CS245 Project 1: Dimensionality Reduction

517021910915 Yu Wu
517021910692 Xinyue Wang
517021910718 Yu Ouyang

Last update: April 24, 2020

1 Introduction

In this paper, we evaluated and compared several feature reduction methods. The data set we use is Animals with Attributes (AwA2) dataset from https://cvml.ist.ac.at/AwA2/. This dataset consists of 37322 images of 50 animal classes with pre-extracted deep learning features for each image. These features has a dimension of 2048. We firstly trained a linear SVM for image classification using the original 2048-d features and set the result as a baseline. Then we evaluated several feature selection, feature projection and feature learning methods and perform image classification again based on the obtained low-dimensional features. In each section, we split the images in each category into 60% for training and 40% for testing.

2 SVM using deep learning features

In this section we use the original deep learning features to train the linear SVM for image classification. To get the best performance, we tried different parameter C and get the test accuracy showed in **Table** 1.

Table 1: Parameters grid of linear SVM

C	0.00001	0.0001	0.001	0.01	0.1
Accuracy	0.8790	0.9175	**0.9262**	0.9187	0.9119

According to the result showed in table.**Table** 1, we choose C = 0.001 and get the accuracy of the baseline method 0.9262.

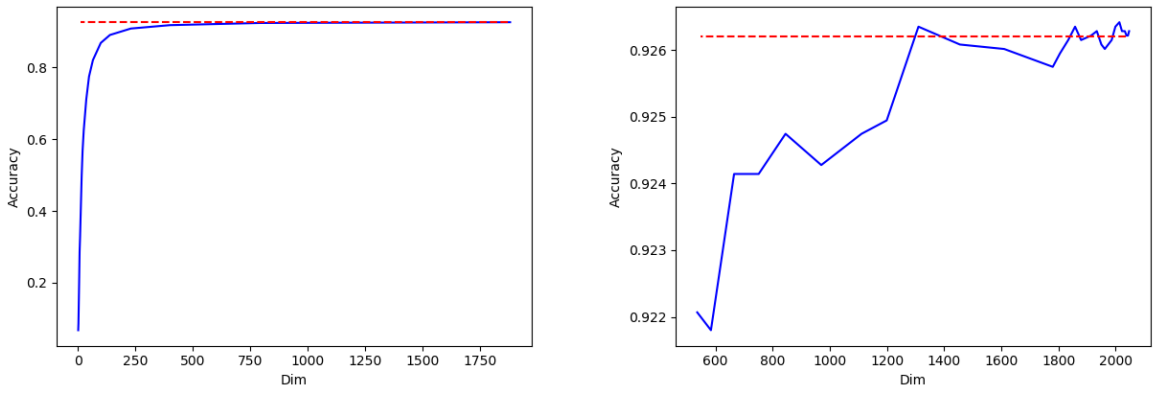
By the way,while realizing the algorithms,we find that different ways to divide the data set into training set and testing set play an important role in the result,leading to different values of benchmark with C=0.001.

3 Feature selection

Feature selection means to select a subset of the original features, and it can be treated as a simple form of feature projection. In this section we evaluated two feature selection methods, which are Variance Threshold and Genetic algorithm.

3.1 Variance Threshold

Variance Threshold reduces the number of feature dimensions by setting a threshold and cutting off the low-variance features whose variance doesn't meet the threshold. Analyze the original features in the training set, the maximum variance among the features is 6.6796, the minimum is 0.0245, the mean is 0.4952, and the median is 0.3029. So we set the threshold from 0.1 to 5.0 with step 0.3 to get the general result. And to find the best performance, we set the threshold from 0.04 to 0.12 with step 0.005, from 0.12 to 0.27 with step 0.03, and from 0.27 to 0.61 with step 0.05. The test result is shown in **Figure** 1.



(a) **(b)**

Figure 1: Performance of Variance Threshold

The red line in **Figure** 1 is the baseline performance. The result shows that With the reduction of feature dimension, the test accuracy fluctuates and goes down in general. **Table** 2 lists the cases that performs better than the baseline result in our experiment.

Table 2: Improved performance of Variance Threshold

Dimension	2047	2032	2022	2013	1999	1934	1858	1310
Accuracy	0.9263	0.9263	0.9263	**0.9264**	0.9263	0.9263	0.9263	0.9263

From **Table** 2 we can see that feature reduction by Variance Threshold achieves a tiny improvement and gets a best test accuracy of 0.9264 with dimension 2013 in our experiment. However, dimension 1310 is the

best choice that uses least features and meanwhile performs well enough with test accuracy 0.9263. And to get an accuracy more than 0.9, dimension 230 is enough with accuracy 0.9086.

3.2 Genetic algorithm

3.2.1 Theory of genetic algorithm

Genetic algorithm is a computational model that simulates the biological selection process of Darwin's biological evolution theory and genetic mechanism, and is a method to search for the optimal solution by simulating the natural evolution process.

The concept of genetic algorithm nouns are as follows:

- `Chromosome` In the parameter optimization problem, a chromosome represents a parameter. In our experiment it represents a particular choice of the features.
- `Genes` Each chromosome corresponds to multiple genes. Genes are binary numbers, and the value length of genes is the length of chromosomes. In our experiment the length is the original number of the features and each binary number indicates whether we choose the feature of not.
- `Population` The range of initial chromosome values.
- `Fitness function` Function that evaluate how a chromosome adapts to nature. In our experiment we computes the SVM-clustering validate accuracy.

In order to find the optimal chromosome, it is necessary to continuously operate the chromosome with excellent fitness function value. The specific operations are as follows:

- `Selection` Chromosome selection is to select the chromosomes with good fitness function values and keep the population number unchanged, that is, to save the chromosomes with good fitness function values to the next generation population with higher probability.
- `Crossover` The crossover operation is to generate new chromosomes and exchange some genes of different chromosomes. Specific methods include single-point crossover and multi-point crossover, and we use only single-point crossover in our experiment.
- `Mutation` The mutation operation is also to generate new chromosomes by changing some genes of some chromosomes.

The computational procedure of genetic algorithm is as follows:

1. Initialize populations, chromosomes and genes.
2. Calculate the fitness function value of each chromosome in the population.
3. Select, cross, and mutate the chromosomes in the population to generate a new population.
4. Determine whether the termination conditions are met? If not satisfied, jump to step two; if satisfied, output the result.

3.2.2 Experiment and analysis

The parameters used in our experiment are listed in **Table** 3.

Table 3: Parameters in genetic algorithm

Parameter	Definition
pop	the size of the population
chro	the length of the chromosome, fixed as 2048 here
iter	the iterations we compute
pc	the probability of crossover
pm	the probability of mutation
pi	the probability of selecting a feature in the initializing step

We tried different sets of parameters. Some of the training process is shown in **Table** 4 and the test result is shown in **Figure** 2.

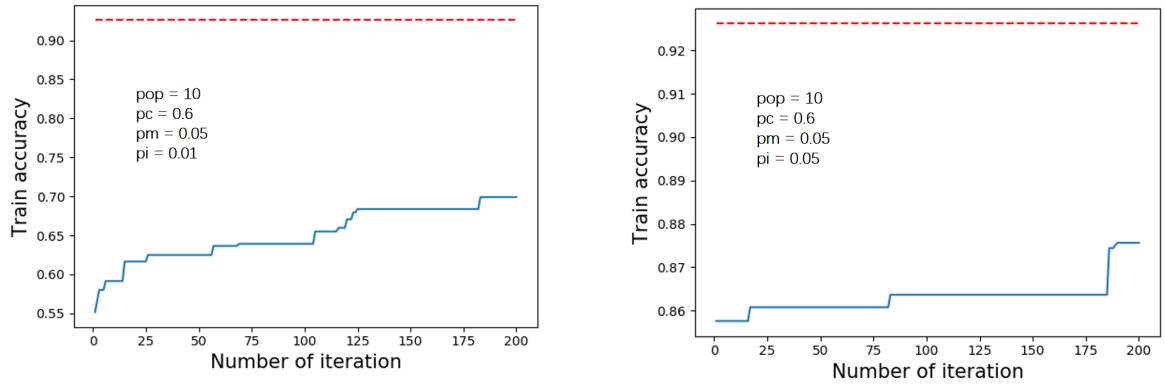


Figure 2: Training process of genetic algorithm

Table 4: Performance of genetic algorithm

pop	iter	pc	pm	pi	accuracy	dimension
5	100	0.6	0.05	0.5	0.9201	1026
10	100	0.6	0.05	0.5	**0.9208**	1027
10	200	0.6	0.05	0.01	0.6858	49
10	200	0.6	0.05	0.05	0.8603	135
20	300	0.6	0.05	0.05	0.8722	142

Figure 2 indicates that a large iteration number is quite important to find a better selection. The largest iteration number in our test is 300, and if the nunber grows larger, the result may be better but meanwhile more time consuming. With limited iterations, the initialization of the chromosomes becomes the key point to get a better result. According to our experiment, a larger pi leads to a better result, which means to choose more features at first. In addition, random initializing leads to more uncertainty with limited iterations, and

the final accuracy with same parameters defers up to 0.073 in our experiment.

4 Feature projection

4.1 Introduction

This part realizes feature projection.We split the original dataset into train set and test set according to the ratio 6:4. The train dataset shape after splitting is (22393, 2048), and the original feature dimension is 2048. The test dataset shape is (14929, 2048). We use sk-learn library to realize PCA and LDA. After training PCA and LDA using the data (and label) in train set, we used SVC to evaluate the performance of the 2 models by classifying the decomposed features of test set and get the score and f1_score of classifying in order to test how well they can maintain the original information contained in the high-dimensional features.

4.2 LDA

4.2.1 Theory of LDA

LDA is Linear Discriminant Analysis. LDA is similar to PCA(Principle Component Analysis) since both of them are feature projection methods and try to project the original high-dimensional features into a new low-dimensional space.

There are some differences between them. PCA is unsupervised model, while LDA is supervised. LDA takes the label of the features into training too. PCA tries to find the component axis that maximize the variance. LDA tries to find component axis which can maximize the class separation and try to make the difference within a class smaller.We chose LDA instead of PCA because LDA has the attribute of maximizing the distance between classes, and the dataset has 50 classes.

In PCA, the dimension of features after projection is unlimited by the class of data. But in LDA, the argument (n_components) should not be bigger than $min(n\_features, n\_classes - 1)$ .The computational procedure of LDA is as follows.

1. Calculate the scatter matrix S_w of each class.
2. Calculate the scatter matrix S_b between different class.
3. Calculate $S_w^{-1} S_b$ and compute the eigenvectors and eigenvalues of it just like PCA
4. Rearrange the eigenvectors and eigenvalues in a way that the columns of the eigenvector matrix V and eigenvalue matrix D are sorted in order of decreasing eigenvalue. The eigenvalues represent the distribution of the source data's energy among each of the eigenvectors, where the eigenvectors form a basis for the data.
5. Select the first k columns of V as the $p \times k$ matrix W,
6. Project the original data onto the new basis: $t_i = W^T x_i$

Table 5: Performance with different a in LDA

a-num	2	4	8	12	16	20	24	32	40	48	49
score	0.197	0.328	0.497	0.630	0.698	0.745	0.814	0.861	0.900	0.922	0.922
F1-score	0.085	0.185	0.328	0.453	0.531	0.586	0.668	0.745	0.818	0.884	0.893

4.2.2 Experiment result and analysis

We use sk-learn library to realize LDA. We tried different components number a from 2 to 49 since a can't be bigger than $class - 1$. After training the LDA model, we get the decomposed features of test set and input them into the SVC model to check the performance of these features in order to test how well they can maintain the original information contained in the high-dimensional features. Score and f1_score are two ways of evaluating the performance of decomposed features in SVC. We found that when a is close to the number of original class number (which is 50), the result will be better. However, when a becomes bigger than 49, the result has no improvement. We thought it's just because we are creating new space based on our feature classes and when a is bigger than 49, there is no space to further improve the class separation. When a is very small, the performance of SVC classification will be quite bad. It is reasonable because it lost a lot of information in these conditions.

Similar to PCA, since LDA discarded some information compared with the original feature, so the final result can't be totally correct. The relationship between the new dimension a and the score is shown in **Figure** 3 and **Table** 5.

4.3 PCA

4.3.1 Theory of PCA

PCA is Principle Component Analysis. PCA is a feature projection method that projects the original high-dimensional features into a new low-dimensional space. The inference of PCA is shown in the **Figure** 4 .

4.3.2 Experiment result and analysis

We use sk-learn to realize PCA. First we tried different components number a from 2 to 49 in order to compare with the performance with LDA. Second, we tried to get the dimension of new features by giving the ratio of selected components in the original features.

4.3.3 PCA vs LDA

We tried different components number a from 2 to 49, which is the number of new features after dimension reduction. Score and f1_score are two ways of evaluating the performance of decomposed features in SVC.

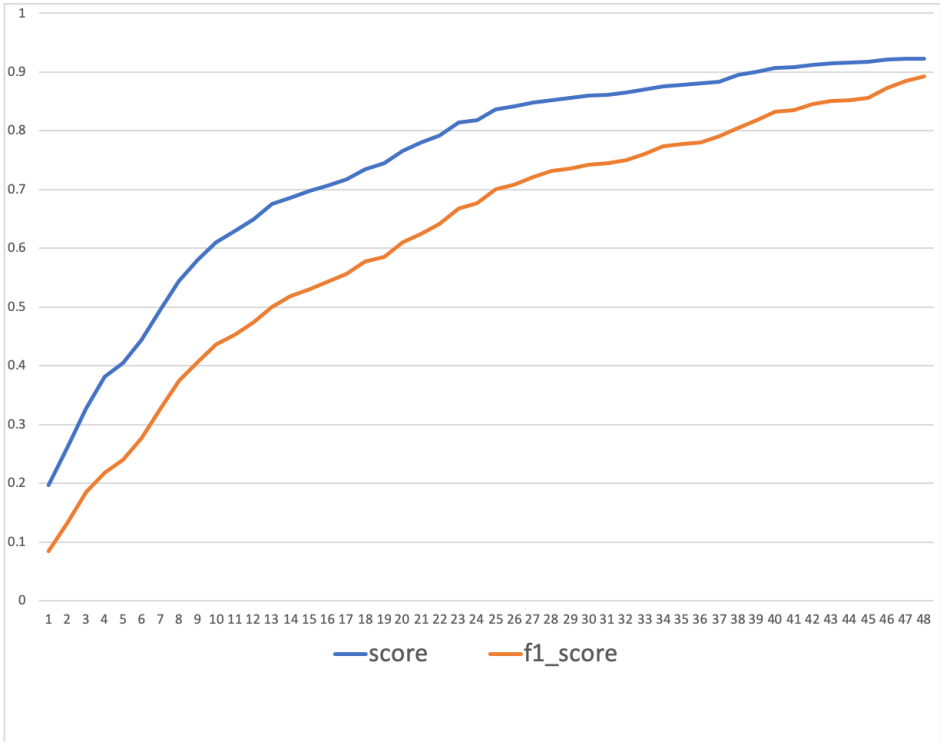
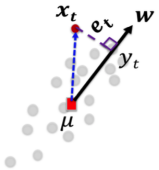


Figure 3: LDA performance with different a

PCA by minimizing MSE

$$J(w) = \frac{1}{N}\sum_{t=1}^{N} ||\boldsymbol{x_t} - (\boldsymbol{x_t^T w})\boldsymbol{w}||^2$$



$$\boldsymbol{x_t^T x_t} - (\boldsymbol{x_t^T w})\boldsymbol{w^T x_t} - \boldsymbol{x_t^T}(\boldsymbol{x_t^T w})\boldsymbol{w} + (\boldsymbol{x_t^T w})\boldsymbol{w^T}(\boldsymbol{x_t^T w})\boldsymbol{w}$$
$$= \boldsymbol{x_t^T x_t} - \boldsymbol{w^T}(\boldsymbol{x_t x_t^T})\boldsymbol{w}$$

Introduce a Lagrange multiplier $\boldsymbol{\lambda}$

$$L(\{\boldsymbol{x_t}\}, \boldsymbol{w}) = J(\{\boldsymbol{x_t}\}, \boldsymbol{w}) - \lambda \cdot (\boldsymbol{w^T w} - 1)$$

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} - \lambda \cdot \frac{\partial (\boldsymbol{w^T w} - 1)}{\partial \boldsymbol{w}} = -2(\Sigma_x \boldsymbol{w}) - \lambda \cdot 2\boldsymbol{w} = \boldsymbol{0}$$

$$||w|| = 1$$

$$\Sigma_x = \frac{1}{N}\sum_{t=1}^{N} \boldsymbol{x_t x_t^T}$$

$$\boxed{\Sigma_x \boldsymbol{w} = (-\lambda) \cdot \boldsymbol{w}}$$

Eigenvalues and Eigenvectors

4

Figure 4: PCA principle

The result is shown in **Figure** 5. We found that when a is bigger the result will be better, which is obvious because it will lose less information.

What's more, when a increases from 2 to 18, the score and f1_score increases faster than after. We suppose it's because the first 18 components take up bigger ratio of the original feature information as is shown in **Figure** 6.

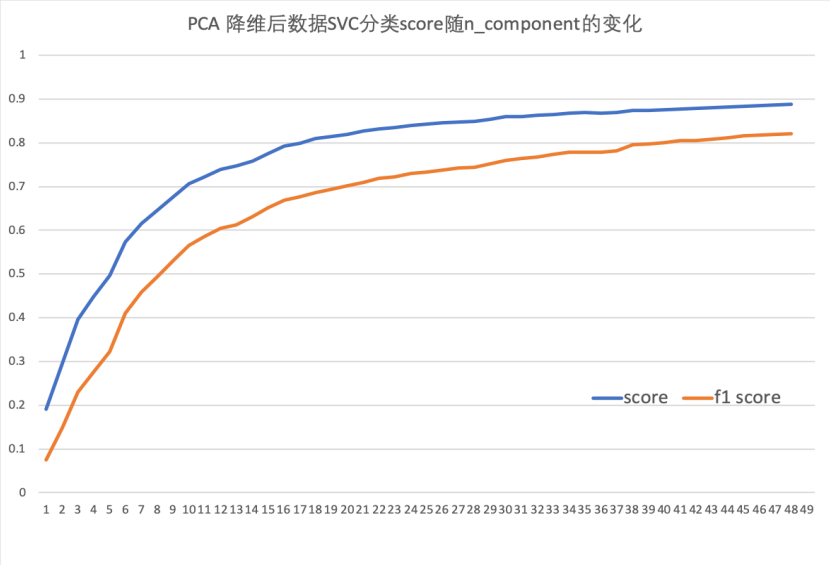


Figure 5: pca perfomance with score and f1score

```
50
[0.05884618 0.048021   0.03694471 0.03008217 0.02698066 0.02368877
 0.02233231 0.02066547 0.01927324 0.01706976 0.01629754 0.01430937
 0.01401745 0.01243661 0.01181911 0.0115305  0.0109071  0.01032496
 0.00981233 0.00911379 0.00905633 0.00853362 0.00826599 0.00802233
 0.00769506 0.00726179 0.00704541 0.00688011 0.00639552 0.00629496
 0.00620496 0.00594396 0.00582322 0.00543705 0.00533173 0.00524232
 0.0051311  0.00489131 0.00473678 0.00459067 0.00445443 0.00436129
 0.00429262 0.00423814 0.00406341 0.00398415 0.00395264 0.00388737
 0.00371562 0.00366204]
numbers 14929, features 50
```

Figure 6: PCA explained_variance_ratio

And we compared the performance of LDA and PCA as shown in **Figure** 7 and **Figure** 8. We can find that when the dimension of new features is low, PCA performs a bit better than LDA. But when the dimension is close to the number of classes, LDA outperforms PCA.

4.3.4 Assign the component ratio

The second way we used to test the performance of PCA is to assign the ratio of the selected components to get the number of the new components. We assign the ratio to be 0.9 and this is the result shown in **Figure** 9. It means that PCA has to select 461 features to maintain 90% of the original features. And the f1_score is 0.89. However, in LDA when the dimension of new features is 49, the f1_score is better than PCA. We

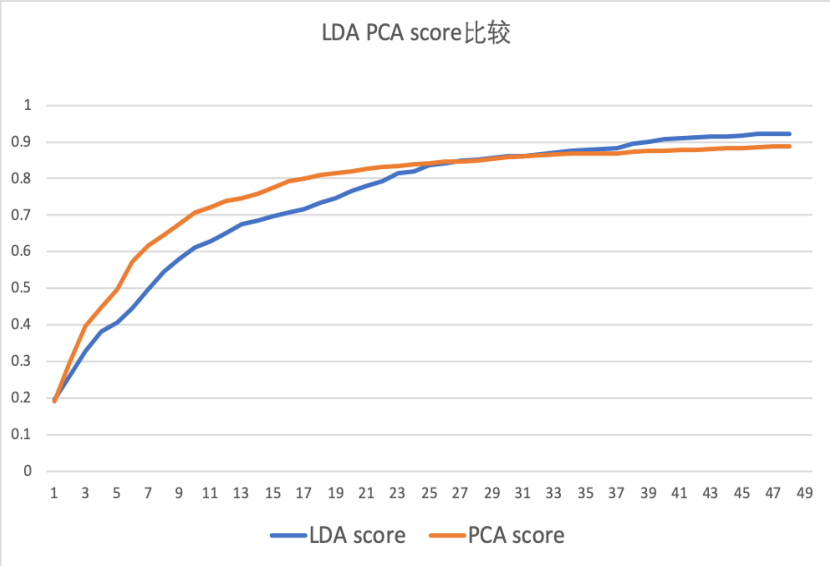


Figure 7: PCA compared with LDA with score

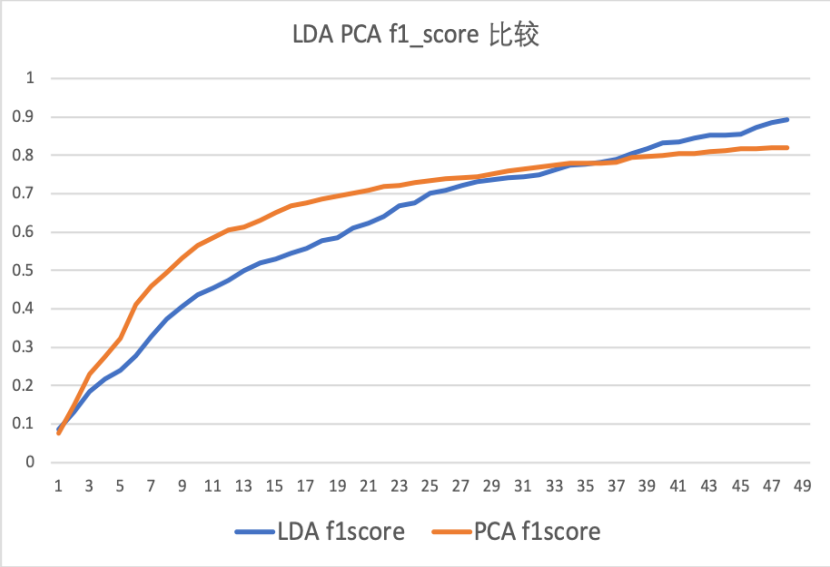


Figure 8: PCA compared with LDA with f1score

suppose it's because when the data has class labels, LDA will perform better in maintaining the information of different classes.

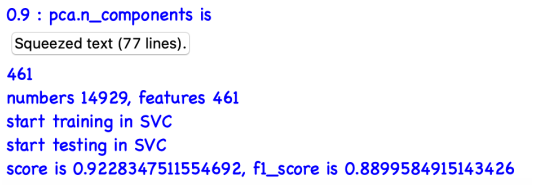


Figure 9: the number of new features

5 Feature learning

In this part,we choose two methods of feature learning to reduce the dimensionality of deep learning features, t-SNE and MDS, and explore the relationship between the number of dimensions and the accuracy of classification.

5.1 t-SNE

5.1.1 Theory of t-SNE

The t-SNE(t-Distributed Stochastic Neighbor embedding) algorithm is a nonlinear dimensionality reduction algorithm which is appropriate to convert high dimensional data to 2 or 3 dimensions,therefore suitable for visualization.

The algorithm applys SNE to the high-dimensional Euclidean distances between data points into conditional probabilities that represent similarities . The similarity of data point x_j to data point x_i is expressed by the conditional probability $p(j|i)$, defined as in the below equation

$$P(j|i) = \frac{(1 + ||x_i - x_j||^2)^{-1}}{\sum_{k \neq i}(1 + ||x_i - x_k||^2)^{-1}}$$

The similarity of data to be obtained is defined as

$$q(j|i) = \frac{(1 + ||\tilde{x_i} - \tilde{x_j}||^2))^{-1}}{\sum_{k \neq i}(1 + ||\tilde{x_i} - \tilde{x_k}||^2)^{-1}}$$

and the cost function is

$$L = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p(j|i) \log \frac{p(j|i)}{q(j|i)}$$

5.1.2 Experiment result and analysis

We use sk-learn library to realize the t-SNE algorithm,and the result of original deep learning features is used as benchmark.With the limitation of dimensionality less than 4 by using Quad-tree,we compare the

Table 6: t-SNE performance with different dimensions

dimension	benchmark	1	2	3
Accuracy	0.9262	0.8871	**0.9010**	0.8977

result in dimension 1,2,3 with the original data.In **Table** 6,it's easy to get the conclusion that t-SNE performs well on very low dimension by achieving a high accuracy,and more or less dimensions may result in worse performance.

5.2 MDS

5.2.1 Theory of MDS

The MDS(Multi Dimensional Scaling) algorithm achieve dimensionality reduction by scaling high dimensional data to lower dimensions and keeping the same distance.

We assume the distance matrix for n samples is $D \in R^{n\times n}$,and d_{ij} is the distance between x_i and x_j. The target is to get the expression of samples in $d^{'}$ dimensional space $X \in R^{d^{'}\times n}$,and $||x_i - x_j|| = d_{ij}$.We define $B = X^T X$,$b_{ij} = {x_i}^T x_j$.By induction,finally we get the formula

$$b_{ij} = -\frac{1}{2}(d_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 - \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2)$$

Therefore,Given distance matrix D, we can calculate B, and then obtain X.

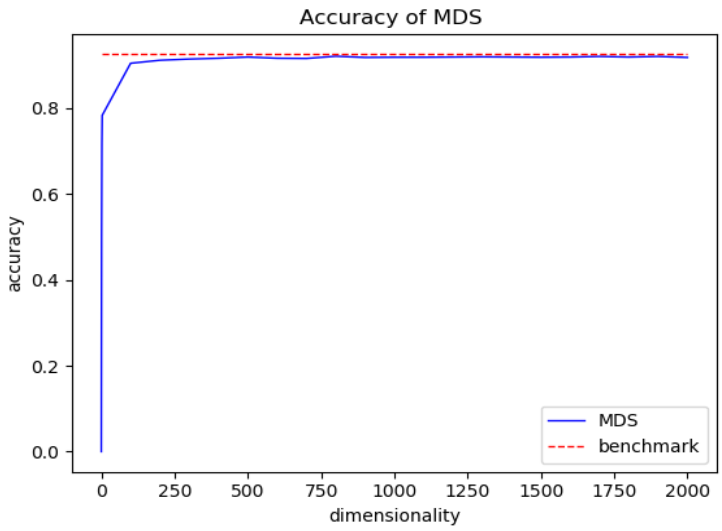


Figure 10: Accuracy of MDS

5.2.2 Experiment result and analysis

We use sk-learn library to realize the MDS algorithm,and the result of original deep learning features is used as benchmark.And we obtain the relationship between dimensionality and accuracy.

In **Table** 10,with increasing of dimensionality,the result becomes better and closer to the benchmark.When the dimensionality is over 100,the ratio of MDS and benchmark is greater than 97% .And when the dimensionality is around 200, the gap can be ignored.

6 Summary

For methods of feature selection, we tried variance threshold and genetic algorithm. We get the best performance of 0.9264 with dimension 2013 in variance threshold. Genetic algorithm is a time consuming method to get a better solution and in our experiment we haven't achieve that. Comparing to feature projection and feature learning, feature selection methods usually gets a much larger dimension of features with tiny improvement of test accuracy.

For methods of feature projection, we tried PCA and LDA. Through comparing the performance and the dimension of the generated features we found that LDA is better when the data has the attribute of different classes. In LDA, when the dimension of the generated features is larger, the SVM score is higher. Because the dataset has 50 classes, so the max dimension of LDA feature is 49. At this time, the SVM with parameter: 0.001 has a score of 0.9223.Since the baseline method in part 2 has an accuracy of 0.9262, it seems that LDA can only roughly maintain the performance of the original feature in SVM but can't make it better.

For methods of feature learning,we try t-SNE and MDS.On the one hand,the performance of MDS is better and approach the benchmark level with the number of dimensions over 200 and its best accuracy is 0.9211.On the other hand,t-SNE has its highest score 0.9010 with data of only 2 dimensions.Therefore,t-SNE algorithm can simplify the dimensions greatly and is suitable for visualization.The shortcoming of both algorithm is time-consuming with high complexity,especially for large amount of data.