Dimensionality Reduction Project Report

Hongkai Zheng Yiming Liu Yang Zhou Yakai Wang

1. Methods

1.1. Feature Selection

In machine learning, feature selection [1] is the process of selecting a subset of relevant features for use in model construction. Feature selection techniques are used for two reasons, to avoid the curse of dimensionality and enhanced generalization by reducing overfitting.

In this section, we try to obtain the best subset of features, with both Relief-F algorithm and random forest method.

1.1.1 Relief-F algorithm

Relief-F algorithm is an update to Relief algorithm to handle multi-class data. The key idea of Relief-F is to estimate attributes according to how well their values distinguish among the instances that are near to each other. For that purpose, given an instance, RELIEF searches for its two nearest neighbors: one from the same class (called nearest hit) and the other from a different class (called nearest miss). [5]

The pseudo code of Relief-F algorithm is shown below. The function $diff(A; R; H)$ calculates the Euclidean distance between instances to find the nearest neighbors.[8] For continuous attributes the difference is the actual difference normalized to the interval $[0, 1]$. The weights are estimates of the quality of attributes.

Algorithm 1: Relief-F algorithm

Input: Train data set D, number of samples to calculate m, number of neighbors to reserve k

Output: weight of features, $W(A)$

set all weights $W(A) = 0.0$ **for** $i = 1$ *to* m **do**
  randomly choose a sample R from D;
  find k nearest hits $H_j(j = 1, 2, \cdots, k)$;
  find k nearest misses $M_j(C)$;
  for $A = 1$ *to* N *All* $feature$ **do**
    $a = \frac{\sum_{j=1}^{k} diff(A,R,H_j)}{mk}$;
    $b = \sum_{C} \frac{\frac{p(C)}{1-p(Class(R))} \sum_{j=1}^{k} diff(A,R,M_j(C))}{mk}$ $(C \notin class(R))$;
    $W(A) = W(A) - a + b$;
return $W(A)$;

1.1.2 Random Forest Method

Random Forests [2] are also used for feature selection. The reason is because the tree-based strategies used by random forests naturally ranks by how well they improve the purity of the node. By pruning trees below a particular node, we can create a subset of the most important features.

1.2. Feature Projection

We tried the three methods mentioned in the class: Principle Component Analysis (PCA), Linear Discriminative Analysis (LDA), and Kernel PCA. However, due to the computer memory limitations, we could not implement the third method. Therefore, we only introduce the first two methods, namely linear methods in this section.

1.2.1 PCA

PCA is a statistical procedure which uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables, called principal components. This transformation is defined in such a way that the first principal component has the largest possible variance, and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components.A graphical representation is shown as follows:

The basic step of PCA is shown as follows. Assume the sample set $D = \{x_1, x_2, \ldots, x_m\}$, and we the expected dimension is d.

1. Centralize all samples: $x_i = x_i - \frac{1}{m} \sum_{i=1}^{m} x_i$;
2. Calculate the covariance matrix: $W = \frac{X^T X}{n-1}$;
3. Do the eigenvalue decomposition of covariance matrix: W;
4. Choose the d biggest eigenvalues,and the corresponding eigenvectors are: $W^* = w_1, w_2, \ldots, w_d$;

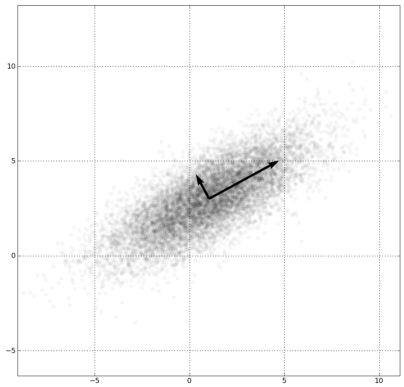

Figure 1. A instance for PCA

5. Convert samples to a low-dimensional space: $W^{*T}x_i$.

1.2.2 LDA

LDA is a generalization of Fisher's linear discriminant, a method used in statistics, pattern recognition and machine learning to find a linear combination of features that characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier, or, more commonly, for dimensionality reduction before later classification.

The objective we expect is to maxmize the class separation as well as minimize the distance between a class. For the property of the rank, the highest dimension we can reduce to is $n-1$, n is the number of the classes. The basic idea of LDA can be shown as follows:

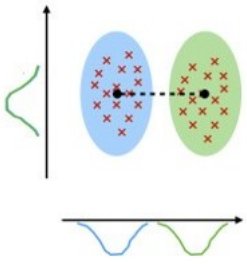

Figure 2. The basic idea of LDA

1.3. Feature Learning

In this project, we progressively investigated three feature learning methods to reduce dimensionality: Autoencoder, Variational Autoencoder, and $\beta-$Variational autoencoder.

1.3.1 Autoencoder

An autoencoder is a neural network consists of two parts: an encoder function $\mathbf{z} = g_\phi(x)$ encodes the input values x and a decoder that produces a reconstruction $\mathbf{x\prime} = f_\theta(\mathbf{z})$. The architecture is presented in figure 3.

In this project, we build the encoder with three fully connected layers: $2048 \times 1024, 1024 \times 512, 512 \times \dim(z)$ and decoder with :$\dim(z) \times 512, 512 \times 1024, 1024 \times 2048$. Autoencoder's objective is to minimize reconstruction error. This helps autoencoder learn important features present in the data. When a representation allows a good reconstruction of its input then it has retained much of the information present in the input. Here, we use mean square error between the input and output:

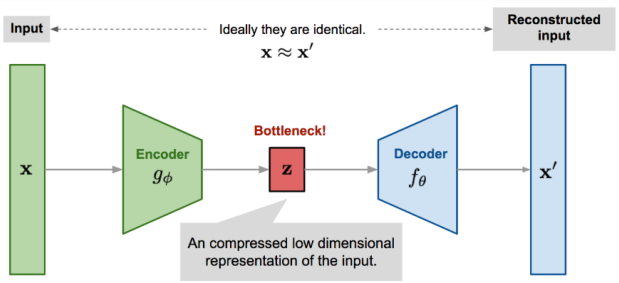

Figure 3. Architecture of Auto-encoder

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_i')^2 \tag{1}$$

1.3.2 Variational Autoencoder

The idea of Variational Autoencoder [4] deeply rooted in the methods of variational bayesian and graphical model. However, the structure looks a lot like an autoencoder as shown in figure 4. $p_\theta(\mathbf{x}|\mathbf{z})$ is a probabilistic decoder and $q_\phi(\mathbf{z}|\mathbf{x})$ plays a role as a probabilistic encoder. The reparameterization trick used in encoder to propagate the gradient.

We build the encoder with three fully connected layers: $2048 \times 1024, 1024 \times 512, 512 \times 2\dim(z)$ and decoder with :$\dim(z) \times 512, 512 \times 1024, 1024 \times 2048$.

The loss function of VAE consists of two parts: reconstruction loss and the Kullback-Leibler divergence between the encoders distribution $q_\theta(z|x)$ and $p(z)$, where $p(z)$ is specified as a standard Normal distribution, i.e., $\mathcal{N}(0, I)$, where I is the identity matrix.

$$L_{\text{VAE}}(\theta,\phi) = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}p_\theta(\mathbf{x}|\mathbf{z}) + D_{\text{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})\right) \tag{2}$$

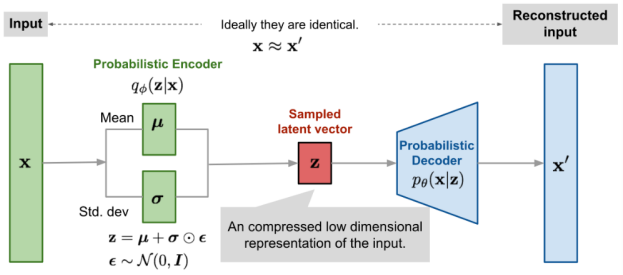

Figure 4. Variational autoencoder model with the multivariate Gaussian assumption.

1.3.3 β-VAE

If each variable in the inferred latent representation z is only sensitive to one single generative factor and relatively invariant to other factors, this representation is called disentangled or factorized. One benefit that often comes with disentangled representation is good interpretability and easy generalization to a variety of tasks.

β-VAE [3] is a modification of Variational Autoencoder with a special emphasis to discover disentangled latent factors. β-VAE follows the same structure of VAE: the encoder has three fully connected layers: $2048 \times 1024, 1024 \times 512, 512 \times 2\dim(z)$ and decoder also has three fully connected layers :$\dim(z) \times 512, 512 \times 1024, 1024 \times 2048$.But it introduce the Lagrangian multiplier β to control the encoding representation capacity. The loss function of β-VAE is as follows:

$$L(\phi, \beta) = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) + \beta D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})\right) \tag{3}$$

2. Experiments

2.1. Dataset and Feature extraction

In this experiment, we will train "animals with attributes 2"(AWA2) dataset [6, 7]. It consists of 37322 images of 50 animals classes with pre-extracted feature presentations for each image. The classes are aligned with Osherson's classical class/attribute matrix.

2.2. Experiment Set-up

We have to train SVM model to compare performance between different dimensionality reduction methods. To find the best parameters(e.g. penalty parameter C) when training SVM, we adopt grid search method and k-fold cross validation. By using grid search method, we search best C among seven numbers generated evenly in the log space from 2^{-15} to 2^{-1}. In k-fold cross validation, we set k to 5.

Method	Relief-F Algorithm
number of samples	50
number of neighbors to reserve	10

Table 1. parameter assignment in random forest method

Method	Random Forest
number of trees in the forest	200
maximum depth of the tree	1085
minimum of samples to be at a node	5
minimum of samples to split a node	23

Table 2. parameter assignment in random forest method

2.3. Feature Selection

To put Relief-F algorithm in life, we must decide two hyper-parameters at first. One is number of samples to be chosen randomly, in other word is number of iterations in the algorithm. The other is number of neighbors to reserve, it decides the size of H and $M(C)$ when calculating weight funciton. The setting of parameters is shown in table 1.

Similarly, To obtain reliable result, the necessary step is to adjust the parameters of random forest so that the model fits data well. Table2 shows all the details.

Relief-F algorithm The final size of feature subset relies on threshold δ that decides which feature should leave. In the experiment, we set it to $0.3000(64d)$, $0.2273(128d)$, $0.1442(256d)$, $0.8190(512d)$ and record score it returns. Score grows as dimension increases, shown in table 2.3. When dimension comes to 512, score reaches its best, 0.926.

Dimension	64	128	256	512
Score	0.814	0.877	0.909	0.926

Table 3. Score of Relief-F Algorithm in Different Dimensions

Random Forest Random forest approach only returns the importance of features, just like $W(A)$ in Relief-F algorithm. For convenience to compare with Relief-F algorithm, the number of reserved dimension is same to Relief-F algorithm. And detailed result is shown in table2.3. The best score turn out to be 0.930, when dimension is $512d$. From function $sklearn.feature_selection.SelectFromModel()$ in $sklearn$ toolbox we obtain the number of dimension, as well as the same score.

Dimension	64	128	256	512
Score	0.880	0.907	0.921	0.930

Table 4. Score of Random Forest method in Different Dimensions

2.4. Conclusion in Feature Selection

Figure 5 compares the results of both methods. As we can see from the figure, when dimension is low both two methods can not reach good results. The result turns out to be better as dimension grows. The best accuracy result is 0.9298 using random forest approach with dimension 512.

Both random forest method and Relief-F algorithm are supervised method. However in experiment, random forest seems to be a better way. The reason is that it overcomes the problem that Relief-F algorithm encounters. Relief-F method only picks features those can differentiate between instances from different classes, but it can not find features with high correlation, resulting in feature redundancy. In random forest method, it picks features of most importance in deciding classes of data, finally reduces feature redundancy in a way. But it may result in overfitting if we don't control number of dimensions. Both two methods achieve high enough accuracy, and prove to be effective approaches.

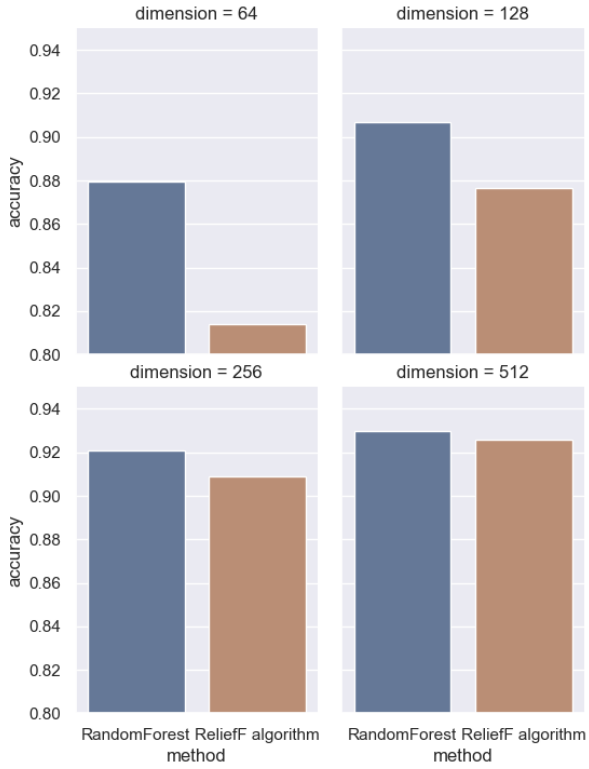

Figure 5. Comparison between Random Forest Method and Relief-F Algorithm

2.5. Feature Projection

We implement the experiments by *sklearn*.

For the comparison between different methods used in the project, we use PCA to respectively reduce the dimension to 512, 256, 64, 32 and 16. Moreover, we use another measurements, variance percentage, namely $95\%(848d)$, $90\%(467d)$, $85\%(287d)$, $80\%(189d)$, to find a proper percentage and dimension.

As is mentioned before, the the highest dimension we can attain using LDA is $n-1$, n is the number of the classes. In this dataset, $n = 50$. Therefore, we reduce the dimension to 32 and 16 for the horizontal comparison, and try 49, 40, 35, 30, 25 dimensions for proper choice in the method itself.

We measure the accuracy of each method, and the result is shown both in figure 6.

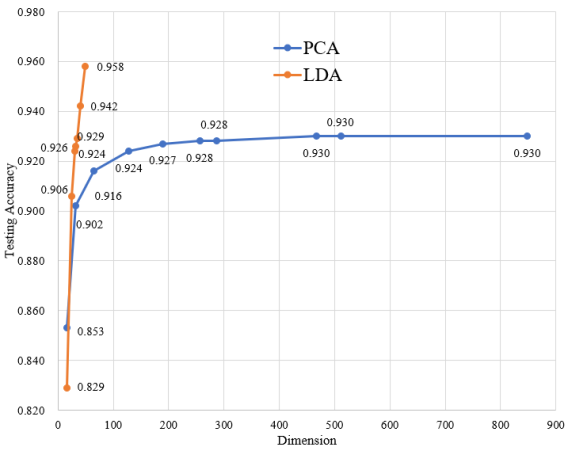

Figure 6. Experimental Results in Feature Projection

PCA From the result, we can find that the final training score rising sharply with the dimension increasing when the dimension is less than 64. And it rising slowly until the dimension reach 256. Finally, the score stays at 0.930 and stop, and 0.930 is the highest score we can get by PCA. We can find that the principle components are constrained into 200 dimensions. And other features may contribute very little to the result.

LDA From the figure, we can find that the score rising sharply with the dimension reduced by LDA increasing, and the highest score is 0.958 when the dimension is the 49, the highest dimension we can attain using LDA. Obviously, the performance is much better than that using PCA with same dimension, even better than the best performance of PCA. As for the reason, LDA is a supervised learning method, and during the process of dimension reduction, it has already used the label information. In comparison, PCA is unsupervised learning method.

2.6. Conclusion in Feature Projection

First, we will summary the difference between PCA and LDA.

1. *The core idea.* PCA mainly finds the better projection method from the covariance angle of the feature, that is, selects the direction in which the sample point projection has the largest variance. However, LDA considers the classification label information more, and

seeks to increase the distance between data points between different categories after projection and minimize the distance between the same category of data points, that is, the direction with the best classification performance.

2. *The learning method.* PCA is unsupervised learning, so most scenarios are only part of the data processing process and need to be combined with other algorithms. LDA is a supervised learning method. In addition to reducing the dimension, LDA can also be used for predictive applications. Therefore, it can be used in combination with other models or independently.

3. *The number of available dimensions after dimension reduction.* After LDA is reduced, a maximum of $n-1$ dimensional subspace (n is the number of classification labels) can be generated. Therefore, LDA is independent of the number of original dimensions, and only the number of data labels is related; while PCA has n dimensions available at most, that is, the maximum can be selected.

LDA is a classic and popular algorithm in the field of machine learning and data mining, however, the algorithm itself still has some limitations:

1. When the number of samples is much smaller than the feature dimension of the sample, the distance between the sample and the sample becomes larger, making the distance metric invalid, namely LDA will be invalid.

2. LDA is not suitable for dimensionality reduction of non-Gaussian samples.

3. LDA may overfit data.

In conclusion, each method has its own pros and cons, so we should choose proper method in different situations in order to get a better performance.

2.7. Feature Learning

We compare AE, VAE, and β-VAEs in this way: for each method, networks with 16, 32, 64, 128, 256-dimension latent representation are first trained on the whole dataset respectively to obtain models. Second, the encoder of those models map original data into 16, 32, 64, 128, 256 dimensions. Then the compressed data are split into train data and test data, where SVM are trained on train data and tested on test data. We use the accuracy on test dataset as the metric to measure the performance of the dimensionality reduction techniques. Figure 7 provides test accuracy curves about compressed dimension of AE, VAE, β-VAEs. There are several experimental observations based on the results above:

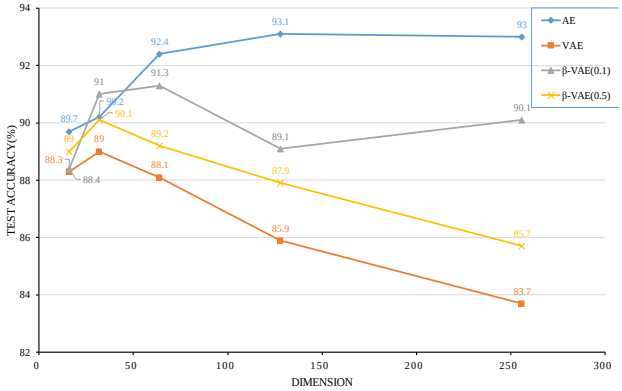

Figure 7. Test accuracy curves about compressed dimension of AE, VAE, β-VAEs

1. The performance of AE goes up as the compressed dimension grows from 0 to 128. However, expanding dimensions over 128 does not promote the performance but causes a slightly decline. Thus 128 is the optimal dimensionality for AE reduction method that maintain the most information of original data with the least dimensionality.

2. As for VAE($\beta = 1$) and β-VAEs, the performance goes down as the dimension grows. VAE and $\beta = 0.5$-VAE reach their top on 32 dimension. $\beta = 0.1$-VAE reach a top on 64 dimension.

3. We find that the reconstruction loss of the model has significant impact on the final performance. Basically ,the lower the loss is , the higher the performance will be.

4. KL loss used in VAE and β-VAEs applies a stronger constraint on the latent bottleneck and limits the representation capacity of z. For some conditionally independent generative factors, keeping them disentangled is the most efficient representation. Therefore a higher β encourages more efficient latent encoding and further encourages the disentanglement. Meanwhile, a higher β may create a trade-off between reconstruction quality and the extent of disentanglement. Thus $\beta = 0.1$-VAE that has higher efficiency, outperforms AE while its reconstruction loss is larger than AE.

References

[1] Feature selection. https://en.wikipedia.org/wiki/Feature_selection.

[2] C. Albon. Feature selection using random forest. https://chrisalbon.com/machine_learning/trees_and_forests/feature_selection_using_random_forest/.

[3] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

[4] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[5] I. Kononenko, M. Robnik-Sikonja, and U. Pompe. Relieff for estimation and discretization of attributes in classification, regression, and ilp problems. 1996.

[6] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE T-PAMI*, 2013.

[7] D. N. Osherson, J. Stern, O. Wilkie, M. Stob, and E. E. Smith. Default probability. *Cognitive Science*, 1991.

[8] M. Robnik-ikonja and I. Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Machine Learning*, 53:23–69, 2003.