Project 3 of CS245: Feature Encoding for Image Classification

515030910369, Sicheng Zuo, Ginga479726995@sjtu.edu.cn
516030910044, Yujie Yang, yangyujie@sjtu.edu.cn
516030910021, Hongxiang Yu, sjtu.yuhongxiang@sjtu.edu.cn
5130309210, Zheng Gong, 573965625@qq.com

May 5, 2019

1 Introduction

In this project, we tried to extract local descriptors from images and used different feature encoding methods to change descriptors to feature vector. The dataset we used is Animals with Attributes (AwA2), which is composed of 37322 images with 50 animals classes. There are three parts in this project. Firstly, we used SIFT algorithm and selective search to extract descriptors and proposals for each image correspondingly. Secondly, we used three different feature encoding methods to convert the descriptors or proposals to feature vectors. Thirdly, the feature vectors were feeded to SVM for image classification. What's more, we also tried to figure out the optimal value of cluster numbers and compare the performance of different methods. The results of experiments shows that the effect of deep learning proposals is much better than the SIFT descriptors. The best result achieved with SIFT descriptors is only $\mathbf{23.05}\%$ and the highest accuracy with deep learning proposals is $\mathbf{89.37}\%$.

2 SIFT descriptors

The scale-invariant feature transform (SIFT) is a feature detection algorithm in computer vision to detect and describe local features in images. Here in this

project, we use SIFT toolbox of opencv to obtain local descriptors. However, only Top 50 SIFT descriptors will be extracted. The reasons are that:

1. Avoid a large number of descriptors. Since there are 37322 images, if the number of descriptors are too large, it will cost plenty of time to train the model.

2. Restricting the number of features points to be extracted can reduce the noise of background efficiently. For example, if the number of feature points is not limited, the extracted SIFT features will be:

Figure 1: SIFT features unlimited

As we can see from the figure-1, there are many noise points belongs to the background. We hope that there are as many SIFT feature points locate on the animal as possible. Limiting the number of feature points may help us reduce the noise points of background. In the figure-2, we can see that most of the feature points are located on the antelope.

Figure 2: SIFT features limited

During this experiments, we splited the the data in each category into 60% for training and 40% for testing. After extracting SIFT descriptors, there are totally 1121346 descriptors in the training set, each descriptors with a length of 128. To accelerate the process of extracting descriptors, we modified our code to do this job on 80 cpu cores in parallel. It takes less that 2 minutes to extract descriptors of 22373 images.

3 Feature encoding methods and SVM

After extracting SIFT descriptors, we tried three methods of feature encoding (Bag-of-Visual-Words, VLAD, Fisher Vector) to encode the descriptors of each image into a feature vector with a fixed size.

3.1 BoVM

The general idea of Bag-of-Visual-Words (BoVM) is to represent an image as a set of features. Features consists of keypoints and descriptors. Keypoints are the ”stand out” points in an image, so no matter the image is rotated, shrink, or expand, its keypoints will always be the same. And descriptor is the description

of the keypoint. We use the keypoints and descriptors to construct vocabularies and represent each image as a frequency histogram of features that are in the image. From the frequency histogram, later, we can find another similar images or predict the category of the image.

The key part of building the BoVM model is the Kmean algorithm. We tried different cluster numbers: 8, 16, 32, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 1000, 2000, 3000. In the BoVM model, the length of encoded feature vectors is the same as cluster numbers. For different cluster numbers, we take different strategies.

- If the cluster number is not big(8, 16, 32, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500), the training data is feeded to the SVM directly. *GridSearchCV* and 5-fold cross validation were used to find the best parameter of SVM.
- If the cluster is big(1000, 2000, 3000), the feature reduction will be applied to the training data before feeding to the model.

3.1.1 Result of small cluster numbers

The results of GridSearchCV with small cluster numbers is shown in table-1.

Table 1: BoVM Results of SVM GridSearchCV

cluster number	linear acc	linear opt params	rbf acc	rbf opt params
8	4.72%	C=10	4.40%	C=1e-5, gamma=1e-4
16	4.71%	C=10	4.40%	C=1e-5, gamma=1e-4
32	4.40%	C=1e-05	4.40%	C=1e-5, gamma=1e-4
50	15.23%	C=0.1	16.26%	C=10, gamma=0.001
100	16.54%	C=0.01	17.25%	C=1, gamma=0.01
150	16.90%	C=0.01	17.55%	C=1, gamma=0.01
200	17.53%	C=0.01	17.98%	C=1, gamma=0.01
250	17.33%	C=0.01	17.84%	C=1, gamma=0.01
300	17.41%	C=0.01	18.04%	C=1, gamma=0.01
350	17.45%	C=0.01	18.18%	C=1, gamma=0.01
400	18.00%	C=0.01	18.45%	C=1, gamma=0.01
450	18.29%	C=0.01	18.74%	C=1, gamma=0.01
500	18.26%	C=0.01	18.73%	C=1, gamma=0.01

Through the observation of table-1, we draw the following conclusions:

- The performance of BoVM model is pool(the reasons will be analyzed in the follows).
- A small cluster number may lead to a very pool performance. This is because a small cluster number make the feature vector not so expressive, so it is hard to distinguish different animals with a short feature vector.
- Increasing the value of cluster number may improve the performance to some degree. But when the cluster number is large enough, the improvement will be small.
- The performance gap between two kinds of SVM kernels is small.

3.1.2 Results of large cluster numbers

Next, we tried larger cluster number (1000, 2000, 3000). Since the dimension of feature vector is equal to the cluster number, we need to use PCA or LDA to reduce the feature dimension. From the result of project-1, we know that larger value of *n_components* in LDA will reserve more information which may lead to a better performance. Hence, we set $n\_components = 49$ in LDA. For the PCA, we tried the value of *n_components* from 0.9 to 1.0 with a step of 0.01. The parameters of SVM is setted according the result in table-1. The optimal parameter for linear SVM is $C = 0.01$ and the optimal parameter for rbf SVM is $C = 1, gamma = 0.01$. Table-2 shows the result of LDA and figure-3, figure-4 show the result of PCA.

Table 2: BoVM Results of SVM + LDA

kernel \ Cluster	1000	2000	3000
linear	16.45%	16.87%	16.32%
rbf	16.58%	16.93%	16.29%

After applying LDA feature reduction, the performance even became worse. The reason may be that LDA lost a lot of effective information in the process of feature reduction, and the 49-dimensional feature vector has insufficient expressive power. As a result, LDA may not be a good method in this problem.

However, LDA can converts very high-dimensional features into 49-dimensional features, so it can accelerate the training of SVM.

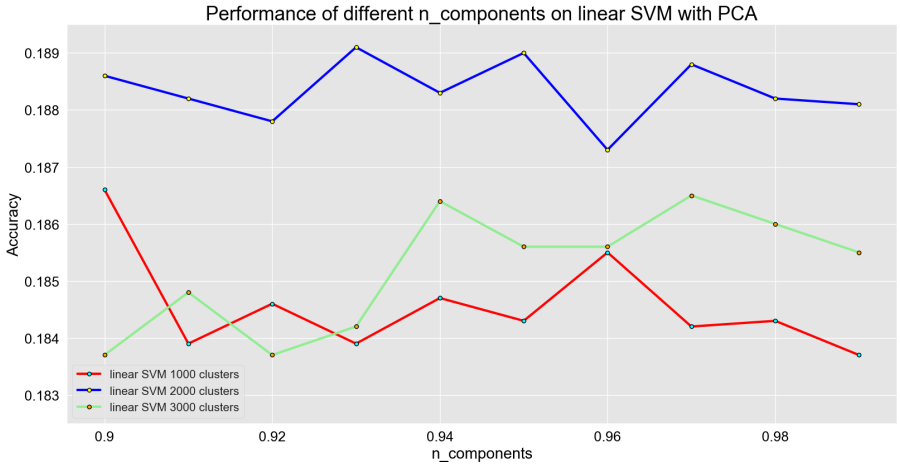

Figure 3: BoVM of PCA on linear SVM

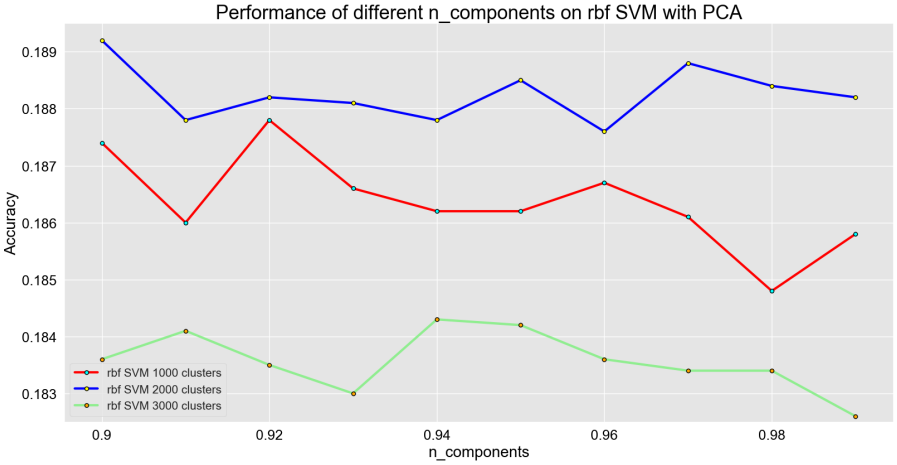

Figure 4: BoVM of PCA on rbf SVM

From the results of figure-3 and figure-4, we are disappointed to find that increasing the cluster number has a limited improvement on the accuracy. And if the cluster number is large enough(more than 2000), the accuracy may even drop.

3.1.3 Analysis and Summary

The different cluster number and performance variance is shown in figure-5. Increasing cluster number does imrove the performance of the model, but when cluster number is big enough, continuing to increase cluster number will actually hurt the model's performance. The best result of BoVM we achieved is $\mathbf{18.92}\%$ with $Cluster = 2000$ and rbf kernel.

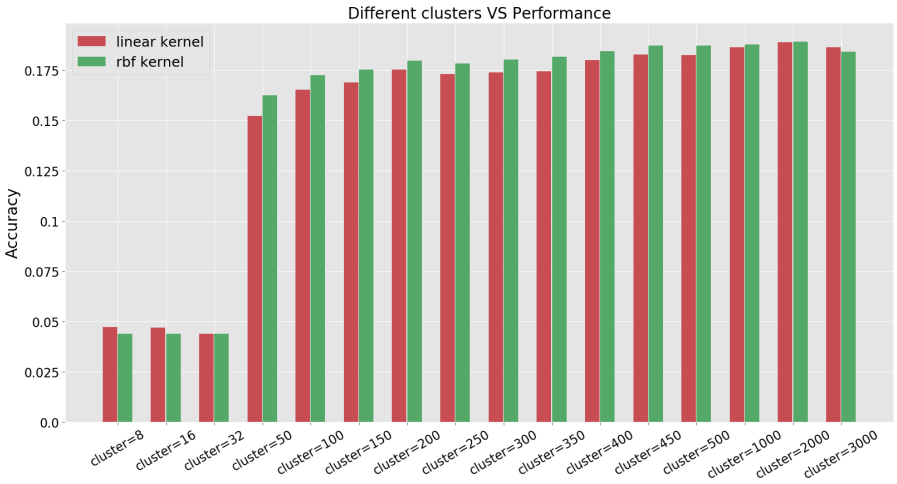

Figure 5: different cluster number and BoVM performance variance

The performance of BoVM model is bad. I think there are three reasons lead to the poor performance:

1. SIFT descriptors has extract large number of features which are irrelevant with the animals, these irrelevant features may do harm to the performance of Kmeans algorithm. For instance, in figure-6, we can see that only few SIFT descriptors is relevant to the animals. As a result, there are many noise points in the train data. That's the most important reason that result in the poor performance.

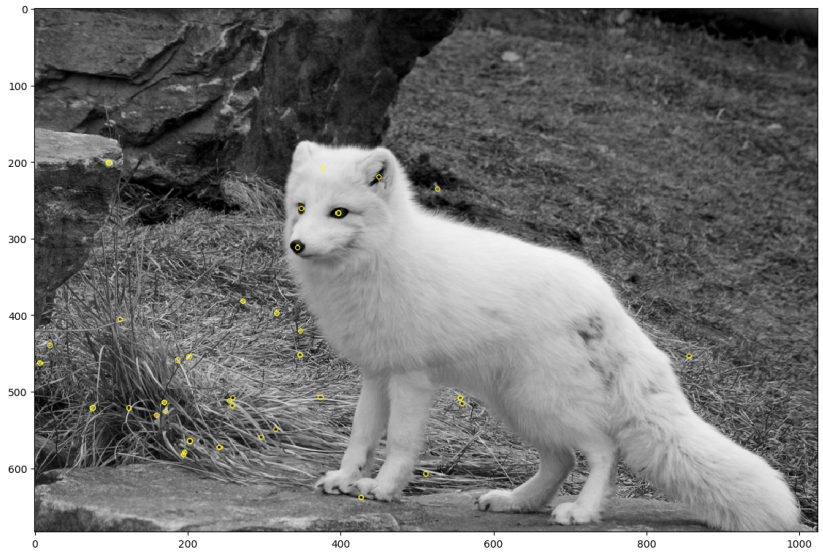

Figure 6: SIFT noise points

2. The extracted SIFT descriptors may be not sufficiently distinguishable. For example, in figure-6, we can see that there are feature points located on the fox's eyes and nose, but most of the animals have eyes and nose. This means that this descriptor is not so useful in distinguishing different animals.

3. As we know that, the initial state has a great influence of the Kmeans algorithm's performance. In our code, we fixed the random seed to 245, this may not be a good random seed.

The future work may includes:

* Design new algorithm to pick the most important and most useful SIFT descriptors and remove the noise points.

* Run the kmeans algorithm multiple times and find a suitable random seed.

3.2 VLAD

Vector of Locally Aggregated Descriptors (VLAD) calculates the descriptors' residual sum of each cluster center. It uses the same descriptors and Kmeans

algorithm with the BoVM. The dimension of VLAD feature vector is $K * D$, where K is the cluster number and D is the dimension of each descriptor. The dimension of each descriptor is 128. Since the feature dimension is too high, the feature reduction methods are needed. Because it is time consuming to run a VLAD experiment, we can't iterate through all the parameter combinations. Therefore, our process of tuning parameters is divided into the following steps:

1. Fix the cluster number and *n_components* of PCA, use *GridSearchCV* and 5-fold cross validation to find the best parameters of SVM.
2. Fix the cluster number and use best parameters found in step 1 to find the best value of *n_components*.
3. After step 1 and step 2, the best parameters of SVM and *n_components* of PCA are determined. And then we can try different value of cluster to find the optimal one.

3.2.1 Tuning parameters of SVM

In this part, we found the best parameters of both linear SVM and rbf SVM by *GridSearchCV*. Before using *GridSearchCV*, we first fix the cluster number to 50 and set parameters of PCA *n_components* to 0.9. By the way, LDA is also used with *n_components* = 49. In the table-3, we showed our results of step 1.

Table 3: GridSearchCV Results of VLAD step1

results / kernel	LDA opt score	LDA opt acc	LDA opt config	PCA opt score	PCA opt acc	PCA opt config
linear	95.54%	18.55%	C=0.001	22.22%	22.62%	C=1
rbf	95.61%	18.48%	C=0.1 γ=0.01	21.18%	21.99%	C=10 γ=0.1

[1] γ denote to the rbf SVM parameter *gamma*

From table-3, we can see that the accuracy of VLAD is better than the BoVM. In this step, we found the best parameters of SVM which will be used in the next 2 steps.

3.2.2 Tuning parameters of PCA

In this part, we will find the best value of PCA parameter *n_components*. The value of cluster number is also fixed to 50 and the parameters of SVM is setted according to the result in section-3.2.1. In figure-7, we can see the *n_components*'s influence to the performance.

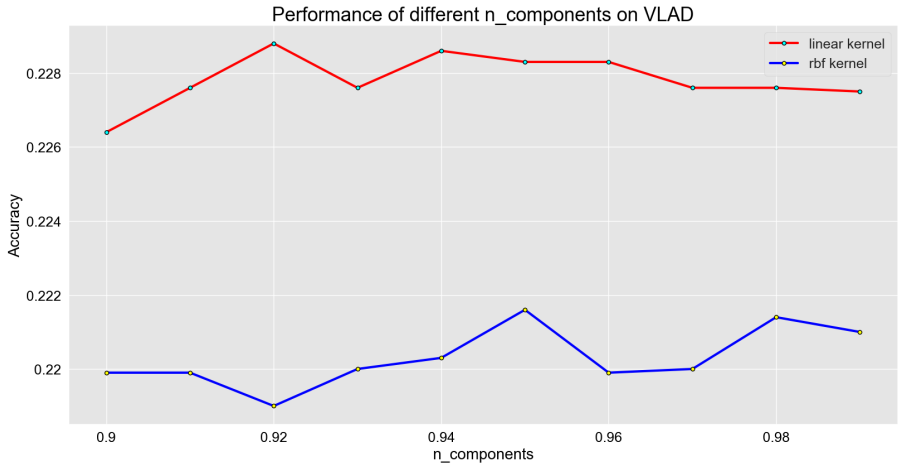

Figure 7: different *n_components* and performance variance

From figure-7, we can get the best parameters of linear SVM and rbf SVM.

- For linear SVM, **n_components** = 0.92
- For rbf SVM, **n_components** = 0.95

3.2.3 Tuning cluster numbers

From the result of BoVM model, we know that cluster number has a big influence to the accuracy. Because running the experiments of VLAD is time consuming, we can not try so many different cluster numbers and large cluster numbers. Therefore, we picked five different cluster numbers to do the tests. The results of our tests are shown in figure-8.

Figure 8: different cluster and VLAD performance variance

The best result we achieved is $\mathbf{23.05}\%$ with $Cluster = 16$ and rbf kernel. There are two points worth noting here:

1. Firstly, the performance of LDA decreases as the cluster number increases. This is because the larger the cluster number, the longer the feature vector, and the more information lost during the dimension reduction to 49 dimensions.

2. Larger cluster number leads to worse performance. The best performance is achieved when $cluster = 16$. Larger cluster numbers lead to longer feature vectors and which is easier to introduce redundant information. This why the performance is worse when increasing value of cluster number.

3.2.4 Analysis and Summary

Apparently, VLAD model's performance is better than the BoVM. But compard with results of project-1 and project-2, the VLAD's performance is still poor. I think the reasons of VLAD's poor performance is the same as BoVM. The extracted descriptors contains too many noise points and the features points are not sufficiently differentiated. The specific reason we have already explained in section-3.1.3, we will not repeat them here.

3.3 Fisher Vector

The Fisher Vector encoding stores the mean and the covariance deviation vectors per component of the Gaussian-Mixture-Model (GMM) and each element of the local feature descriptors together. In this method, the dimension of each feature is $2 * K * D$. Here, the K is the cluster number and D is the dimension of each descriptors. We used vlfeat toolbox of matlab to get the Fisher Vector encoding. Since the underlying of vlfeat uses OpenMP parallelism, it can take advantage of multiple cores for fast extraction. However, the feature dimension $2 * K * D = 256 * D$ is too large, it needs large computation during feature reduction and SVM training, in order to save time and reduce the amount of calculation, we did two things:

1. Choose $cluster = 8, 16, 32, 50, 100$ to do the tests. These value of cluster value are not large. What's more, from the results of VLAD and BoVM, we know that increasing cluster number brings limited improvement to performance or even hurt the performance. Therefore, choose small cluster number is reasonable.

2. Only use linear SVM. From the result of BoVM and VLAD, we found that the performance gap between linear SVM and rbf SVM is not obvious, and sometimes linear SVM works better than rbf SVM. Hence, only linear SVM is enough.

Firstly, fix the cluster number to 50, and set *n_components* to 0.9. Use the *GridSearchCV* to find optimal parameters. The result is shown in table-4.

Table 4: GridSearchCV Results of Fisher Vector step1

results / kernel	LDA opt score	LDA opt acc	LDA opt config	PCA opt score	PCA opt acc	PCA opt config
linear	99.90%	13.67%	C=0.0001	18.83%	19.71%	C=0.001

Secondly, still fix the cluster number to 50 and use the parameter of linear SVM in table-4, change *n_components* from 0.9 to 1.0 with a step 0.01. The result is shown in figure-9.

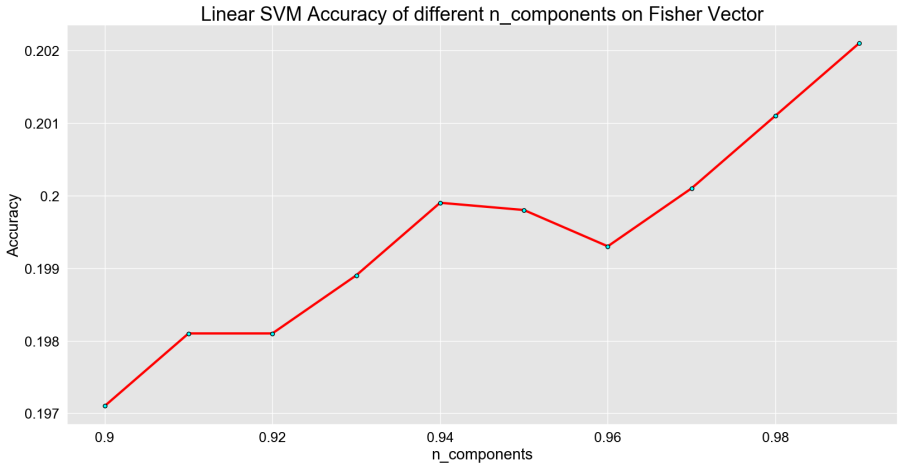

Figure 9: different *n_components* and performance variance

Figure-9 shows that remain more information tends to higher accuracy. Therefore, *n_components* will be setted to 0.99 in the following tests.

Thirdly, after the parameters of SVM and PCA are determined, the cluster number can be tuned. The result is shown in figure-10.

Figure 10: different cluster and Fisher Vector performance variance

After all the tests above, the highest accuracy we achieved with Fisher Vector is **20.46**%. Maybe we can get better performance by increasing the cluster number. Because the figure-10 shows that larger cluster number leads to better

performance of linear SVM + PCA.

4 Deep Learning Proposals

In this section, we tried another way to get the local descriptors. First, we used the selective search to select regions in the image that may have objects. Then we used the Inception v3 model to extract the deep learning features of these regions. Just like SIFT features, there may be many candidate areas in a single picture. If we do not limit the number of candidate areas, there will be large number of descriptors. For example:

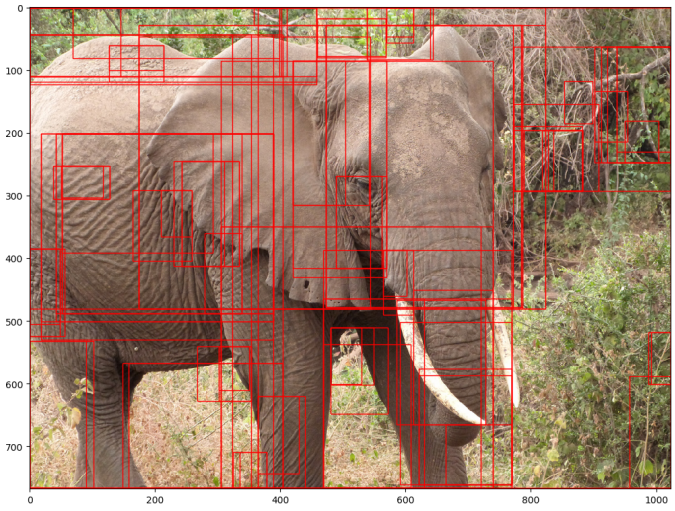

Figure 11: Deep learning proposals unlimited

There are more than 100 candidate areas in this picture. If we use all of them, it will take large amount of time to extract deep learing features and train SVM. What's more, a lot of noise areas will be bringed to the dataset, this is fatal problem for the kmean algorithm. Therefore, we applied some key tricks to reduce the number of candidate areas.

- Delete duplicate areas and included areas.
- Remove too large and too small areas(If the area is too large, then the features are not obvious. If the area is too small, then there is no valuable information).

- Arrange the areas according to the size, take the Top 5(Larger areas also have a higher probability of containing objects).

After applying these tricks, the candidate areas is displayed in figure-12. Not only we reduced the number of candidate areas greatly, but also we removed large number of noise areas. These key tricks really helpful for us to improve the performance of our model.

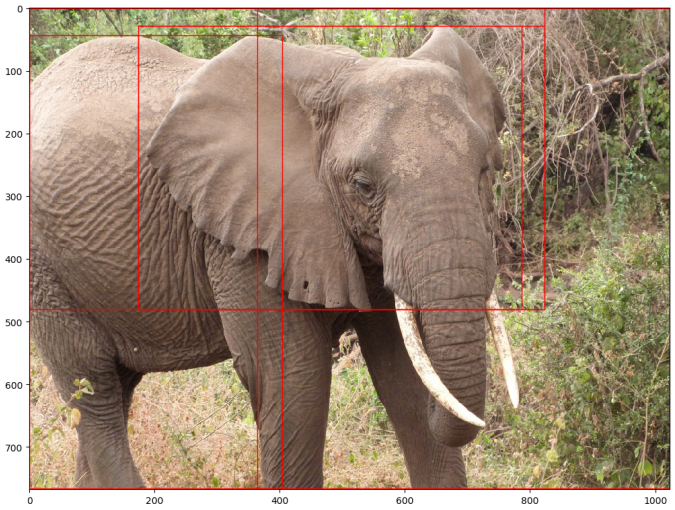

Figure 12: Deep learning proposals limited

4.1 BoVM

Just like we mentioned above, different strategies were applied to different cluster numbers. For the small cluster number, we select 8, 16, 32, 50, 100, 500 to do the tests. The results is shown in table-5.

Table 5: BoVM Results of SVM GridSearchCV

cluster number	linear acc	linear opt params	rbf acc	rbf opt params
8	22.68%	C=10	24.49%	C=1, gamma=0.1
16	42.35%	C=10	43.08%	C=10, gamma=0.1
32	65.99%	C=10	65.81%	C=10, gamma=0.1
50	80.19%	C=1	80.37%	C=10, gamma=0.01
100	83.98%	C=1	83.85%	C=10, gamma=0.01
500	85.47%	C=0.1	85.70%	C=10, gamma=0.01

According to the optimal configurations in table-5, we can get the SVM parameters which will be used in the following tests:

- linear SVM: C=0.1 (larger cluster number with smaller C)
- rbf SVM: C=10, gamma=0.01

Then, we did tests on larger cluster numbers (1000, 2000, 3000).

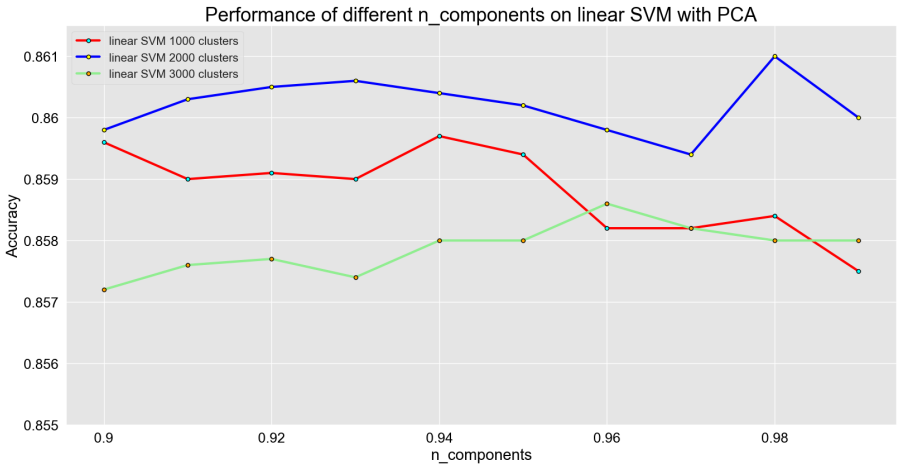

Figure 13: Large cluster number BoVM with deep learning proposals

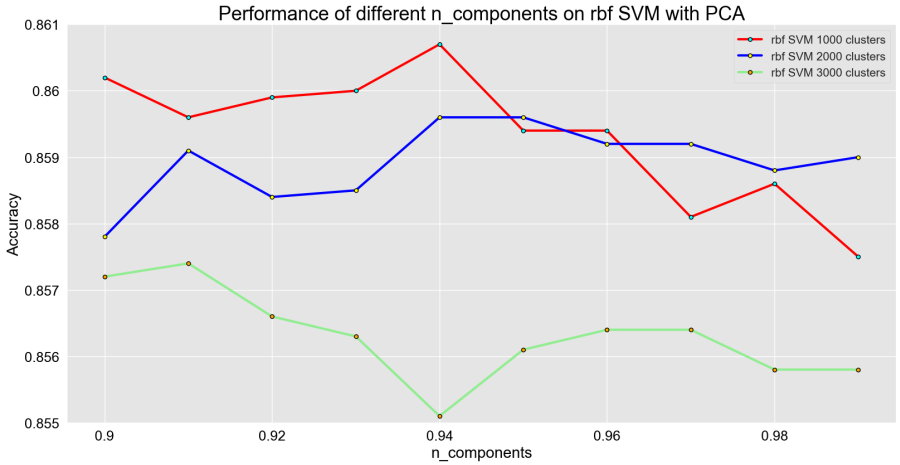

Figure 14: Large cluster number BoVM with deep learning proposals

From the figure-13 and figure-14 we can easily see that different values of *n_components* has little effect on the results and increasing the cluster number has a limited improvement on the performance. These two observations are consistent with our previous conclusions. The summary results of BoVM is displayed in figure-15. The best performance $\mathbf{86.10}\%$ is achieved when $cluster = 2000$ using linear SVM.

Figure 15: Summary results of BoVM with deep learning proposals

4.2 VLAD and Fisher Vector

We know that the dimension of VLAD feature vector is $K * D = 2048 * K$ and the dimension of Fisher Vector is $2 * K * D = 4096 * K$. Here K is the cluster number and D is the dimension of each descriptor. The descriptors are extracted by Inception v3 model, so the dimension of each descriptor in this experiment is 2048. Accroding to the results above, we know that different values of *n_components* has little effect on the results and the performance gap between linear SVM and rbf SVM is small. Due to time and computing power limitations, we took the following methods to complete the experiment:

1. Since the feature dimension of VLAD and Fisher Vector is too high, we only tried three small cluster numbers (8, 16, 32). By this way, we can avoid memory overflows and problem of excessive time overhead.

2. Since the performance gap between rbf kernel and linear kernel is small, and for the same dataset, rbf kernel takes more time than linear kernel.

Therefore, we only used linear kernel.

3. Since the different values of *n_components* has little effect on the results, we did not tuned this parameter. Instead, we set this parameter to 0.9. In this way, we have further reduced the amount of computation.

Here is our results of VLAD and Fisher Vector in table-6.

Table 6: Results of VLAD and Fisher Vector using deep learning proposals

methods \ Cluster	8	16	32
VLAD	89.25%	87.46%	74.30%
Fisher Vector	89.37%	88.66%	78.36%

We got the same result as when we used SIFT features. The performance of Fisher Vector and VLAD is better than BoVM. The best performance of VLAD is **89.25**% and the best performance of Fisher Vector is **89.37**%.

4.3 Analysis and summary

By using deep learning proposals, the performance of the three models BOVM, VLAD, Fisher Vector has been greatly improved. I think deep learning proposals have the following advantages over SIFT descriptors:

1. By filtering the candidate regions, we not only remove a lot of noise points, but also reduce the size of the data and improve the performance of the model. But there are plenty of irrelevant descriptors in SIFT descriptors, which is the fatal problems. That's why the performance of using SIFT descriptors are so bad.

2. The Inception v3 model is powerful. The features extracted by the Inception v3 model are more expressive and more distinguishable than the SIFT features.

But using deep learning proposals need a lot of time to extract image features with Inception v3 model. And the dimension of each descriptor is 2048 larger than the SIFT descriptor 128. This will further increase the complexity of the data, making the problem of time consumption even more serious.

5 Project Summary

In this project, we used SIFT and deep learning to extract the local descriptors and proposals of figures, and use three methods (BoVW, VLAD, Fisher Vector) to convert the descriptors and proposals to feature vectors with fixed length. Finally, use SVM to do the image classification. Table-7 summarizes all our experiments' results.

Table 7: Results Summary

Features	Model	Cluster	accuracy
SIFT	BoVW	2000	18.92%
SIFT	VLAD	16	23.05%
SIFT	Fisher Vector	100	20.46%
Deep Learning	BoVW	2000	86.10%
Deep Learning	VLAD	8	89.25%
Deep Learning	Fisher Vector	8	89.37%

Obviously, for BoVW model, the most suitable cluster number is around 2000, too small or too large cluster number will not get a good performance. However, for the VLAD model and Fisher Vector model, a small cluster tends to higher accuracy. When we use deep learning proposals, from the perspective of accuracy, Fisher Vector > VLAD > BoVM; from the perspective of speed, BoVM > VLAD > Fisher Vector. Because the dimension of BoVM feature vector is equal to the cluster number, the speed of BoVM is much faster than the other two models. By the way, the performance gap between BoVM and the other two models are not so large. Therefore, I think BoVM is the best feature encoding method.

What's more, although using deep learning proposals takes a lot of time, but we get a much better performance than SIFT descriptors. The key reason that deep learning proposals is much better than SIFT descriptors is there are less noise points in deep learning proposals. As a result, if we want to improve the performance of SIFT descriptors, we need to find a new ways to pick important feature descriptors and remove irrelevant descriptors.