TRAINING DATA REDUCTION IN DEEP NEURAL NETWORKS WITH PARTIAL MUTUAL INFORMATION BASED FEATURE SELECTION AND CORRELATION MATCHING BASED ACTIVE LEARNING

Jian Zheng, Wei Yang, and Xiaohua Li

State University of New York at Binghamton
Department of ECE, Binghamton, NY 13902
{jzheng65, wyang17, xli}@binghamton.edu

ABSTRACT

In this paper, we develop a novel scheme to reduce the amount of training data required for training deep neural networks (DNNs). We first apply a partial mutual information (PMI) technique to seek for the optimal DNN feature set. Then we use a correlation matching based active learning (CMAL) technique to select and label the most informative training data. We integrate these two techniques with a DNN classifier consisting of layers of unsupervised sparse autoencoders and a supervised softmax layer. Simulations are then conducted over the breast cancer data set from the UCI repository to show that this scheme can drastically reduce the amount of labeled data necessary for the DNN training, and can guarantee the superior performance in reduced training data sets.

Index Terms— Deep neural network, feature selection, partial mutual information, active learning, classification

1. INTRODUCTION

We have seen an explosion of research in deep learning since its emergence [1]. Deep neural networks (DNNs) distinguish from those shallow-structured neural networks by their depth or the number of hidden layers. The deep depth makes it possible to extract more complex latent structures, to learn a hierarchy of features automatically from data, and to achieve unprecedented performance [2].

One of the challenges of DNNs is that they need an enormous amount of labeled training data. The input data of DNNs tend to be high dimensional, the optimization problems tend to be highly nonlinear and complex, and the training procedure tends to converge slowly. These factors lead to the requirement of a large amount of training data. However, in practice, it can be very difficult to obtain a sufficient amount of training data, and it is extremely labor intensive to label the data accurately.

To reduce the amount of training data that are needed in DNNs, firstly, we need to reduce the effective dimension of the input data. The input data usually have a large proportion of irrelevant or redundant features which not only degrade the accuracy of the learning models [3] but also call for an unnecessarily large training data set.

Since irrelevant or redundant features in the input data may not improve the performance of learning, many input variable selection (IVS) algorithms have been developed to choose the features which are highly informative and least redundant. The IVS algorithms can be subdivided into two categories, i.e., filters and wrappers. One type of the filters algorithms exploits the partial mutual information (PMI) as optimization objective [4]. As an extension of mutual information, PMI measures the dependency among multiple variables. PMI-based feature selection is capable of resolving the feature redundancy problem without assuming any dependence structure among the features or variables. Nevertheless, the IVS algorithms are conventionally studied in terms of making the training task computationally more efficient or improving the learning performance, rather than training data reduction.

Training data reduction is conventionally a subject of active learning (AL). AL aims to select and label the most ambiguous and informative training samples [5]. This is desirable when the training data are costly to label [6] or the size of the training data set is too big. An active learning algorithm that directly minimizes the expected generalization error was developed in [7]. Algorithms in [8] and [9] used the stratification and vector norm maximization techniques, respectively. Sequential active learning was studied in [10][11] under a concept of integrated human and machine learning.

In this paper, we apply jointly the PMI-based feature selection technique and a correlation matching based active learning (CMAL) technique to reduce the amount of training data needed in DNNs. This scheme will also improve the robustness of the DNNs in case of reduced training data sets. The PMI technique is firstly applied to select the optimal feature set by abandoning those irrelevant and redundant features. Then, the simple yet effective CMAL technique is applied to select the most informative training data. We use

This work is supported by NSF via grant CNS-1443885.

these two techniques to construct a new DNN classification scheme, and test it with a hand-crafted data set.

The remainder of this paper is organized as follows. In Section 2, the DNN classification model is introduced. The integration of PMI and CMAL with DNN is presented in Section 3. Simulations are presented in Section 4 and conclusions are given in Section 5.

2. DNN CLASSIFICATION MODEL

In this paper, we consider a DNN classification model that consists of layers of sparse autoencoders and a softmax layer, followed by a fine tuning procedure. The sparse autoencoders are trained in an unsupervised way to learn features of different levels from the input data. The softmax layer is trained in a supervised way for classification. Fine tuning is conducted via backpropagation to further improve the performance.

2.1. Unsupervised Training of Sparse Autoencoders

An autoencoder is a neural network that learns features from unlabeled data. Given an unlabeled training data set $\{\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_N\}$, the output of an autoencoder $\mathbf{y}_i$ are set to be the same as the input $\mathbf{x}_i$, i.e., $\mathbf{y}_i = \mathbf{x}_i$, where $\mathbf{x}_i, \mathbf{y}_i \in \Re^K$, K is the number of features, and N is the number of training data records.

The autoencoder tries to learn a function that approximates the identity function whose output is similar to the input. For the input data vector $\mathbf{x}_i \in \Re^K$, a single-layer autoencoder maps it to a hidden representation through $h(\mathbf{x}_i) = f(\mathbf{W}_1\mathbf{x}_i + \mathbf{b}_1)$, where $f(z)$ is a non-linear activation function (e.g., $f(z) = 1/(1+\exp(-z))$), $h(\mathbf{x}_i) \in \Re^M$ is the neuron activation vector, $\mathbf{W}_1 \in \Re^{M\times K}$ is a weight matrix, $\mathbf{b}_1 \in \Re^M$ is a bias vector and M is the number of hidden units. The reconstructed output of the autoencoder is $\hat{\mathbf{x}}_i = f(\mathbf{W}_2 h(\mathbf{x}_i) + \mathbf{b}_2)$, where $\mathbf{b}_2 \in \Re^K$, $\mathbf{W}_2 \in \Re^{K\times M}$, and we have $\hat{\mathbf{x}}_i \approx \mathbf{x}_i$.

With all the training data $\mathbf{x}_i$, backpropagation can be used to adapt the weight matrices and bias vectors so as to minimize the reconstruction error

$$J(\mathbf{W}, \mathbf{b}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|^2. \tag{1}$$

If $M > K$, the autoencoder would not compress the input information, but preserves all the input information trivially. To avoid this trivial solution, sparsity constraints can be imposed on the hidden units to reduce the amount of information put into the hidden layer. A way to realize the sparsity constraint is to apply the penalty regularization term $\alpha\sum_{j=1}^{M} KL(\rho\|\hat{\rho}_j)$, which adds together the Kullback-Leibler divergence [12]

$$KL(\rho\|\hat{\rho}_j) = \rho\log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}. \tag{2}$$

In (2), $\hat{\rho}_j = \frac{1}{N}\sum_{i=1}^{N} h_j(\mathbf{x}_i)$ is the average activation over the jth hidden unit, ρ is a sparsity level, and α is used to control the impact of the sparsity regularizer.

In addition, a weight decay term $\beta\|\mathbf{W}\|_2^2$ can also be used to decrease the weights of the network in terms of magnitude to get rid of overfitting, where $\|\mathbf{W}\|_2^2 = tr(\mathbf{W}'\mathbf{W})$, $(\cdot)'$ denotes transpose, and β is utilized to control the impact of the above weight regularizer [13].

As a result, the overall cost function of a sparse autoencoder becomes

$$J_{sparse}(\mathbf{W}, \mathbf{b}) = J(\mathbf{W}, \mathbf{b}) + \alpha\sum_{j=1}^{M} KL(\rho\|\hat{\rho}_j) + \beta\|\mathbf{W}\|_2^2. \tag{3}$$

In our model, multiple such sparse autoencoders are applied sequentially to learn more and more abstracted or compressed features automatically from the input data. Specifically, let $\mathbf{X} \in \Re^{K\times N}$ be the input training data set, each sparse autoencoder ℓ, where $\ell = 1, \cdots, L$, is trained and its output feature set $\mathbf{X}_\ell \in \Re^{M_\ell \times N}$ is used as the input to the next autoencoder $(\ell+1)$ to achieve a more abstract and compressed feature set $\mathbf{X}_{\ell+1} \in \Re^{M_{\ell+1}\times N}$. Note that M_ℓ is the number of hidden units of the sparse autoencoder ℓ, and we have $M_\ell > M_{\ell+1}$.

2.2. Classification and Fine-tuning

After extracting features at various levels with the L sparse autoencoders in the unsupervised way, a supervised softmax classification layer is added. This softmax layer has the last extracted feature set $\mathbf{X}_L$ and the corresponding target set $\mathbf{y} \in \Re^N$ as input, which means this layer is trained with the labeled training data set. Therefore, the autoencoders are used to extract more and more abstract features, while the softmax layer is used to classify the data in a supervised manner.

We stack all the $L+1$ trained layers together to form the overall DNN that is used to conduct the classification tasks. The output of the DNN can be further improved with a fine tuning procedure, which is conducted by applying backpropagation over the multilayer DNN iteratively.

3. INTEGRATING PMI AND CMAL WITH DNN

In order to reduce the amount of training data for the DNN, in this section, we integrate the DNN scheme of Section 2 with a PMI-based feature selection technique and a CMAL-based training data selection technique.

3.1. Partial Mutual Information for Feature Selection

Let $\mathbf{Z}$ be a set of pre-selected features of the input data. For the input discrete variable X (with realizations x_i) and the discrete variable Y (with realizations y_i), the corresponding

PMI is calculated as

$$PMI = \frac{1}{N}\sum_{i=1}^{N} \ln\left[\frac{f_{X',Y'}(x_i', y_i')}{f_{X'}(x_i')f_{Y'}(y_i')}\right], \quad (4)$$

where $x_i' = x_i - E(x_i|\mathbf{Z})$ and $y_i' = y_i - E(y_i|\mathbf{Z})$ represent the residual information of x_i and y_i, $i = 1, \ldots, N$, respectively. In (4), $E(\cdot)$ denotes expectation, while $f_{X'}(x_i')$, $f_{Y'}(y_i')$ and $f_{X',Y'}(x_i', y_i')$ are the marginal and joint probability densities. The conditional expectation $E(x|\mathbf{Z})$ can be estimated as

$$E(x|\mathbf{Z}) = \frac{1}{N}\sum_{i=1}^{N} \omega_i x_i, \quad (5)$$

where, with a bandwidth parameter λ, we define

$$\omega_i = \frac{\exp\left(-\frac{\|\mathbf{Z}-\mathbf{Z}_i\|}{2\lambda^2}\right)}{\sum_{j=1}^{N}\exp\left(-\frac{\|\mathbf{Z}-\mathbf{Z}_i\|}{2\lambda^2}\right)}. \quad (6)$$

The bandwidth parameter is calculated as

$$\lambda = \left(\frac{1}{p+2}\right)^{\frac{1}{p+4}} \sigma N^{-\frac{1}{p+4}}, \quad (7)$$

where σ is the standard deviation of the data sample, p is the dimension of each $\mathbf{Z}_i$, and $\|\mathbf{Z} - \mathbf{Z}_i\|$ is the Mahalanobis distance. Note that the Mahalanobis distance is defined as

$$\|\mathbf{Z} - \mathbf{Z}_i\| = (\mathbf{Z} - \mathbf{Z}_i)'\sum^{-1}(\mathbf{Z} - \mathbf{Z}_i), \quad (8)$$

where $\sum$ is the sample covariance matrix.

Kernel density estimation (KDE) is used to estimate the marginal probability density

$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N} K_\lambda(x - x_i), \quad (9)$$

where

$$K_\lambda = \frac{1}{\left(\sqrt{2\pi}\lambda\right)^p \sqrt{|\sum|}} \exp\left(\frac{-\|x - x_i\|}{2\lambda^2}\right) \quad (10)$$

is the Gaussian kernel. Similarly, the joint probability density defined in [14] is estimated as

$$\hat{f}(x, y) = \frac{1}{N}\sum_{i=1}^{N} K_\lambda(x - x_i)K_{\lambda_y}(y - y_i). \quad (11)$$

The procedure of using PMI as a metric to reduce the number of features is shown in Algorithm 1. The features are selected one by one. The next most informative feature is selected as the one that gives the maximum PMI when taking into consideration of those features that have already been selected. This process is performed iteratively until a stopping criterion is met. With the selected features, the input data set $\mathbf{X}$ can be rewritten as $\mathbf{X}_{pmi}$.

3.2. Correlation Matching based Active Learning

In our scheme, we use CMAL to select and label T of the most informative training data out of the overall N data samples $\mathbf{X}_{pmi}$. This is conducted iteratively as follows. Considering the jth iteration, where $j = 1, \cdots, T$, we need to select a new data sample $\mathbf{x}_{t_j} \in \Re^L$. Note that in the previous iterations we have already selected $j-1$ training data samples and formed the training data set $(\mathbf{X}_{j-1}, \mathbf{y}_{j-1})$, where $\mathbf{X}_{j-1} = [\mathbf{x}_{t_1}, \cdots, \mathbf{x}_{t_{j-1}}]'$ and $\mathbf{y}_{j-1} = [y_{t_1}, \cdots, y_{t_{j-1}}]'$.

The new data sample $\mathbf{x}_{t_j}$ is then selected from the data pool $\mathcal{X}_j = \{\mathbf{x}_i | 1 \le i \le N,\ \mathbf{x}_i \ne \mathbf{x}_{t_\ell},\ 1 \le \ell \le j-1\}$. We can simply evaluate each of the $N - j + 1$ data samples in $\mathcal{X}_j$ and choose

$$\mathbf{x}_{t_j} = \arg\min_{\mathbf{z}\in\mathcal{X}_j} \left\| \frac{1}{j}\left(\mathbf{X}_{j-1}'\mathbf{X}_{j-1} + \mathbf{z}\mathbf{z}'\right) - \mathbf{R}\right\|^2, \quad (12)$$

where $\mathbf{R} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i'$ is the sample correlation matrix of the overall data pool. The objective of the optimization (12) is to rapidly match the correlation of the training data set with that of the overall data pool.

With the selected $\mathbf{x}_{t_j}$, we acquire its corresponding label y_{t_j} and append $(\mathbf{x}_{t_j}, y_{t_j})$ into the set $(\mathbf{X}_{j-1}, \mathbf{y}_{j-1})$ to get $(\mathbf{X}_j, \mathbf{y}_j)$. In this way, T training data samples $\mathbf{X}_{cmal} \in \Re^{L\times T}$ and the corresponding data label set $\mathbf{y} \in \Re^T$ are determined. These data will be used in the subsequent training of the DNN.

3.3. DNN Classification Algorithm

The overall procedure of applying the PMI and CMAL techniques to optimize the number of training data is shown in Algorithm 1. The PMI-based technique is first applied to select the most informative and least redundant input data features. Then the CMAL-based technique is applied to select and label the most informative training data. The reduced training data set with the optimized feature set serves as the input to train the multilayer DNN.

The PMI iteration continues until the performance of the overall DNN scheme starts to degrade when adding the last selected feature whose PMI value is I_{min}.

4. SIMULATIONS

We have conducted intensive simulations to evaluate the performance of our proposed scheme, in particular the DNN classification accuracy under reduced training data sets. In simulations, we compared our algorithm **DNN_CMAL** with four other algorithms: **SVM_Full** and **DNN_Full** which applied SVM (support vector machine) and DNN without training data optimization; **DNN_Random** which selected training data randomly; and **DNN_PBAL** which applied DNN and the active learning algorithm of [7].

Algorithm 1 DNN feature and training data optimization

procedure PMI
Input:$\{\mathbf{X}_{in} \in \Re^{K\times N} | \mathbf{x}_i, i = 1, \ldots, K\}$, $\mathbf{y}_{in} \in \Re^N$, $\mathbf{Z} \rightarrow \emptyset$
While $\mathbf{X}_{in} \neq \emptyset$
    For $\mathbf{y}_{in}$ and each $\mathbf{x}_i$, estimate the residue information
        y_i' and x_i' according to (5)-(8)
    Estimate the marginal and joint probability densities
        according to (7)-(11)
    Estimate each PMI value I with (4) and select the
        candidate feature $\mathbf{x}$ that maximizes I
    if $I < I_{min}$ **then**
        Exit
    Move $\mathbf{x}$ to $\mathbf{Z}$
Output: Optimal input variable set $\mathbf{X}_{pmi} = \mathbf{Z} \in \Re^{L\times N}$
procedure CMAL
Input:$\mathbf{X}_{pmi} \in \Re^{L\times N}$
    Select T data samples iteratively according to (12)
Output: Optimal training data $\mathbf{X} = \mathbf{X}_{cmal} \in \Re^{L\times T}$

We used the Breast Cancer Wisconsin (Diagnostic) data set [15] from the UCI data repository to evaluate the classification accuracy. There are altogether 569 data records. This data set has been used widely in many breast cancer detection studies. As shown in Table 1, a classifier named AIRS in [16] obtained an accuracy of 97.2%. A classifier named LS-SVM was proposed in [17] with an accuracy of 98.53%. RS-BPNN was used in [18] with an accuracy of 98.6%. More recently, an algorithm integrating unsupervised deep belief network (DBN) with supervised back propagation [19] achieved an accuracy of 99.68%.

Methods	**AIRS**	**LS-SVM**	**RS-BPNN**	**DBN**
Accuracy	97.2%	98.53%	98.6%	99.68%

Table 1. Performance of some classifiers in literature.

Methods	**T**	**Full**	**Random**	**PMI**
SVM_Full	N	37.3%	86.3%	91.8%
DNN_Full	N	95.2%	95.2%	96.8%
DNN_Random	0.5N	60.0%	72.9%	62.9%
DNN_PBAL	0.5N	87.1%	91.4%	94.3%
DNN_CMAL	0.5N	95.7%	95.7%	$\geq$99.9%

Table 2. Performance of the five classifiers compared in our simulation.

In our simulation, the training/testing ratio was set as 78%/22%. For the training data set, $N = 499$, $T = \frac{1}{2}N$, and $K = 30$.

Firstly, in order to evaluate the performance of CMAL, we simulated the five algorithms without turning on the PMI-based feature selection function. It can be seen from Table 2 that without feature selection, **SVM_Full** and **DNN_Full** achieved an accuracy of 37.3% and 95.2%, respectively. By randomly selecting T training samples from the overall N training samples, the accuracy of **DNN_Random** was only 60.0%. In contrast, selecting T training samples with active learning algorithms, **DNN_PBAL** and **DNN_CMAL** achieved the accuracy of 87.1% and 95.7%, respectively.

Next, we applied the PMI technique to select the optimal feature set, which reduced the number of features from 30 to 9 (K to L). Table 2 shows that the performance of all the five classifiers improved at different levels. **SVM_Full** and **DNN_Full** jumped to 91.8% and 96.8%, respectively. As for **DNN_PBAL**, it obtained an accuracy of 94.3%. **DNN_Random** achieved the lowest accuracy of 62.9%. In contrast, when selecting L (9) features out of K (30) features randomly, the accuracy of **DNN_Random**, **SVM_Full** and **DNN_PBAL** were increased to 72.9%, 86.3% and 91.4% individually. It is important to note that with PMI, our **DNN_CMAL** witnessed a dramatic improvement from 95.7% to over 99.9%. Figure 1 shows the excellent performance of our proposed classifier **DNN_CMAL** with PMI.

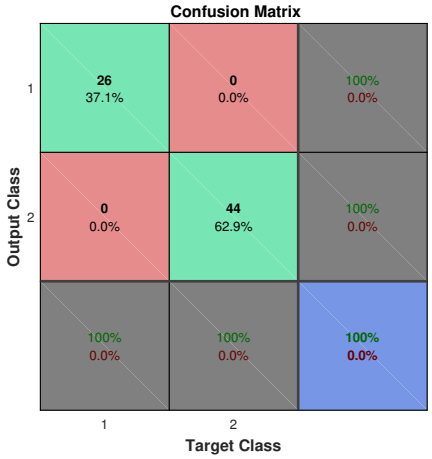

Fig. 1. Performance of the proposed DNN scheme with the PMI and CMAL techniques.

5. CONCLUSIONS

In this paper, we developed a novel scheme that reduces the training data amount necessary for DNN by applying the partial mutual information technique to select the optimal feature set and by applying the correlation matching based active learning technique to select the most informative training data. Simulations over the UCI breast cancer data set show that with only half of the original training data and 9 out of 30 features, our proposed scheme outperforms many other algorithms.

6. REFERENCES

[1] Yoshua Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[2] Yoshua Bengio, Aaron C Courville, and Pascal Vincent, "Unsupervised feature learning and deep learning: A review and new perspectives," *CoRR, abs/1206.5538*, vol. 1, 2012.

[3] Lei Yu and Huan Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *ICML*, 2003, vol. 3, pp. 856–863.

[4] Robert J May, Holger R Maier, Graeme C Dandy, and TMK Gayani Fernando, "Non-linear variable selection for artificial neural networks using partial mutual information," *Environmental Modelling & Software*, vol. 23, no. 10, pp. 1312–1326, 2008.

[5] Scott Doyle, James Monaco, Michael Feldman, John Tomaszewski, and Anant Madabhushi, "An active learning based classification strategy for the minority class problem: application to histopathology annotation," *BMC bioinformatics*, vol. 12, no. 1, pp. 424, 2011.

[6] Xiaohua Li and Jian Zheng, "Active learning for regression with correlation matching and labeling-error suppression," 2015.

[7] Masashi Sugiyama and Shinichi Nakajima, "Pool-based active learning in approximate linear regression," *Machine Learning*, vol. 75, no. 3, pp. 249–274, 2009.

[8] Sivan Sabato and Remi Munos, "Active regression by stratification," in *Advances in Neural Information Processing Systems*, 2014, pp. 469–477.

[9] Carlos Riquelme, Baosen Zhang, and Ramesh Johari, "Online active linear regression via thresholding," *arXiv preprint arXiv:1602.02845*, 2016.

[10] Xiaohua Li and Jian Zheng, "Joint machine learning and human learning design with sequential active learning and outlier detection for linear regression problems," in *2016 Annual Conference on Information Science and Systems (CISS)*. IEEE, 2016, pp. 407–411.

[11] Xiaohua Li, Yu Chen, and Kai Zeng, "Integration of machine learning and human learning for training optimization in robust linear regression," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2613–2617.

[12] Andrew Ng, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, pp. 1–19, 2011.

[13] Jun Xu, Lei Xiang, Renlong Hang, and Jianzhong Wu, "Stacked sparse autoencoder (ssae) based framework for nuclei patch classification on breast cancer histopathology," in *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2014, pp. 999–1002.

[14] Shotaro Akaho, "Conditionally independent component analysis for supervised feature extraction," *Neurocomputing*, vol. 49, no. 1, pp. 139–150, 2002.

[15] M. Lichman, "UCI machine learning repository," 2013.

[16] Donald E Goodman, Lois Boggess, and Andrew Watkins, "Artificial immune system classification of multiple-class problems," *Proceedings of the artificial neural networks in engineering ANNIE*, vol. 2, pp. 179–183, 2002.

[17] Kemal Polat and Salih Güneş, "Breast cancer diagnosis using least square support vector machine," *Digital Signal Processing*, vol. 17, no. 4, pp. 694–701, 2007.

[18] Kindie Biredagn Nahato, Khanna Nehemiah Harichandran, and Kannan Arputharaj, "Knowledge mining from clinical datasets using rough sets and backpropagation neural network," *Computational and mathematical methods in medicine*, vol. 2015, 2015.

[19] Ahmed M Abdel-Zaher and Ayman M Eldeib, "Breast cancer classification using deep belief networks," *Expert Systems with Applications*, vol. 46, pp. 139–144, 2016.