Modeling High-Dimensional Classification Problems Using Deep Learning, and Feedforward Neural Networks

Suraj Narayanan Sasikumar

Research School of Computer Science, Australian National University
u5881495@anu.edu.au

Abstract. Deep learning methods have been successfully applied to convert high-dimensional to low-dimensional codes [1]. These low-dimensional codes are essentially higher-level features that can provide good discriminability for classification tasks. In this study we train a deep network for feature extraction and classification on the 10,000 dimensional ARCENE [2] dataset. The unsupervised pre-training for feature extraction is done by a stacked autoencoder (SAE) [4], and the subsequent supervised logistic regression by a softmax classifier. We then compare the balanced error rate (BER) performance measure of the deep network with that of a feedforward neural network. The report also highlights why a Bayesian neural network (BNN) was considered but not selected for the study. We observed that the SAE classifier bettered ARCENEs baseline BER [3], where as the feedforward network was two points shy.

Keywords: Classification, Dimensionality reduction, Deep Learning, Autoencoders, Feedforward Neural Networks, Feature Selection, Feature Extraction, Cancer Diagnosis

1 Introduction

In todays world of Big-Data, high-dimensional data is often confronted during data mining. Input features needed for a good neural network design can either be a direct subset (feature selection), or can be higher-level features created as a function of the original features (feature extraction), this process is called dimensionality reduction. The data set that is the focus of this study, ARCENE, is one of the most challenging data set owing to the fact that its dimensionality exceeds the number of available data points by orders of magnitude. ARCENEs task is to classify between cancerous and non-cancerous patterns found within the mass-spectrometric data. In this study we use two types of neural networks, namely, deep learning neural networks and feedforward neural networks to achieve both dimensionality reduction and classification.

1.1 Dimensionality Reduction via Feature Extraction

Generally we would use principle component analysis (PCA), or linear discriminant analysis (LDA) as feature extraction techniques, however these methods

fail [8] to model the nonlinear features found in ARCENE. Since deep learning methods can learn nonlinear features, and also have the capability to process large-scale data sets, we decided to use a deep network for feature extraction from ARCENEs massive input space. The process of deep learning for deep neural networks take place in two stages:

- Unsupervised layer-wise pre-training [4]
- Supervised logistic regression

Training a deep network, as a whole is difficult, hence we use layer-wise pre-training to initialize the weights of each layer one by one, and then finally use the individual layer weights as the initial weights for the whole deep network. The supervised logistic regression further tweaks the network weights by minimizing the mis-classification error rate. The advantage of using deep learning neural networks is that we get both high level features as well as classification from the same network.

1.2 Dimensionality Reduction via Feature Selection

Feature selection is the process of selecting a subset of relevant features for use in model construction, in this study we use mainly three techniques to remove irrelevant features:

- Variance threshold filtering
- Target Correlation filtering
- Univariate feature selection with Relief score [6][7]

In variance threshold filtering we compute the variance of each feature in the give data set and remove those feature that have near zero variance, i.e. those feature that vary very little in the sample space. Subsequently we calculate the correlation of each feature with the target and filter out those features whose correlation value fall in the range $[-0.1, 0.1]$. This means that the value of those features change very little as the target changes. Lastly, we rank the remaining features based on its relief score and choose the top 1400 features as our final feature subset.

With the reduced subset we train a feedforward neural network using the scaled conjugate gradient (SCG) [9] learning algorithm by minimizing the cross-entropy cost function.

2 Methodology

2.1 Modeling the Deep Network

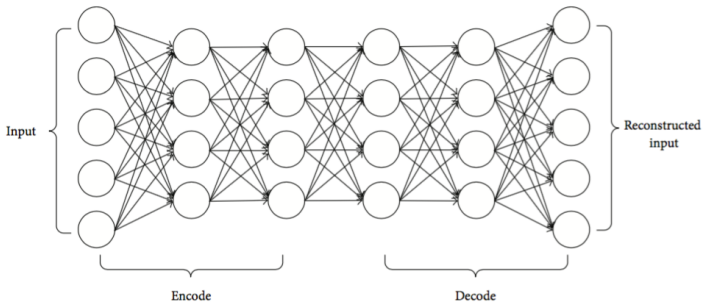

Fig. 1. The SAE network structure

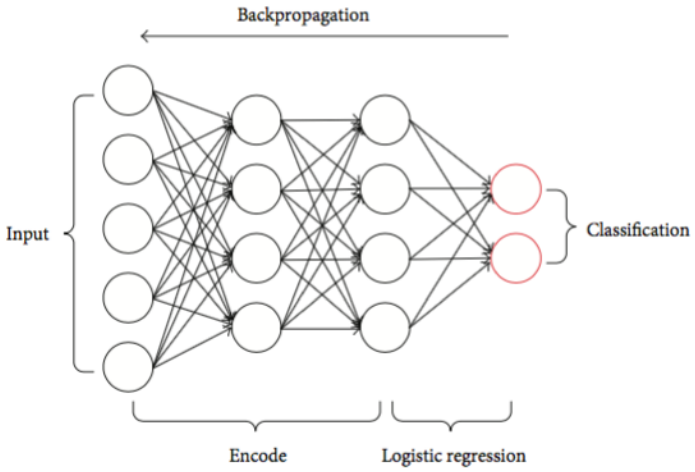

Fig. 2. Feature Extraction and classification

An autoencoder is a constrained neural network whose outputs are the same as the inputs. The network encodes the data into a compressed format and then decodes it back into an approximation of the original data. In doing so the hidden layer of the network essentially learns a representation of the original data. Autoencoders can be stacked together to obtain higher-level representation of the data, which we can extract as features to be used in a classification task.

Figure 1 shows a typical instance of an SAE structure, which includes two encoding layers and two decoding layers. In the encoding part, the output of the first encoding layer acts as the input data of the second encoding layer. Figure 2 shows a classification problem using the output of the encoding part of the SAE as its input.

In our study we use a two layer stacked autoencoder network to extract higher-level features [5]. Layer-wise pre-training is performed on the SAEs to obtain an initial set of weights on the network. The hidden layer of the autoencoder is modelled to contain 50 hidden neurons thus essentially extracting 50 high-level features from the $10,000$ input features of ARCENE. Each layer of the autoencoder is trained using SCG learning algorithm by minimizing the mean-squared error cost function. A value of 10 for the regularization coefficient in the cost function was found to give the best results. Another layer with a softmax classifier is added on top and is trained using the target labels to perform logistic regression. We further train the whole deep network on our training data to further fine-tune the network weights.

2.2 Modeling the FeedForward Network

Modeling a feedforward neural network for classifying ARCENE involves the following steps:

- feature selection and data normalization
- 10-fold cross-validation to optimize the number of hidden neurons
- Generalizing the network by re-training on all cross validation partitions

The neural network is trained using the SCG learning algorithm with cross-entropy as the cost function. Since presence of multiple hidden layers did not produce any improvement in the results, we modeled the network to have only one layer of hidden neurons. The hidden neuron count was considered as a hyper-parameter and optimized using 10-fold cross validation in range $[1, 100]$. It was found that a single hidden layer with 29 hidden neurons produce the best result.

Since each training session starts with different initial weights, biases, and division for training, test, and validation data sets, the network gives a different solution for the same problem. To overcome this we train the network inside all partitions of the 10-fold cross validation setup, this coupled with early stopping and regularization can ensure a generalized stable neural network.

3 Results and Discussion

The primary metric with which the performances of both the network were assessed is the balanced error rate (BER). BER is the average of the error rate of the positive class and the error rate of the negative class. Table 1 shows the BER, true negative (TN), true positive (TP), false positive (FP), false negative (FN), and the percentage of dimensionality reduction for the network under investigation and the values mentioned in the reference paper [3].

Table 1. Performance comparison for Deep Learning and Feedforward network

Network Type	BER	TN	TP	FP	FN	%Dim Reduction
Stacked Autoencoder + SoftMax	0.1437	45	40	11	4	99.5
Feedforward	0.1648	43	36	13	8	86
Bayesian	0.2321	44	33	12	11	98.5
Reference Baseline	0.1470	NA	NA	NA	NA	89
Reference Best(SVC)	0.1073	NA	NA	NA	NA	0

The reference paper does not mention any deep learning techniques used for dimensionality reduction. The fact that our SAE network was able to better the baseline score with not much effort shows that with more careful analysis we would be able to better the best score of 0.1073. The score of feedforward neural networks, though not spectacular, was in the upper echelons of most results cited in the reference paper.

During the initial phase of the study the use of Bayesian Neural Network (BNN) was considered, as the best result in the reference paper was obtained by the use of BNN. Bayesian methods allow us to consider an entire distribution of model parameters, i.e. initial weights, biases, and regularization coefficient, without the need for separately cross-validating them as hyper-parameters. This means training over a huge parameter space. Though possible with Mote Carlo Sampling methods [11], the use of BNN for this study was deemed out of scope due to time and resource constraints.

4 Conclusion and Future Work

The results from this study strongly suggest that using SAE classifier is a strong contender for classification problems with high-dimensions. In the future, model parameters such as the number of hidden neurons in the SAE can be optimized using model selection techniques to get even better results. One of the challenges faced during the study was the time taken by SAEs to train; this concern can be mitigated by the use of a different class of autoencoders known as Marginalized Denoising Autoencoders. These are known to provide better generalization and faster training times [12].

References

1. Hinton, G.E. and Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. Science, 313(5786), pp.504-507.
2. Guyon, I., Gunn, S.R., Ben-Hur, A. and Dror, G., 2004. Arcene data set. Access mode: http://archive. ics. uci. edu/ml/datasets/Arcene.
3. Guyon, I., Li, J., Mader, T., Pletscher, P.A., Schneider, G. and Uhr, M., 2007. Competitive baseline methods set new standards for the NIPS 2003 feature selection benchmark. Pattern recognition letters, 28(12), pp.1438-1444.

4. Bengio, Y., Lamblin, P., Popovici, D. and Larochelle, H., 2007. Greedy layer-wise training of deep networks. Advances in neural information processing systems, 19, p.153.
5. Shin, H.C., Orton, M.R., Collins, D.J., Doran, S.J. and Leach, M.O., 2013. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 35(8), pp.1930-1943.
6. Kira, K. and Rendell, L.A., 1992, July. The feature selection problem: Traditional methods and a new algorithm. In AAAI (Vol. 2, pp. 129-134).
7. Kononenko, I., imec, E. and Robnik-ikonja, M., 1997. Overcoming the myopia of inductive learning algorithms with RELIEFF. Applied Intelligence, 7(1), pp.39-55.
8. Cheriyadat, A. and Bruce, L.M., 2003, July. Why principal component analysis is not an appropriate feature extraction method for hyperspectral data. In Geoscience and Remote Sensing Symposium, 2003. IGARSS'03. Proceedings. 2003 IEEE International (Vol. 6, pp. 3420-3422). IEEE.
9. Mller, M.F., 1993. A scaled conjugate gradient algorithm for fast supervised learning. Neural networks, 6(4), pp.525-533.
10. Guyon, I., Gunn, S., Ben-Hur, A. and Dror, G., 2004. Result analysis of the nips 2003 feature selection challenge. In Advances in neural information processing systems (pp. 545-552).
11. Neal, R.M. and Zhang, J., 2006. High dimensional classification with Bayesian neural networks and Dirichlet diffusion trees. In Feature Extraction (pp. 265-296). Springer Berlin Heidelberg.
12. Chen, M., Weinberger, K.Q., Sha, F. and Bengio, Y., 2014. Marginalized denoising auto-encoders for nonlinear representations. In Proceedings of the 31st International Conference on Machine Learning (ICML-14) (pp. 1476-1484).