Handwritten data extraction from filled forms using Restricted Boltzmann Machines

Abstract

Restricted Boltzmann Machines (RBM's) are fundamentally a type of recurrent neural network and function as an associative memory by being able to remember the correlation between the input vector dimensions. In this project, we exploit this property of an RBM in order to extract handwritten data from a filled-up form. Several other experiments are also carried out in order to gain insights into the working of an RBM, the learning algorithm and the weights corresponding to hidden units. Two layers of RBM's are then stacked in order to form a Deep Belief Net, which performs better than a single layer of RBM for handwritten data extraction.

Introduction

Theory related to Restricted Boltzmann Machines:

Restricted Boltzmann Machines (RBM's) are a basically a type of recurrent neural network, whose binary activations depend on the neighbors they are connected to. RBM's have a set of visible and hidden nodes, and they are able to function as a stochastic RNN, by incorporating these additional hidden units that are different from the inputs. For an image, the pixels correspond to the 'visible' units because their states are observed and the feature detectors correspond to the 'hidden' units [1].

RBM's are used in unsupervised learning in order to learn the underlying probability distribution of the data. While training an RBM, the goal is to minimize system energy, or maximize the likelihood of the observations. The input patterns used while training the weights are statistically regular. In the training phase, given multiple observations of data, the RBM learns, in its weights, the connections or correlation between the input vector dimensions. Given any new input, the correlated aspects of the new data are retained and are visible at the output, whereas the component of data that is uncorrelated does not appear at the output. Thus the RBM's function as associative memory [2] [3].

The following equations show that the gradient of the log probability (used in the learning rule) turns out to be an inner product of the visible layer states and the hidden layer states. This makes the learning rule simple.

The energy of the joint configuration of the hidden and visible units is [1]:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i \in \text{visible}} a_i v_i - \sum_{j \in \text{hidden}} b_j h_j - \sum_{i,j} v_i h_j w_{ij}$$

v_i, h_j are the binary states of visible unit i and hidden unit j, a_i, b_j are their biases and w_{ij} is the weight between them. The network assigns a probability to every possible pair of hidden and visible units:

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v},\mathbf{h})}$$

where Z is the partition function given by:

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$$

The probability that the network assigns to a visible vector v, is given by summing over all pairs of hidden and visible vectors:

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$$

The derivative of the log probability is given by:

$$\frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}$$

The learning rule for performing stochastic steepest ascent is given by:

$$\Delta w_{ij} = \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model})$$

Hence we see that the correlation in data are seen as inner products in the learning rule, which is constantly updated to obtain the best correlations or connections in the data.

Unlike a Boltzmann Machine, in an RBM there are connections between layers of hidden and visible variables or nodes and not between two variables or nodes of the same layer. In probabilistic terms, this can be interpreted as the hidden variables are independent given the state of the visible variables and vice versa. This makes it easier to calculate the marginal distribution of the visible variables. And we can easily get an unbiased sample of the data given the hidden units and vice versa.

Given a randomly selected training image, v, the binary state, h_j, of each hidden unit, j, is set to 1 with probability:

$$p(h_j = 1 \mid \mathbf{v}) = \sigma(b_j + \sum_i v_i w_{ij})$$

where σ(x) is the logistic sigmoid function 1/(1 + exp(−x)).

($v_i h_j$) is then an unbiased sample.

The assumption of conditional independence as seen above, provides a data-driven selection of the correlation structure (network connections), which is equivalent to the clique (or neighborhood connectivity) system in a Markov Random Field [2].

RBM's help overcome modelling problems associated with high-dimensional input. Instead of having to learn the full covariance matrix (for example in a multi-variate Gaussian we would need to estimate the covariance matrix which involves a lot of parameters, not having enough data could render the covariance matrix underdetermined and not full rank), the RBM is able to circumvent this problem of having too many parameters related to the dimensions to estimate, since it is based on a latent variable model. In the latent variable model, conditional independence is assumed (the data are conditionally independent given hidden nodes, and the hidden nodes are independent given the data) and this makes modelling the input data much simpler. The RBM learns the weights which represent the precision matrix itself. Hence the problem of an ill–conditioned covariance matrix whose inverse may not exist is overcome. Hence not having a lot of data for training does not cause too much of a problem in RBM's.

The way we use Restricted Boltzmann Machines:

Here we apply the correlation properties of an RBM along with the latent variable model for dimensionality reduction, to extract handwritten data from forms.

The idea is that the pixels of a form show correlation, especially in the case of a fixed form. When we train an RBM with these clean forms, it learns the correlation between pixels. Later, when a form with handwritten data is given as input to the RBM, it reconstructs the output based on what it has learnt, which is the clean form itself, leaving out the handwritten data which is the uncorrelated part.

The output form can then be subtracted from the inputted handwritten form in order to obtain handwritten data. The dataset is expected to be noisy due to noise introduced during scanning of the handwritten forms and other non-linear aspects including normalization during the digitizing process. This leads to differences in the members of a dataset although they might be coming from the same form. This noise present in the dataset leads to slight distortion which is seen in the learned weights of the RBM (that do not exactly correspond to the form). This leads to an error in the reconstruction, which can affect the extraction of handwritten data. In order to enhance the output, image processing techniques are applied.

The pixels of the input image are represented by the visible nodes of the RBM, and the hidden nodes model the dependencies between the visible nodes (pixels). The hidden units are what make the modelling capacity of the RBM more powerful. By using an RBM, we try to learn the underlying probability distribution of the form in the form of correlation between its input pixels.

Dataset

For the purposes of this project, we start with a dataset consisting of twenty images, all of the same form scanned multiple times, hence introducing natural variations due to scanning in the dataset itself. In further experiments, it is possible to increase the dataset size by either scanning multiple forms or just augmenting the current dataset by introducing noise in the images. This might also help the RBM to become more robust to any new images or distortions in the dataset.
Below is an example form (one of the 20 images in the dataset):

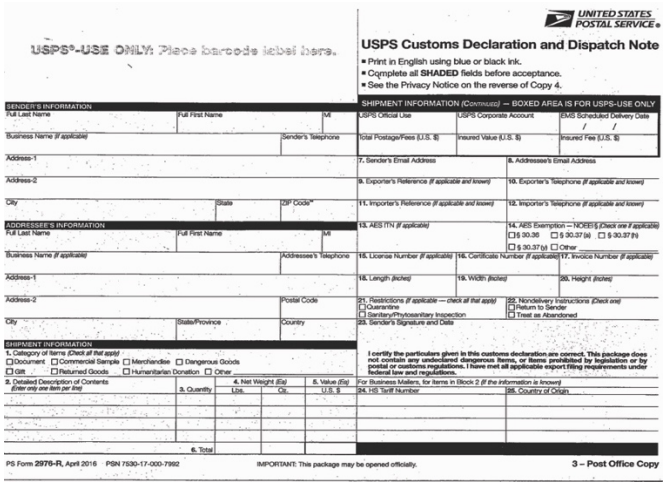

Fig. 1 Sample form (clean, without handwritten information)

Preprocessing:

The preprocessing of the forms involves changing the forms to grayscale (from RGB) and then rescaling them so that they are all of the same size in order to be given to the RBM. It is expected that rescaling will cause interpolation among the pixel values and hence incorporate some distortion in the dataset.

For some of the experiments, using the entire form was proving to be a computational burden and hence only a section of the form was cropped and considered.

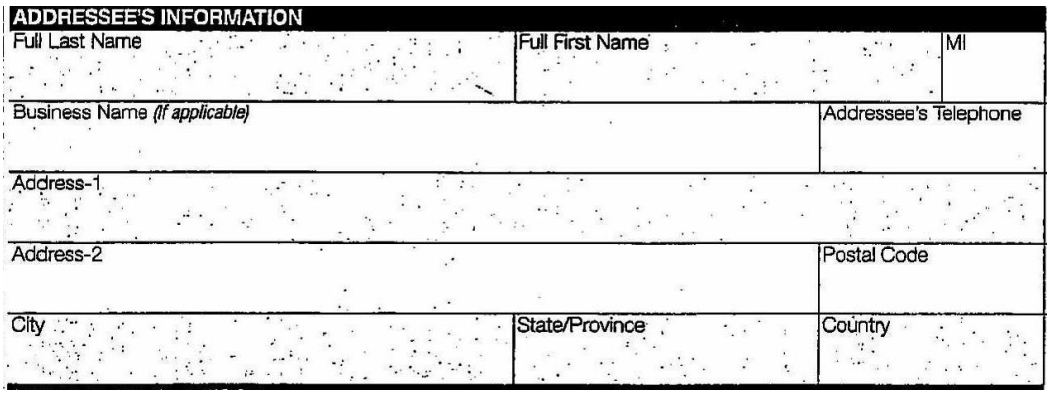

Fig. 2 Section of a form (cropped)

Since the input to the RBM needs to be binary or in the form of a probability between 0 and 1, we normalized our pixel values so that they would lie in the range of 0,1 (expressed as probabilities). Visually, it was seen that this kind of a normalization did not introduce any distortion in the form.

Implementation

Initial experiments:

All the experiments were carried out using MATLAB. A library written in MATLAB for RBM's by Andrej Karpathy was used as a starting point.

Each of the images was also reshaped in the form of a row vector of size 1 by (m*n), where m and n are the dimensions of the above cropped form image (Fig. 2). The RBM was then trained to learn the weights based on inputs which are the forms. As in all learning algorithms, stochastic relaxation was applied to the learning rate parameter with increasing number of epochs.

The figure below shows the the image obtained after reconstruction for the first form (out of 20)

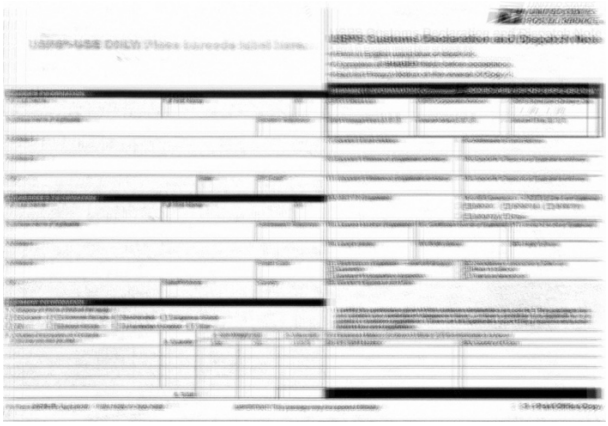

Fig. 3 Reconstructed form using 20 hidden units

As can be seen the form is not exactly clear and has a certain reconstruction error associated with it.

Initially only 20 hidden units were used to train the RBM. Later, we experimented by using 50 and 100 hidden units as well. It was observed that the reconstruction error would reduce by increasing the number of hidden units. Allowing the network to run for increasing number of epochs also seemed to decrease the reconstruction error. The RBM was allowed to train for about 1000 epochs to see if the reconstruction error would go down to zero. It was observed that towards the end, with increasing number of epochs, there would be oscillations in the reconstruction error and it seemed to saturate. As per theory, it is expected that, the noise in the gradient estimates might cause the reconstruction error on the individual mini-batches to fluctuate gently after the initial rapid descent. Since we had only 20 forms as the size of the dataset, we used only one mini-batch.

In order to probe into the behavior of the RBM and learn what the weights of the RBM are able to capture, certain other experiments were carried out.

1. Instead of giving 20 variations (due to scanning) of a form as input, only one variant of a form, repeated 20 times was given as input to the RBM. Only one hidden unit was considered. The weight associated with the hidden unit was found to capture the image pattern itself as shown below: (A mean square error of 0.3774 was reported between the image and the weight). Also the reconstructed form was found to be visually identical to the original form itself with an MSE of 0.0023 between the original form and the reconstruction.

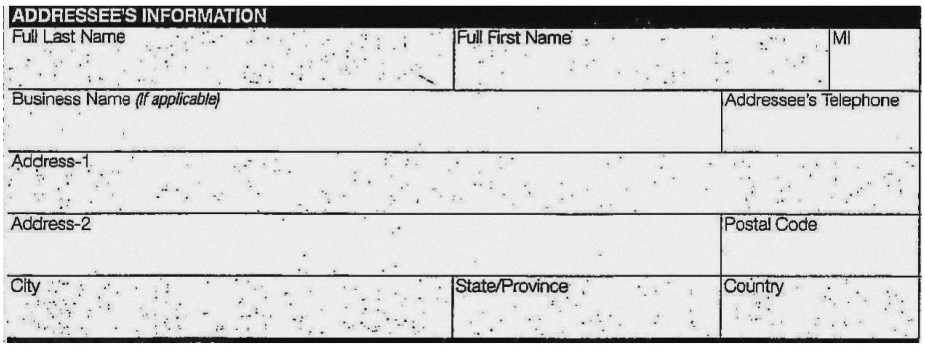

Fig.4 Weight associated with the hidden unit (one form and one hidden unit)

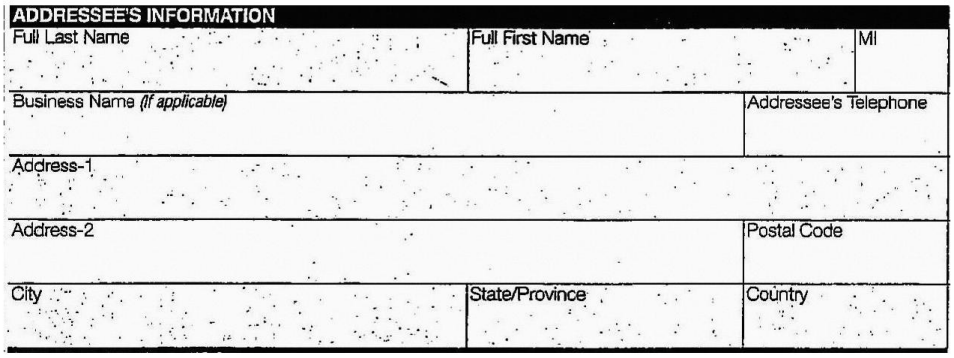

Fig.5 Reconstructed form

2. Two variations of a form repeated 10 times each was given as an input to the RBM. Only 2 hidden units were considered, and the weight obtained in each hidden unit was found to be the average of the two forms. It was expected that, each hidden unit would capture the probability distribution one form each. instead each hidden unit was found to capture the average of both forms. Even on increasing the number of hidden units to three or four, the same behavior was observed (each hidden unit would capture the average of the forms).

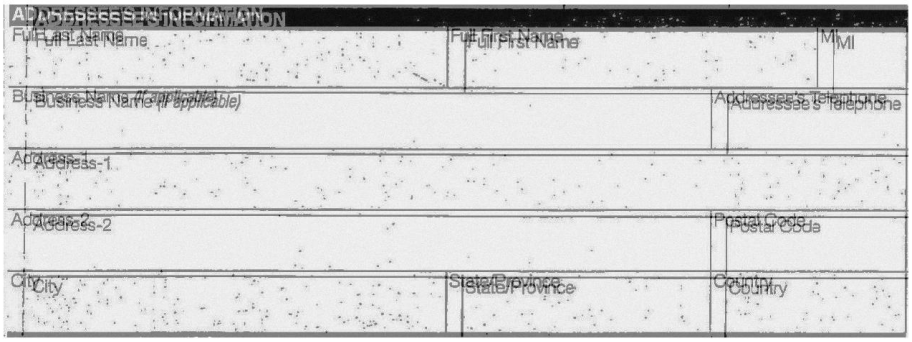

Fig.6 Weight associated with the hidden unit (two form and two hidden unit)

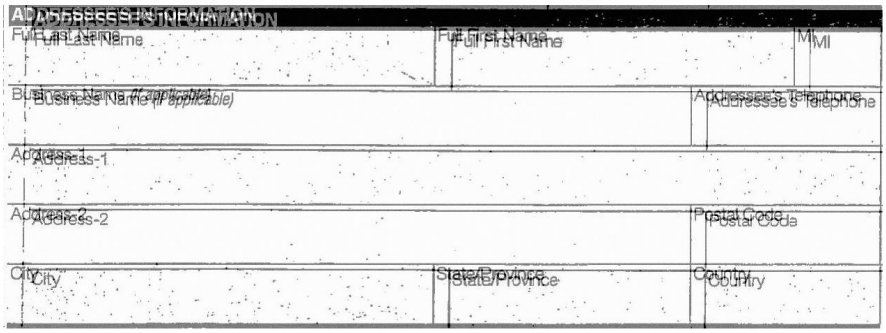

Fig.7 Reconstructed form (seen as a fusion of two forms)

3. The same experiment was tried out with 4 variations of the form repeated 5 times each in order to have a dataset size of 20. In this too, keeping 4 hidden units it was observed that each hidden unit had a weight that was an average of all the 4 forms in the dataset.

A reconstruction error between the reconstructed form and the original forms would always exist.

Such a reconstruction error is expected when there is noise in the dataset. But even when all forms were same, there was still a reconstruction error and Mean square error associated with the reconstructed form. The reason why this error appears can be related to the inherent approximations in the learning algorithm itself.
One, the contrastive divergence learning algorithm (used here) is only an approximate representation of the gradient of the log probabilities required to be computed during learning. Also, the Gibbs sampling procedure, which is a key component of the learning in RBM's is itself random in nature (it is only a sampling of the probability distribution function, although it has been proven that performed repeatedly, Gibbs sampling eventually converges). It is possible that these approximations lead to the inherent error in reconstruction even after the algorithm is allowed to run for multiple epochs.

Adding the handwritten data:

The handwritten information was added synthetically to the clean form by overlaying the scanned handwritten information on top of the clean form. This would cause some errors during normalization as each image would be individually digitized as opposed a form that comes with handwritten data and would be digitized at once.

Two kinds of handwritten data, one was a random scribble and the other was an actual handwritten input was used in the experiments.

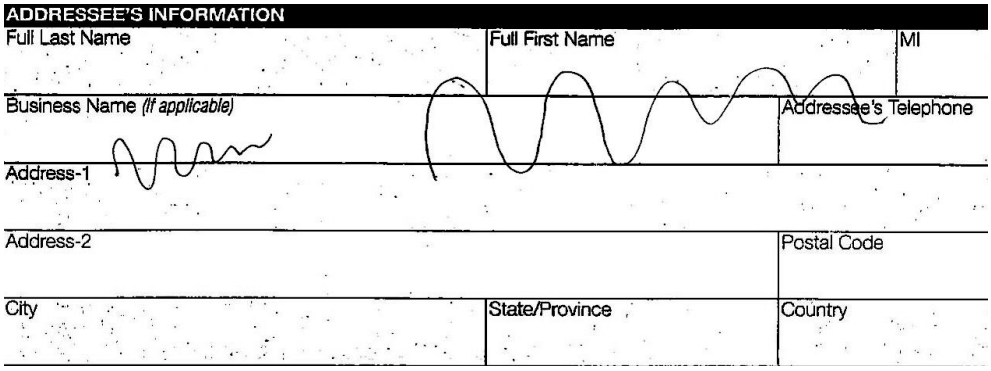

Fig.8 Filled form (with handwritten scribble)

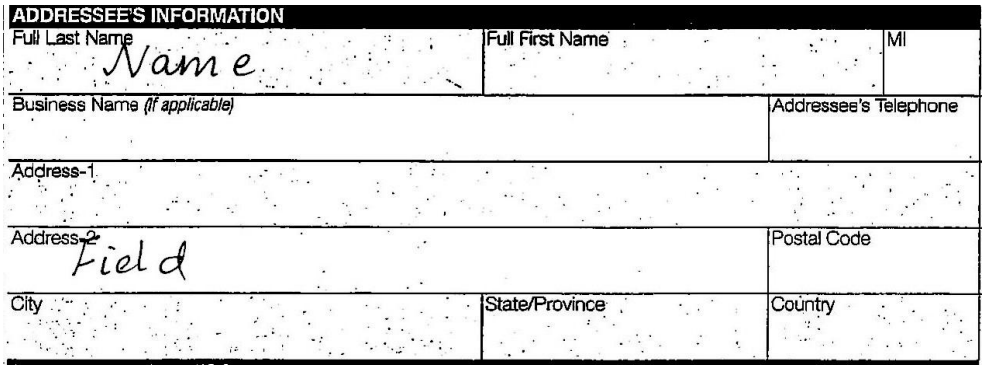

Fig.9 Filled form (with handwritten information)

After the RBM was trained on a clean form, a form containing handwritten data was given as input to the RBM in order to obtain a reconstruction. As expected, the reconstructed output would converge to the learned clean form itself leaving out the handwritten data.

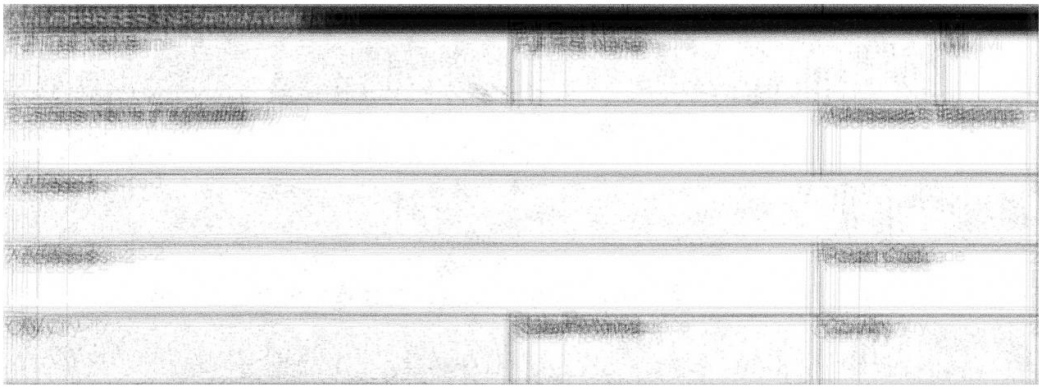

Fig. 10 Reconstructed form, although blurry; the image does not contain any handwritten data as expected

In order to extract the handwritten part, the reconstructed form was subtracted from the handwritten input form. As can be seen below, the subtracted form has fixed contents also due to the reconstruction error (that causes a blurry reconstruction as above)

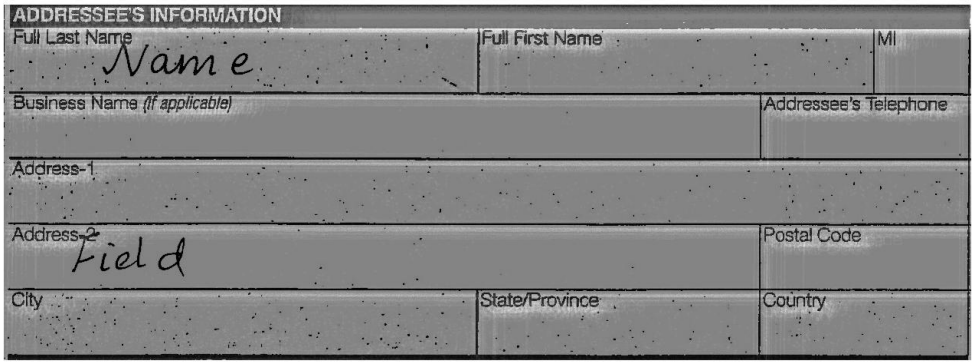

Fig. 11 Subtracted form

Since the handwritten fields are clearly visible it is possible to apply image thresholding techniques to keep only the handwritten part.

Image processing to extract the handwritten data:

Since the handwritten data was the blackest (as seen in Figure 11, in terms of pixel value it was close to one or 255), a suitable threshold was applied to the image. Figure 12 is the result of applying image thresholding to the subtracted form (Figure 11).

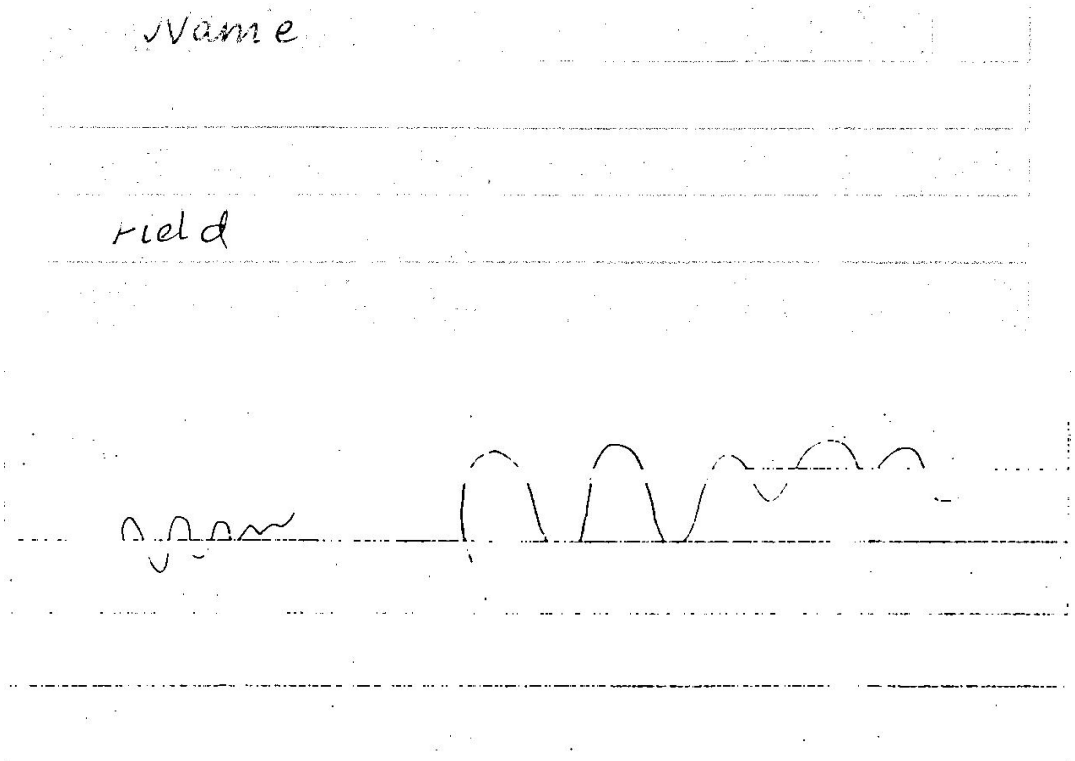

Fig. 12 Form after image processing applied

It can be seen that the handwritten data has been extracted, along with a few lines that might have been uncorrelated with the input due to noise in the dataset leading to a reconstruction error in the original form. In order to enhance the output further, another layer of RBM was added to see if handwritten data can be extracted more cleanly.

Adding another layer, Deep Belief Nets:

Since RBM's act as good denoising agents and are able to impute missing information, theory has found that a stack of RBM's can perform even better. We made use of this theory, by adding an additional layer of RBM on top of the first that is trained on output from the first.
The output from the first layer of RBM (blurred reconstruction) is given as input to the second RBM which learns its weights accordingly. Once trained, the handwritten form in given as input to the second RBM which outputs a reconstruction of the original form. On subtracting from the handwritten form and applying image processing as above we observe the following results:

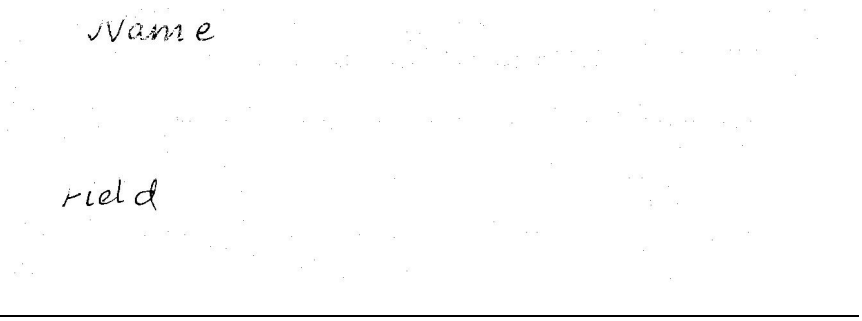

Fig. 13 Handwritten information extracted after final processing

As can be seen there is improved performance over using a single layer of RBM and handwritten data has been reasonably well extracted.

Conclusion

Here we use the correlation properties of an RBM to extract handwritten data from forms. We see that a single layer of RBM's after applying image processing techniques to the reconstruction, performs reasonably well; although a two layer RBM can be used to improve performance. Additional experiments were carried out to understand the probabilistic patterns captured in the weights associated with the hidden units of an RBM.

Future Work:

The scanned images or forms would all be slightly different due to various distortions introduced during the process of data collection and digitization. These include distortions due to change in rotation and translation during feeding the form to the scanner, as well as distortions introduced to due to pixel interpolation while preparing the dataset. Probabilistic models could be used to model these distortions. Also incorporating the knowledge of these image distortion models would help in an evaluation of our reconstructed form that would be less sensitive to distortion (would not evaluate to poorly due to distortion). Also, this knowledge of distortion models can be used in the convolutional layers of convolutional neural networks that use weights that are initialized based on the distortion model corresponding to the distortion present in the dataset.

Further, the neural network corresponding to a trained RBM or DBN can be augmented by an output layer, where units in the newly added output layer represent labels corresponding to observations (forms in our case). We could label different forms (variations) differently in order to further probe into the weight patterns captured by the RBM.

References:

[1] Hinton, Geoffrey E. "A Practical Guide To Training Restricted Boltzmann Machines". *Lecture Notes in Computer Science* (2012): 599-619. Web. 24 Apr. 2017.
[2] Juang, Biing Hwang. "Deep Neural Networks – A Developmental Perspective". *APSIPA Transactions on Signal and Information Processing* 5 (2016): n. pag. Web.
[3] Fischer, Asja, and Christian Igel. "Training Restricted Boltzmann Machines: An Introduction". *Pattern Recognition* 47.1 (2014): 25-39. Web.