Original Research Article

Auto Encoder Fixed-Target Training Features Extraction Approach for Binary Classification Problems

ABSTRACT

The main issues with machine learning-based feature extraction techniques are the requirement of extensive domain-level knowledge, experience, and the need to be supported by large amounts of data that are sometimes not available. Moreover, it is often difficult to apply domain-level knowledge to extract the necessary features for building a machine-learning classifier. Therefore, it is significantly important to find and develop feature extraction techniques that depend mainly on the training data and don't require or depend on domain-level knowledge and experience. To address these issues for binary classification problems, a novel feature extraction approach, *AE-FT(Fixed Target)* for extracting common features using a Deep Belief Network (DBN)-based Autoencoder (AE) is proposed in this paper. In this approach, common features are extracted by a DBN trained on a dataset sample's binary using *the Fixed Target training approach.*

The proposed common features extraction approach is tested and evaluated on two different data sets. For each dataset, the extracted features are used to train seven of the common machine learning binary classification algorithms and compared their performances. Moreover, the number of extracted features is very small compared to other existing feature extraction methods. Therefore, the proposed common features extraction method improves the performance of the binary classification algorithms by reducing the number of features reducing laborious processes, and increasing the recognition accuracy effectively.

The results show that the proposed common features extraction approach, without any domain-level knowledge or human expertise, provides a very good performance compared to other feature extraction techniques.

Keywords: Features Extraction; Deep Learning; AE; Fixed-Target Training; Common Features.

1. INTRODUCTION

With the rapid development of machine learning technology, as a binary classification problem that helps people to find the law from the massive data to achieve the prediction effectively, data prediction has become an important part of people's daily lives. Feature extraction is a basic and important matter for the classification problem because the original data contain noise and irrelevant information which decreases the classification accuracy.

Feature extraction is about finding a good data representation, which is very domain-specific, often requires human expertise, and is related to available measurements. The primary idea behind feature extraction is to compress the data to maintain most of the relevant information.

As to feature selection techniques, these techniques are also used for reducing the number of features from the original feature set to reduce model complexity, and model overfitting, enhance model computation efficiency, and reduce generalization error. Therefore, improve the accuracy of the learning algorithm and shorten the training and output time.

The feature extraction methods are useful for different applications as mentioned in [1], such as social science, healthcare, environment, agriculture, spam filtering, antivirus technology, economics, medical diagnosis, face recognition, action recognition, speech recognition, gesture recognition, marketing, wireless network, gene expression, software fault detection, internet traffic prediction, etc. Therefore, the research of machine learning algorithms in feature extraction problems is a research hotspot in recent years.

The main issue with machine learning-based feature extraction techniques is the requirement of time, extensive domain-level knowledge, and experience as mentioned by Verdonck et al. [2].Moreover, it is often difficult to apply domain-level knowledge to extract the necessary features for building a machine-learning classifier. Therefore, it is significantly important to find and develop feature extraction techniques that depend mainly on the training data and don't require or depend on domain-level knowledge and experience, and this is our main purpose. Therefore, the focus of this paper is on using machine learning for common features extraction that can be used in binary classification in general, that applicable in many fields such as spam filtering, antivirus technology, and medical diagnosis.

This paper presents the use of denoising stacked autoencoders with supervised fixed-target training in order to extract the common features of the training data that can be used in binary classification. The method relies on training a deep belief network (DBN) [3], i.e., a deep unsupervised neural network implemented with a deep stack of denoising autoencoders, in a supervised manner to create an invariant compact representation of the general behavior of the training datasets. In recent years DBNs have proven successful in generating invariant representations for many challenging domains. We used only positive training samples for training the common features extractor. Then, we used the common features extractor for extracting the values of the common features of the training samples and used it for training the binary classifiers.

In contrast to most existing approaches that normally have a separate stage for data preprocessing followed by domain-dependent feature extraction. We developed a domain-independent deep neural network framework for common feature extraction which enables us to easily, without the need for domain-level knowledge or expertise, extract features that can be effectively used in binary classification problems.

We trained the proposed feature extractor, using the binary representation of the datasets, to extract the common features and then used it for training binary classification models. In the experiments, we used the extracted common features to build binary classifiers using seven binary classification methods Naive Bayes, Logistic Regression, K-Nearest Neighbors, Support Vector Machine, Decision Tree, Random Forest, and Voting Classification for binary classification.

The proposed method, fixed-target training of a deep stacked autoencoder, enabled a good recognition accuracy, better generalization, and more stability than that which could be achieved with the other methods. The proposed approach achieves 73.50% accuracy, which is so far a good result that does not need any domain expertise.

The remainder of this paper is organized as follows: The next section presents the study background**,** Section 3 describes our proposed approach, and Section 4 presents the experimental results. In section 5, we discussand evaluate the results of the study and present our conclusions in Section 6, while presenting some directions for future studies.

2. BACKGROUND

The success of machine learning often depends strongly on the success of feature extraction, the features used influence the result more than everything else. No algorithm alone can supplement the information gain given by correct feature extraction. So, feature extraction is a basic and important matter for the classification problem because the original data contain noise and irrelevant information which decreases the classification accuracy.

There are two broad categories for feature extraction algorithms: linear and nonlinear. Linear feature extraction assumes that the data lies in a linear subspace. Use matrix factorization to protect them. On the other hand, in nonlinear feature extraction ordimensionality reduction, a low-dimensional surface can be mapped into a high-dimensional space so that a nonlinear relationship among the features can be found and easily detected. Theoretically, a transformation function f(x) can be used to map the features into a higher-dimensional space and then mapped back into the lower-dimensional space, so that the relationship can be viewed as nonlinear. We focus on the nonlinear feature extraction algorithms.

2.1. Kernel Principal Component Analysis (KPCA)

KPCA introduced by Scholkopf et al. [4], is an extension of Principal Component Analysis (PCA) that allows for the separability of nonlinear data by making use of kernels. The basic idea behind it is to project the linearly inseparable data onto a higher dimensional space where it becomes linearly separable. Unfortunately, it has a serious limitation in terms of space complexity since it stores all dot products of the training set and therefore the size of the matrix increases quadratically with the number of data points as presented in [5].

Another drawback of the KPCA, however, is the cost of computation could be extremely high, which could lead to the attendant numerical problems of diagonalizing large matrices, which limits its applicability in many large dataset problems. But, an Expectation-Maximization (EM) algorithm for KPCA to overcome these drawbacks was proposed in [6], which is an expectation-maximization approach for performing kernel principal component analysis. Experimental results showed that EM is an efficient method computationally, especially for a large number of data points.

2.2. Locally Linear Embedding (LLE)

Locally Linear Embedding, proposed by Saul et al. [7], is a dimensionality reduction technique based on Manifold Learning that involves the computation of low-dimensional neighborhood preserving embeddings of inputs that are of high dimension in nature. Manifold Learning aims to make a manifold object, an object of D dimensions that is embedded in a higher-dimensional space, representable in its original D dimension instead of being represented in an unnecessarily greater space.

LLE has the ability to learn the global structure of nonlinear manifolds like those from images of faces or documents of text by exploiting the local symmetries of linear reconstructions. LLE has been applied successfully in a wide range of applications which includes face recognition and remote sensing, MRI, shape analysis of the hippocampus in AD, diffusion tensor imaging, breast lesion segmentation, feature fusion, and image classification according to [8].

LLE is popular among researchers because of its ability to deal with large data sets of high-dimensional data and its non-iterative way of finding embeddings. However, it has some drawbacks which include sensitivity to noise, the inability to deal with novel data, and the inevitable ill-conditioned Eigen problems. Some efforts have recently been made to develop extensions of the classical LLE.

Supervised and semi-supervised versions of LLE were proposed by [9] and [10], respectively, for plant classification based on images of leaves.

2.3. Linear Discriminant Analysis (LDA)

LDA is a supervised learning dimensionality reduction, feature extraction technique, and Machine Learning classifier that was invented by Fisher et al. [11]. LDA uses within-classes and between-classes measures by maximizing the distance between the mean of each class and minimizing the spreading within the class itself. This is a good choice because maximizing the distance between the means of each class when projecting the data in a lower-dimensional space can lead to better classification results.

An advantage of LDA is that it is able to use information from both features to create a new axis which in turn minimizes the variance and maximizes the class distance of the variables. Although the LDA is one of the most well-used data reduction techniques, it has some limitations. The small sample problem (SSS), is one of the main problems of LDA, which happens when the dimensions are much higher than the number of samples in the data matrix, LDA is unable to find the lower dimensional space resulting in the within-class matrix becoming singular. Different approaches have been proposed to solve this problem, such as what was proposed in [12] and [13]. In addition to the assumption that the input data follows a Gaussian Distribution, therefore applying LDA to not Gaussian data can lead to poor classification results.

A semisupervised variant of LDA, which performed better than the classical LDA, was proposed by Zhang et al. [14] that mainly combines both labeled and unlabeled data for training LDA and allows using LDA for the situation where the labeled data are few.

Application of LDA includes facial recognition, text recognition, automatic diagnosis of machine operations, early detection of diseases, person reidentification, hand movement classification, motor imagery EEG, and groundwater redox conditions.

2.4. t-distributed Stochastic Neighbor Embedding (t-SNE)

t-Stochastic Neighbor Embedding (t-SNE) is an unsupervised Non-linear Dimension Reduction Technique (NLDRT) that was introduced by Maaten et al. [15]. The technique is a variation of the Stochastic Neighbor Embedding introduced by Hinton et al.[10], whose main objective is the construction of probability distributions from pairwise distances such that larger distances correspond to smaller probabilities and vice versa. t-SNE is typically used to visualize high-dimensional datasets, it works by minimizing the divergence between a distribution constituted by the pairwise probability similarities of the input features in the original high-dimensional space, which is modeled using a Gaussian Distribution and its equivalent in the reduced low-dimensional space, modeled using a Student's t-distribution.

t-SNE makes use of the Kullback-Leiber (KL) divergence in order to measure the dissimilarity of the two different distributions, as mentioned in [16]. The KL divergence is then minimized using gradient descent. t-SNE is the most commonly used in single-cell analysis. However, it has some limitations as mentioned in [17]. The limitations include slow computation time, the inability to meaningfully represent very large datasets, and the loss of large-scale information.

2.5. Deep Learning Approach

The major difference between deep learning and traditional pattern recognition methods is that deep learning automatically learns features from big data, instead of adopting handcrafted features, as stated in [18]. Deep learning is able to quickly acquire new effective feature representations from training data.

In recent years DBNs [3], deep unsupervised neural networks, have proven successful in generating invariant representations for many challenging domains. Autoencoders are feed-forward DBNs that were first introduced by Rumelhart et al. [19]. They can learn a compressed and distributed representation of data, which can be used as a dimensionality reduction or feature extraction technique. They use nonlinear transformations to project data from a high dimension to a lower one. An autoencoder usually has at least one hidden layer between the input and output layers. The number of neurons in the hidden layer is usually set to less than those in the input and output layers, thus creating a bottleneck, with the intention of forcing the network to learn a higher-level representation of the input as presented in [20].

Autoencoders are typically trained in an unsupervised manner, using backpropagation with stochastic gradient descent, to approximate a function by which data can be classified, as mentioned in [21]. For every training input, the difference between the input and the output is measured (using squared error) and it is back-propagated through the neural network to perform weight updates on the different layers.

Compared with other machine learning methods, deep learning is able to detect complicated interactions in features, learning lower-level features from nearly unprocessed original determine characteristics that are not easy to be detected. Furthermore, they hand class members with high cardinal numbers and process untapped data. Unfortunately, if all input features are independent of each other, then the autoencoder will find it particularly difficult to encode the input data into a lower-dimensional space.

The advantages are higher discriminating power and control overfitting when it is unsupervised. On the other hand,there are some bottlenecks in deep learning based feature extraction methods, time-consuming data pre-processing, domain expertise, the need for large amounts of data, loss of data interpretability, and transformation may be expensive.

There are many other deep learning-based feature extraction approaches. A skeleton-based abnormal gait recognition approach was proposed by Jun et al. [22]. They proposed a feature extraction method using the RNN AEs to minimize the irrelevant information of the original skeleton data. They used two-step training of a hybrid RNN and AE-DM model and approved that it is more effective than the single-step training of the End-to-End model that has the same data flow. Ma and Yuan [23] proposed a method for extracting features from images based on deep CNN and PCA. They used a neural network to extract features and a PCA algorithm for feature dimension reduction. Then they compared the performance of the PCA before and after the improvement claim achieving memory, and time optimization. Moreover, the SVM classifier accuracy was enhanced. Dahouda et al. [24] proposed a deep learning-based feature extraction approach with a modular neural network, they employed a pre-trained neural architecture search net (NASNet) as a feature extractor on a custom dataset of raw copper and cobalt image. Then, they used the extracted features to build a deep neural network and machine learning algorithms for the image classification of copper and cobalt raw minerals. However, it is an empirical not an exhaustive study, and the data preprocessing was ignored.Petrovska et al. [25] used pre-trained neural networks to extract features, then applied PCA to reduce the dimensionality of the extracted features. However, pre-trained neural networks were used in both [24] and [25]. Morever, these are domain-dependent approaches that work only for the specific domain not for binary classification problems in general.

Table 1. Comparison of Feature Extraction Methods.

Feature Extraction Technique	Domain level Knowledge	Data preprocessing	Limitations
Kernel PCA	yes	yes	space complexity
LLE	yes	yes	sensitivity to noise, the inability to deal with novel data and the inevitable ill-conditioned Eigen problems
LDA	yes	yes	small sample problem (SSS)
t-SNE	yes	yes	slow computation time
AE	often	yes	Loss of data interpretability Transformation may be expensive

The main issue with machine learning-based feature extraction techniques is the requirement for extensive domain-level knowledge and experience. Moreover, it is often difficult to apply domain-level knowledge to extract the necessary features for building a machine-learning classifier. Table 1 shows that all the current approaches have the same problems. Therefore, it is significantly important to find and develop feature extraction techniques that depend mainly on the training data and don't require or depend on domain-level knowledge and experience, and this is our main purpose.

3. METHODOLOGY

In this section, we describe and discuss our proposed novel deep learning-based approach for common feature extraction, our datasets, and training methods in detail. The main question we are trying to answer is the following:

Is it possible to extract the common features from the raw binary representations without any domain expertise of a given dataset that could be used in binary classification?

In recent years, deep learning methods have proven very successful in accomplishing dimensionality reduction and feature extraction tasks in many domains, especially computer vision, and cybersecurity according to [27, 28, 29, 30]. The proposed methodology works as follows:

Firstly, training the proposed common features extraction model to extract features to use in binary classification using various common machine learning classification algorithms. Secondly, select seven binary classification algorithms and use the extracted common features to build binary classifiers using common algorithms.

Our method uses stacked denoising autoencoders for extracting the common features of the training dataset that can be used effectively in binary classification. The input to the DBN has a fixed length, but the dataset sample length is variable. In order to represent the dataset binary as a fixed-sized vector, which would be the input to the neural network, we repeatedly pad the sample binary until the specified size is met. This process gives better results than 0's or 1's padding that the DBN learns as a common feature. Then we use two different datasets, the IMDB dataset [31] and the Enron-Spam dataset [32], to train and test the proposed feature extractor for extracting the common features and then used for training binary classification models using the extracted features of the binary representation of the datasets. We focus on the top seven most common binary classification algorithms Naive Bayes, Logistic Regression, K-Nearest Neighbours, Support Vector Machine, Decision Tree, Random Forest, and Voting Classification.

3.1. Deep Belief Network Fixed-Target Training

In stacked denoising autoencoders, first introduced by Vincent et al. [26], the data at the input layer is replaced by noised data while the data at the output layer stays the same; therefore, the autoencoder can be trained with much more generalization power, according to [18].

Usually, denoising autoencoder training is unsupervised according to Vincent et al. [26]. The input sample is corrupted by adding noise (or more often by zeroing the values). That is, given an input $\mathbf{S}$, first it is corrupted to $\mathbf{S}^{\wedge}$ and then fed to the input layer of the network. The objective function of the network in the output layer remains to generate the

uncorrupted version of the input (see Figure 1-(a)). But in our approach, we use a novel supervised training strategy, which we call a *fixed-target training* strategy.

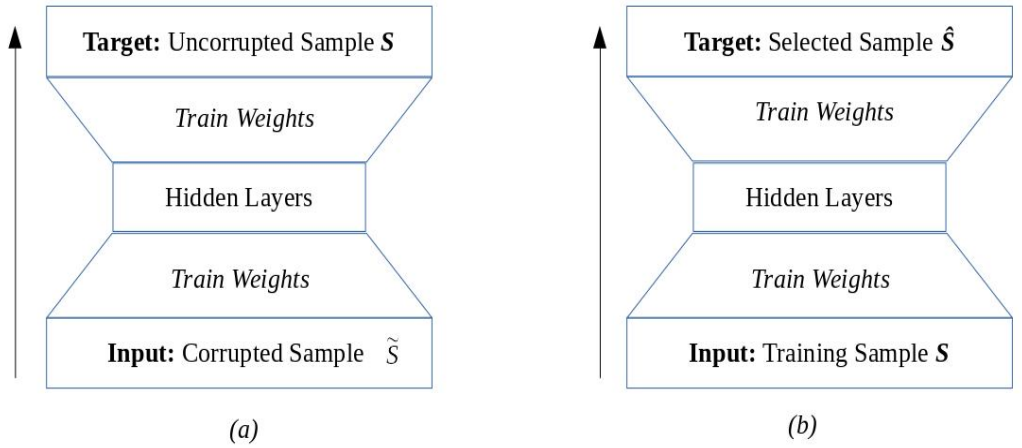

Fig. 1. Comparison between Denoising autoencoder training. (a)-traditional unsupervised training, (b)-fixed-target supervised training.

In the *fixed-target training* strategy, we randomly select one of the training samples $\mathbf{S}^{\wedge}$, fix it in the output layer, and for every input $\mathbf{S}^{\wedge}$ of the training samples, the objective function of the network is to generate $\mathbf{S}^{\wedge}$, (i.e. we consider all training samples as corrupted versions of the selected sample $\mathbf{S}^{\wedge}$ (see Figure 1-(b)). So, the "hidden units" of DBN compute internal representations analogous to the extracted common features.

This training approach works better than traditional training. By fixing the target output, the network is forced to generalize better and determine more high-level common patterns. Moreover, the network learns better even when few training samples are available. When a DBN's training is complete, we discard the decoder layer, fix the values of the encoder layer, and use the encoder as the common features extractor. In a typical implementation, the extracted features may then be used for supervised binary classification.

In order to achieve our goal, we create a DBN by training a deep stack of denoising autoencoders. We use fixed-target training to train a deep denoising autoencoder consisting of five layers: 8,192–2,048–512–128–32. At the end of this training phase, we have a deep network that is capable of converting 8,192 length input vectors into 32 floating point feature values. Note that the network is trained only using the samples in the training set, and for all future samples it will be run in prediction mode;(i.e., receiving the 8,192-sized vector it will produce 32 output values, without modifying the weights). See Figure 2.

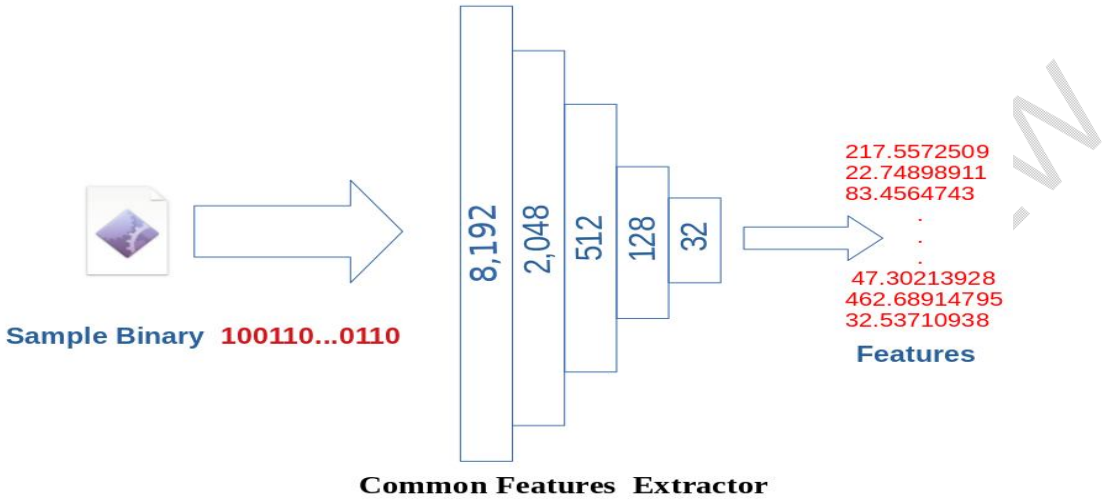

Fig. 2. Illustration of the common features extraction stages from feeding the sample binary to features extraction using DBN.

Our goal is to train the proposed feature extractor for extracting the common features and then used it for training binary classification models using the extracted features of the binary representation of the datasets. We focus on the top seven most common binary classification algorithms Naive Bayes, Logistic Regression, K-Nearest Neighbours, Support Vector Machine, Decision Tree, Random Forest, and Voting Classification. The sample's binary bit string is fed to the neural network, and the deep neural network generates a 32-sized vector at its output layer, which we treat as the common feature values of the sample.

3.2. Datasets

We are using two different datasets, the Internet Movie Database (IMDB) dataset [31] and the Enron-Spam dataset [32], for testing our approach. The IMDB dataset, Large Movie Review Dataset v1.0, of highly polar movie reviews in the form of text comments on different movies and a positive or negative score. This dataset contains movie reviews along with their associated binary sentiment polarity labels. It is intended to serve as a benchmark for sentiment classification. The core dataset contains 50,000 reviews split evenly into 25k train and 25k test sets. The overall distribution of labels is balanced (25k pos and 25k neg). Furthermore, the train and test sets contain a disjoint set of movies, so no significant performance is obtained by memorizing the movie's unique terms and they are associated with the observed labels. In the labeled train/test sets, a negative review has a score < 4 out of 10, and a positive review has a score > 7 out of 10. Thus, reviews with more neutral ratings are not included in the train/test sets.

These datasets have been used as a benchmark in several research papers on spam filtering, text classification, and natural language processing as mentioned in [33, 34]. Therefore, the results of this work are hopefully comparable to other similar works within the area, without having to account for unique datasets.

3.3. *Common Features Extraction Method*

As described in the previous section, we use a fixed-target training strategy to train a deep denoising autoencoder consisting of five layers (8,192–1,024–512–128–32). We randomly pick one of the training samples as a pivot sample, and each time a new input is given to the network we put the pivot sample in the output as the target. This way, we enforce the autoencoder to learn the common features of the training samples.

Fixed-target training is essential for the network to learn common features in a very small time and even when few samples are available. We use rectified linear units (ReLU) for the nonlinearity function when training deep neural networks to diminish the gradient vanishing problem and result in faster convergence, as approved in [1, 20].

We build the model using machine learning algorithms in *the Keras library*. Other parameters we use are 50 training epochs, a learning rate of 0.003, a batch size of 10, and no momentum. Figure 3 shows the learning curves, which approve the feasibility of our proposed common feature extraction approach.

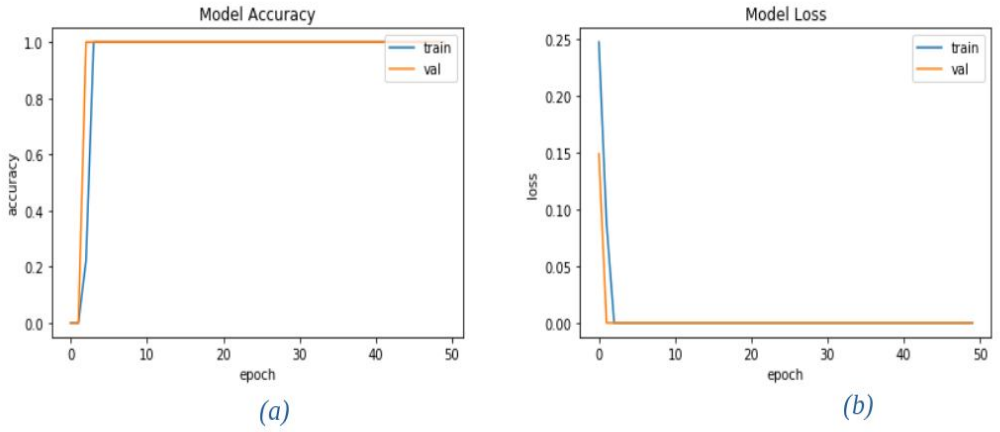

Fig. 3. Common Features Extraction Model training and validation. (a)-Accuracy, (b)-Loss

EXPERIMENTAL RESULTS

In this section, we conducted various experiments to evaluate the features of the DBN AEs for binary classification. First, we compared the number of features used in [31, 32] and the extracted common features. Then, we conducted experiments to show the effectiveness of the proposed DBN AEs. Finally, we showed the performance of the features of the DBN AEs used in binary classification problems, we used seven binary classification methods Naive Bayes, Logistic Regression, K-Nearest Neighbours, Support Vector Machine, Decision Tree, Random Forest, and Voting Classification- for binary classification.

In the experiments, we trained a deep denoising autoencoder consisting of five layers, 8192–2048–512–128–32, using a Fixed-Target training strategy. We used only positive training samples for training the common features extractor. Then, we used the common features extractor for extracting the values of the common features of the training samples and used it for training the binary classifiers.

In the testing phase, we first extract the values of the common features of the

Fig. 4. Overview of our deep learning approach for common features extraction, illustration of all stages from feeding the sample binary to the Common Features Extractor ending with the decision made by the binary classifier.

sample and pass them to the binary classifier for classification. See Figure 4.

We used the training score and test score to evaluate our model, the test score measures how well our model learn from our training data, while the test score measures the accuracy. Higher the test score better the model is generalized. The results demonstrate that, with proper structure and parameters, the performance of the proposed deep learning method on common feature extraction is useful even in the lack of domain expertise in binary classification.

Table 2. Performance of Binary Classification Methods, trained by the common features on IMDB and Enron-Spam datasets.

Binary Classification Method	IMDB Dataset		Enron-Spam Dataset	
	Training Score	Test Score	Training Score	Test Score
Naive Bayes	63.0%	60.7%	63.0%	66%
Logistic Regression	88.0%	73.5%	87.0%	73%
K-Nearest Neighbours	87.3%	73.2%	88.7%	73.2%
Support Vector Machine	67.0%	66.3%	67.3%	67%
Decision Tree	100%	70.5%	100%	72%
Random Forest	93.5%	73.4%	93.5%	73.5%
Voting Classification	85.0%	72.7%	84.7%	73%

The accuracy results are given in Table 2, test samples are 73.5% correctly classified by conducting a Logistic Regression classifier for the proposed approach with the extracted common features of the IMDB dataset. For the Enron-Spam dataset, the best results are obtained by Random Forest 73.5%. The binary classification performances of the approaches are given in Table 2. According to the results, the common features extracted by

the proposed approach give a very good performance for all seven binary classification methods. Logistic Regression and Random Forest are the most effective classifiers for this approach.

Compared with other methods, the proposed approach fixed-target training of a denoising deep stacked autoencoder achieves 73.50% accuracy, which is very good for binary classification without any need for domain-level expertise or data preprocessing.

5. DISCUSSION AND EVALUATION

We use the proposed fixed-target training strategy to train a deep denoising autoencoder consisting of five layers (8,192–1,024–512–128–32). We randomly pick one of the training samples as a pivot sample, and each time a new input is given to the network we put the pivot sample in the output as the target. This way, we enforce the autoencoder to learn the common features of the training samples. Unfortunately, it is not easy to find the optimal autoencoder structure.

We trained and tested, using two different datasets, the proposed feature extractor for extracting the common features. Then, we used the extracted features for training seven of the most common binary classification algorithms Naive Bayes, Logistic Regression, K-Nearest Neighbours, Support Vector Machine, Decision Tree, Random Forest, and Voting Classification.

The learning curves in Figure 3 approve the feasibility of our proposed common feature extraction approach. According to the results reported in Table 2, the best accuracy values are obtained by conducting Logistic Regression and Random Forest classifiers. When comparing the number of features used, 32 common features were extracted using the proposed feature extraction algorithm, on the other hand, 3000 and 10000 features were used in [31] and [32], respectively, in addition to the preprocessing carried to extract the features used in [31] and [32]. However, by applying the proposed common features extraction algorithm, it is observed that high success rates are achieved with very few features and increase the overall process performance.

6. CONCLUSION

In this paper, we reviewed past approaches for feature extraction and proposed a novel method based on deep belief networks for common features extraction. Current approaches for feature extraction are time-consuming and require extensive domain-level knowledge and experience. Therefore, it is significantly important to find and develop feature extraction techniques that depend mainly on the training data and don't require or depend on domain level knowledge and experience.

Our proposed feature extraction approach, AE fixed-target supervised training, extracts the common characteristics of the original data and minimizes the irrelevant information without the need for any domain-level expert or expensive data preprocessing. Furthermore, it needs less training data and produces fewer features compared to other feature extraction techniques. Therefore, the extracted features improve the binary classification performance. The most interesting result of our evaluation was the very good

performance of our common features extension approach using all binary classification algorithms tested on both datasets that have been used, but further experiments are needed to be more confident.

Future work could include examining how to improve the accuracy of the proposed common feature extraction method by finding the optimal autoencoder structure and activation function. Furthermore, examining different ways for the pivot sample selection. Of course, we could also look at different types of classifiers and use different datasets in different domains. One could also see, whether it is useful to use the proposed method in data collection; when only a few samples are available. Finally, it would be interesting to see if the proposed method could mitigate the problem of the need for domain expertise and data preprocessing for feature extraction and the need for a big dataset to train binary classifiers.

Data Availability: Tow Datasets are used, The IMDB and Enron-Spam data supporting this research are from previously reported studies and datasets, which have been cited. The processed data are available at http://www.cs.cornell.edu/people/pabo/movie-review-data/ and http://www2.aueb.gr/users/ion/data/enron-spam/ , respectively.

ACKNOWLEDGEMENTS

Funding: The study attracted no funding.

Conflict of Interest: All authors declare that they have no conflict of interest.

REFERENCES

1. Ghojogh, B., Samad, M. N., Mashhadi, S. A., Kapoor, T., Ali, W., Karray, F., & Crowley, M. (2019). Feature selection and feature extraction in pattern analysis: A literature review. *arXiv preprint arXiv:1905.02845*.
2. Verdonck, T., Baesens, B., Óskarsdóttir, M., & vanden Broucke, S. (2021). Special issue on feature engineering editorial. *Machine Learning*, 1-12.
3. Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, *18*(7), 1527-1554.
4. Schölkopf, B., Smola, A., & Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*,*10*(5), 1299-1319.
5. Liu, X., & Yang, C. (2009, October). Greedy kernel PCA for training data reduction and nonlinear feature extraction in classification. In *MIPPR 2009: Automatic Target Recognition and Image Analysis*(Vol. 7495, pp. 768-775). SPIE.
6. Rosipal, R., & Girolami, M. (2001). An expectation-maximization approach to nonlinear component analysis.*Neural Computation*, *13*(3), 505-510.
7. Saul, L. K., & Roweis, S. T. (2003). Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of machine learning research*, *4*(Jun), 119-155.

8. Nanga, S., Bawah, A. T., Acquaye, B. A., Billa, M. I., Baeta, F. D., Odai, N. A., ... & Nsiah, A. D. (2021). Review of Dimension Reduction Methods. *Journal of Data Analysis and Information Processing*, *9*(3), 189-231.

9. Kouropteva, O., Okun, O., & Pietikäinen, M. (2005, June). Incremental locally linear embedding algorithm. In *Scandinavian Conference on Image Analysis*(pp. 521-530). Springer, Berlin, Heidelberg.

10. Pan, Y., Ge, S. S., & Al Mamun, A. (2009). Weighted locally linear embedding for dimension reduction. *Pattern Recognition*, *42*(5), 798-811.

11. Xanthopoulos, P., Pardalos, P. M., & Trafalis, T. B. (2013). Linear discriminant analysis. In *Robust data mining*(pp. 27-33). Springer, New York, NY.

12. Sharma, A., & Paliwal, K. K. (2012). A new perspective to null linear discriminant analysis method and its fast implementation using random matrix multiplication with scatter matrices. *Pattern Recognition*, *45*(6), 2205-2213.

13. Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, *19*(7), 711-720.

14. Zhang, Y., & Yeung, D. Y. (2011). Semisupervised generalized discriminant analysis. *IEEE Transactions on Neural Networks*, *22*(8), 1207-1217.

15. Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, *9*(11).

16. Hinton, G. E., & Roweis, S. (2002). Stochastic neighbor embedding. *Advances in neural information processing systems*, *15*.

17. Xie, B., Mu, Y., Tao, D., & Huang, K. (2011). m-SNE: Multiview stochastic neighbor embedding. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *41*(4), 1088-1096.

18. Wang, H., & Raj, B. (2017). On the origin of deep learning. *arXiv preprint arXiv:1702.07800*.

19. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). *Learning internal representations by error propagation*. California Univ San Diego La Jolla Inst for Cognitive Science.

20. David, O. E., & Netanyahu, N. S. (2015, July). Deepsign: Deep learning for automatic malware signature generation and classification. In *2015 International Joint Conference on Neural Networks (IJCNN)*(pp. 1-8). IEEE.

21. Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504-507.

22. Jun, K., Lee, D. W., Lee, K., Lee, S., & Kim, M. S. (2020). Feature extraction using an RNN autoencoder for skeleton-based abnormal gait recognition. *IEEE Access*, *8*, 19196-19207.

23. Ma, J., & Yuan, Y. (2019). Dimension reduction of image deep feature using PCA. *Journal of Visual Communication and Image Representation*, *63*, 102578.

24. Dahouda, M.K. & Joe, I. (2022). Neural Architecture Search Net-Based Feature Extraction With Modular Neural Network for Image Classification of Copper/ Cobalt Raw Minerals. IEEE Access, 10, pp. 72253-72262. DOI: 10.1109/ACCESS.2022.3187420.

25. Petrovska, B., Zdravevski, E., Lameski, P., Corizzo, R., Štajduhar, I., & Lerga, J. (2020). Deep learning for feature extraction in remote sensing: A case-study of aerial scene classification. *Sensors*, *20*(14), 3906.

26. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P. A., & Bottou, L. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, *11*(12).

27. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84-90.

28. Lee, H., Ekanadham, C., & Ng, A. (2007). Sparse deep belief net model for visual area V2. *Advances in neural information processing systems*, *20*.

29. Le, Q. V. (2013, May). Building high-level features using large scale unsupervised learning. In *2013 IEEE international conference on acoustics, speech and signal processing*(pp. 8595-8598). IEEE.

30. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

31. Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011, June). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*(pp. 142-150).

32. Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006, July). Spam filtering with naive bayes-which naive bayes?. In *CEAS*(Vol. 17, pp. 28-69).

33. Eyheramendy, S., Lewis, D. D., & Madigan, D. (2003, January). On the naive bayes model for text categorization. In *International workshop on artificial intelligence and statistics*(pp. 93-100). PMLR.

34. Gómez-Hidalgo, J. M., López, M. J. M., & Sanz, E. P. (2000). Combining text and heuristics for cost-sensitive spam filtering. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.