights show how when the hidden units are asked to create a Ψ (psi), for the first RBM on the left, the neuron most likely firing is the 15th neuron (counting from top-left to right). Similarly in the second RBM on the right it can be seen that neurons 19 and 21 will certainly fire when the RBM is attempting to create an Ω (omega). Extending this precedent to Figure 4.2, it is possible to see the numerous patterns that have emerged in attempts to form the Latin characters that the RBMs were trained on.

It is also interesting to note that some of the neurons are more ambiguous and have multiple characteristics of different characters. These weights store features from each character set and have been copied to be used as the beginning weights for the FFNN as explained in the Methodology section. It can be seen that the RBM produces what appear to be salient features for the 10 characters of each data set. These and similar weights will be shown to be effective at implementing transfer learning.

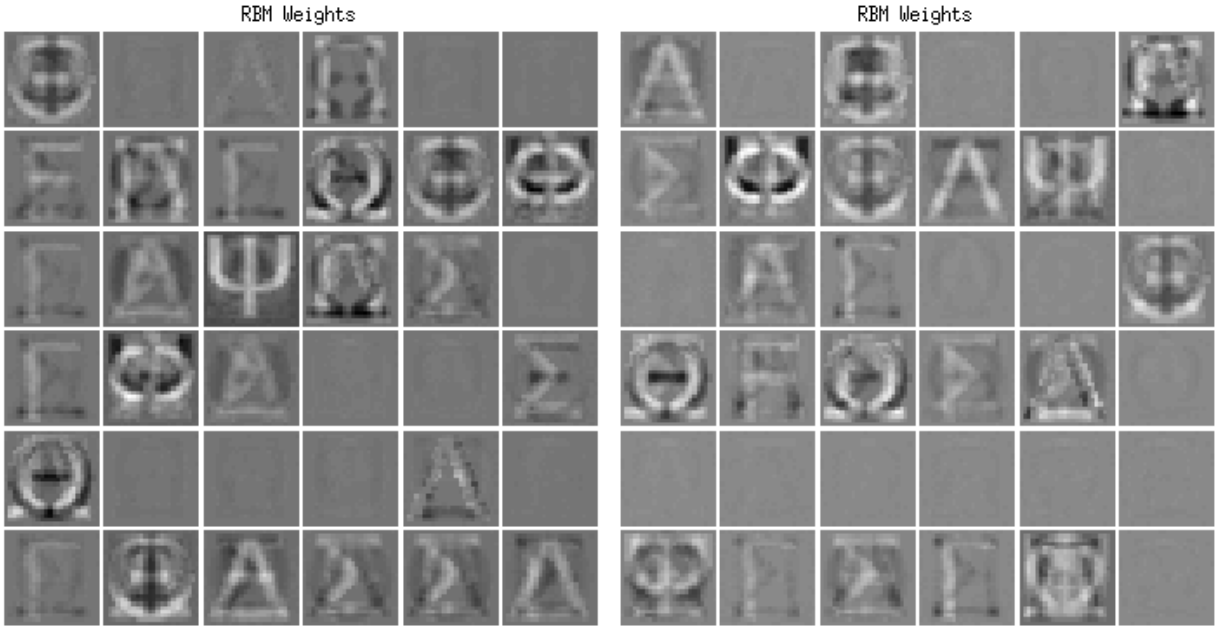

Figure 4.1: Two separate weight representations of 36 neuron, 24×24 pixel, RBMs trained on Greek characters

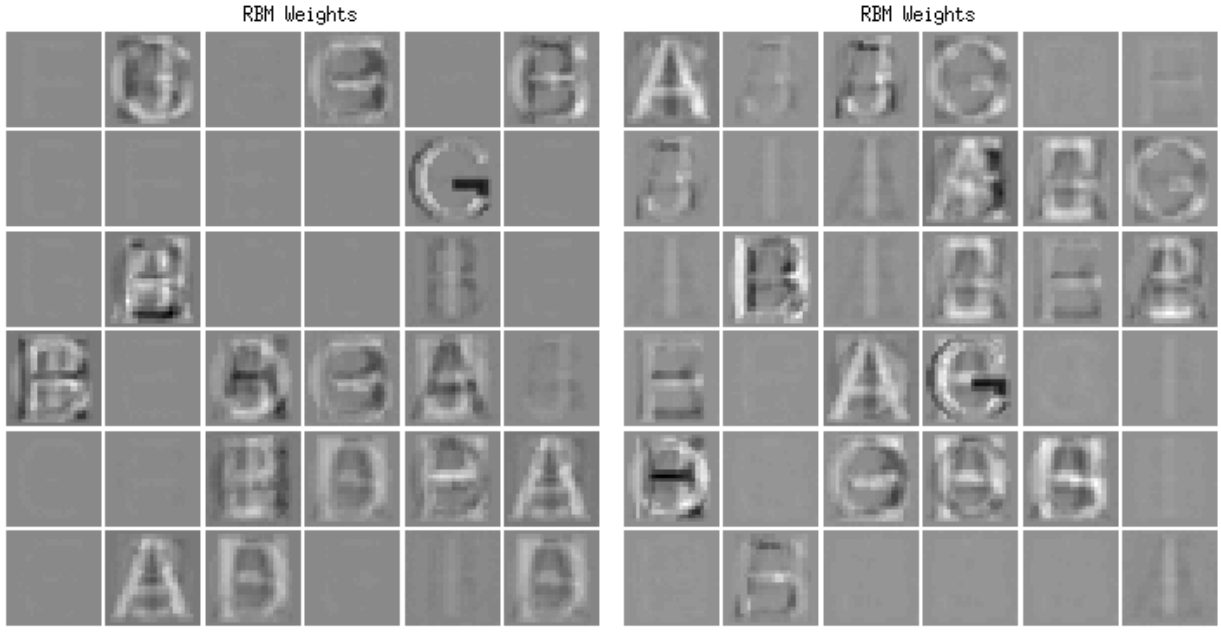

Figure 4.2: Two separate weight representations of 36 neuron, 24×24 pixel, RBMs trained on Latin characters

Figure 4.3 shows an example of the Greek characters that were reconstructed by a RBM. It can be seen that these representations generally capture all of the characters rather successfully.

Figure 4.3: Reconstructed Greek letters using a 24×24 pixel based Restricted Boltzmann Machine

Using the methodology detailed above, the following results for transfer learning were tabulated. Each character set's RBM weights are used for a classification neural network for itself and then for the other data set. The statistics provided were calculated over fifty examples. It can be noted from the standard deviation of each test case that there was a wide breadth of error values. This can happen in neural nets due to the randomized nature of initial weights though generally it can be seen that even large errors would not exceed 20% and sometimes would reach 0%.

A surprising and promising result for transfer learning is the error rate of the transfer learning case between Greek to Latin characters actually outperformed the Latin to Latin case, 7.3% to 11.91% respectively. The result for the Latin to Greek transfer learning case was also slightly better than the Latin to Latin case. At the current time, it is unclear to what this can be definitively attributed. However, it is reasonable to posit that the transfer learning aids in generalization. When the weights are transferred to the FFNN, they have less bias towards the data set than one that is already trained for the data set, potentially leading to fewer local minima in the weight space. The learning time for all the results was similar.

Compared Alphabet	μ of Error	σ of Error	μ of FFNN Training Time (s)
Latin $\rightarrow$ Latin	0.1191	0.0995	2.2578
Greek $\rightarrow$ Greek	0.0544	0.0645	2.5805
Greek $\rightarrow$ Latin	0.0730	0.0765	2.3048
Latin $\rightarrow$ Greek	0.1075	0.0925	2.2872

Table 4.1: Classification statistics (using FFNN) for 24×24 pixel Restricted Boltzmann Machine using transfer learning

Scaled Features for Transfer Learning

The scaling of the learned features was more successful than anticipated, with the scaled RBM weights performing better than a RBM that was trained on the larger sized images. As noted in the methodology section, two RBMs were trained, one for 24×24 images and one for 70×70 images. In

Figure 4.4, an enlarged detail of the weights can be seen with the scaled weights looking smoother than the non-scaled version. The scaling function that was used was intended for use on images and thus, the algorithm inferred values for the newly created pixels. This led to an apparent smoothing of the weights along with preserving the general characteristics of the extracted features.

Figure 4.5 displays the Restricted Boltzmann Machine which was trained with the larger images (70×70). This RBM was trained with the same parameters as the previous one used on the 24×24 images. From observation it is apparent that the representation of the various characters is less distributed among the neurons than in the previous cases with the smaller images. This may have led to the poorer performance of these extracted features as seen in Table 4.2.

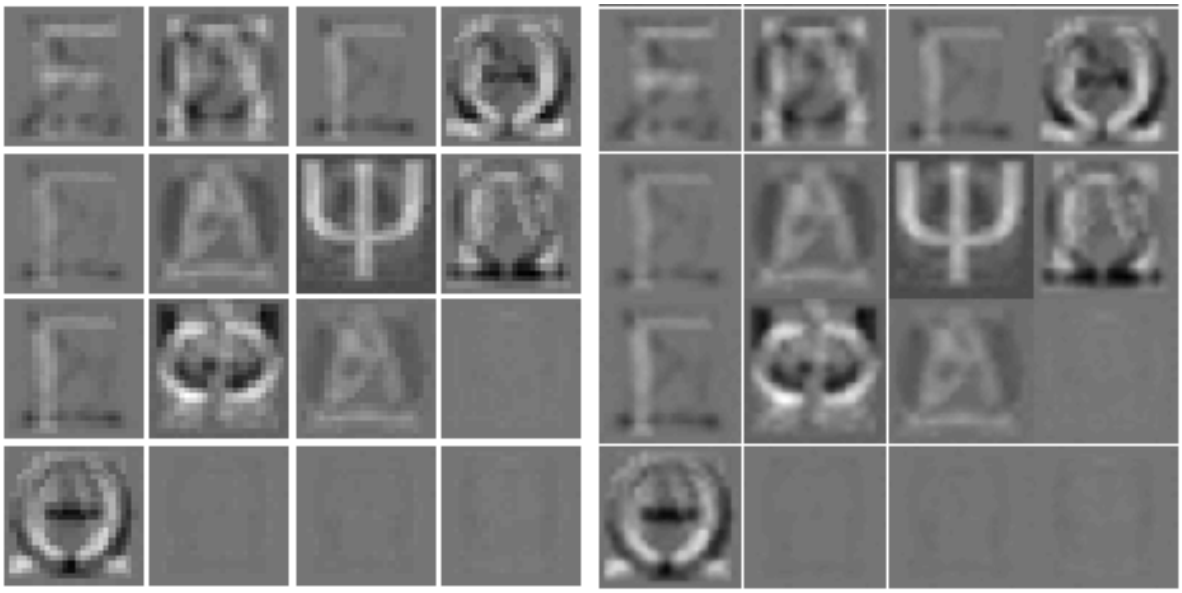

Figure 4.4: Comparison of non-scaled (24×24) versus scaled (70×70) weight representations

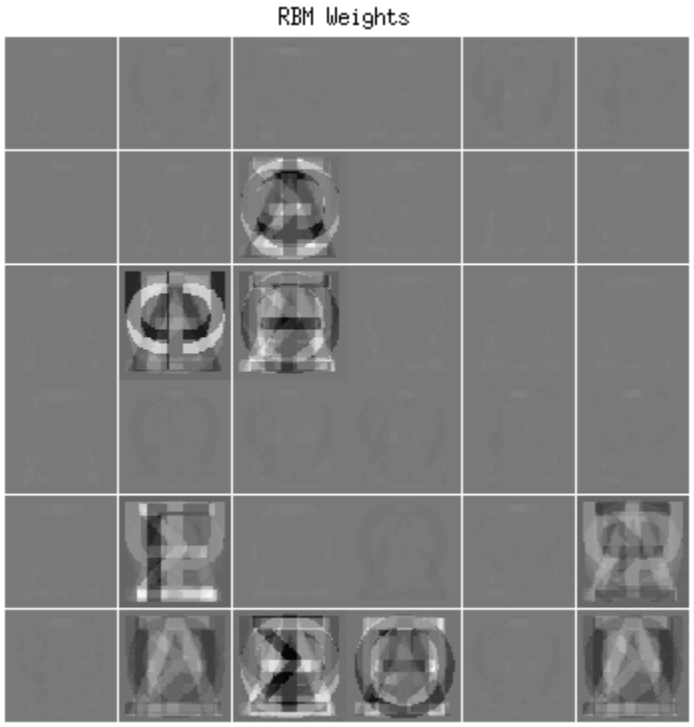

Figure 4.5: Weight representations of 36 neuron, 70×70 pixel, RBMs trained on Greek characters

It can be seen from the results in Table 4.2 that scaling the weights from the smaller images actually led to better error results for classification than using the larger images to train the RBM. Transfer learning was similarly tested with the scaled weights of the Greek character set being used for the FFNN classifier with 70×70 pixel Latin characters. The result was also good in comparison to both the results of the previous table as well as compared to the other method for classifying the characters in Table 4.2. The added advantage of better classification error is that the training of the RBM for the smaller images and the scaling took much less time than it took to train the larger RBM as is noted in the Table below. The results were averaged over 50 iterations.

The reasoning for the improvement in accuracy can be accounted for partially by looking at the example of features that were extracted in Figure 4.5 for the 70×70 pixel based RBM. As noted before, the distribution of the features is limited to a smaller number of neurons than that of the previous RBMs. Consequently, the during classification of the characters, there would be less distinction between neuron sets firing, and therefore characters would have similar firing patterns leading to poorer performance of the RBM trained with the larger images. It is possible to see in Figures 4.1 and 4.2 (RBMs trained with 24×24 images) that the representations of the characters was more distributed across all 36 neurons.

Compared Alphabet	μ of Error	σ of Error	FFNN Train Time (s)	RBM Train Time (s)
Greek(24x24) $\rightarrow$ Greek(70x70)	0.0906	0.0844	15.9220	5.8270
Greek(70x70) $\rightarrow$ Greek(70x70)	0.1870	0.0902	16.1084	46.430
Greek(24x24) $\rightarrow$ Latin(70x70)	0.1176	0.0755	15.4984	5.8270
Greek(70x70) $\rightarrow$ Latin(70x70)	0.1585	0.0707	15.3024	46.430

Table 4.2: Classification statistics (using FFNN) for non-scaled and scaled weight transfer learning

5 | Conclusion and Future Work

The project put forth a method for testing and demonstrating the efficacy of transfer learning using unsupervised learning techniques. A single layer Restricted Boltzmann Machine was trained without supervision to generate the Greek and Latin character sets. The weights generated by the Restricted Boltzmann Machine were then used in a supervised feed forward neural network to do classification of the other data set. The results showed that the training time of the FFNN was not significantly increased or decreased when using the opposing character set compared to the native set. Results also showed that successful transfer learning had taken place as the classification error was similar if not better than when the Greek or Latin-based weights were used for classification of the other data set.

Larger images were also classified using scaled weights that were trained using smaller images. Similarly promising results were achieved with this method. The scaled weights actually outperformed the weights trained on the larger images. This was paired with the shorter training time of the smaller image RBM.

Further work on this topic would include more detailed analysis of why the learning works better for one alphabet than the other. Analysis of whether transfer learning prevents over-fitting and improves generalization would also be pursued. This project focused on visual inputs but it would also be interesting to pursue more abstract data and test whether similar results could be achieved for data sets that are traditionally modelled in similar fashions.

References

[1] Rosenblatt, Frank. The perceptron, a perceiving and recognizing automaton Project Para. Cornell Aeronautical Laboratory, 1957.

[2] Minsky, Marvin, and Seymour Papert. "Perceptron: an introduction to computational geometry." The MIT Press, Cambridge, expanded edition 19 (1969): 88.

[3] Hopfield, John J. "Neural networks and physical systems with emergent collective computational abilities." Proceedings of the national academy of sciences 79.8 (1982): 2554-2558.

[4] Hinton, Geoffrey E., et al. "The" wake-sleep" algorithm for unsupervised neural networks." Science 268.5214 (1995): 1158-1161.

[5] Hinton, Geoffrey E., and Zoubin Ghahramani. "Generative models for discovering sparse distributed representations." Philosophical Transactions of the Royal Society B: Biological Sciences 352.1358 (1997): 1177-1190.

[6] Salakhutdinov, Ruslan, Andriy Mnih, and Geoffrey Hinton. "Restricted Boltzmann machines for collaborative filtering." Proceedings of the 24th international conference on Machine learning. ACM, 2007.

[7] Coughlin, James P., and Robert H. Baran. Neural computation in hopfield networks and boltzmann machines. University of Delaware Press, 1995.

[8] Hinton, Geoffrey. "Training products of experts by minimizing contrastive divergence." Neural computation 14.8 (2002): 1771-1800.

[9] Hebb, D.O. "The Organization of Behavior". New York: Wiley & Sons, 1949.

[10] Oquab, Maxime, et al. "Learning and transferring mid-level image representations using convolutional neural networks." Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on. IEEE, 2014.

[11] Bengio, Yoshua. "Deep learning of representations for unsupervised and transfer learning." Unsupervised and Transfer Learning Challenges in Machine Learning, Volume 7 (2012): 19.

[12] Yosinski, Jason, et al. "How transferable are features in deep neural networks?." Advances in Neural Information Processing Systems. 2014.

[13] D. Ciresan, U. Meier, J. Schmidhuber, "Transfer Learning for Latin and Chinese Characters with Deep Neural Networks." The 2012 International Joint Conference on Neural Networks (IJCNN), June 2012.