CS245 Project 3

Xiang Gu 5130309729

Tao Zhu 515020910243

Weihong Lin 515030910643

Chacha Chen 515021910302

May 1, 2019

1 DataSet

In this project, we still use the dataset AwA2, which consists of 37322 images of 50 animal classes. We split the images in each category into 60% in training and 40% in testing. The sizes of training set and testing set are 22373 and 14949.

2 Main Work

In this project, we use SIFT to extract some keypoints for each image and then compute the local descriptors. Since some images may contain a big background, a better method is using a detection algorithm to generate the proposals for animals and then extract the descriptors on each proposal. However, we don't have GPU resource to do that and we find that some images have more than one animal. So, finally we just use the whole image to get the local descriptors. Figure 1 shows an example of the SIFT keypoints.

After getting the local descriptors, we use three algorithms Bag-of-Word, VLAD and Fisher Vector to encode the descriptors of each image into a feature vector with a fixed size. We try different cluster number of K-means or GMM algorithm to generate features. Finally, we train some SVM classifiers for these features with the same parameters and observe the performance.

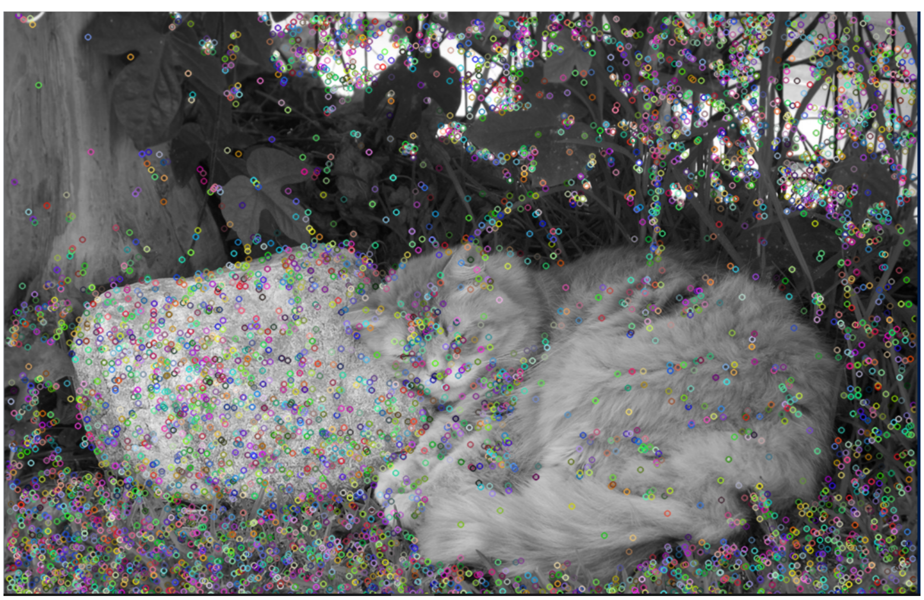

Figure 1: SIFT Keypoints of a image

3 Experiment Setting

3.1 SIFT

We use the SIFT toolbox of Opencv. Since the deep learning features which are used in Project 1 & 2 are extracted by ResNet101 with input size $224 * 224$, we resize each image to $224 * 224$ and run SIFT on its gray view. The dimension of each descriptor is 128.

3.2 Bag-of-Word

Since the descriptors of the whole training set is too large($\sim 10^7$). We subsample 10% images' descriptors of each category($\sim 90W$) and run K-means on these descriptors. We try different cluster numbers: 32, 64, 128, 256, 512. In this algorithm, the dimension of features is the same as the cluster number.

3.3 VLAD

The same descriptors($\sim 90W$) are used for the K-means. We also try different cluster numbers: 8, 16, 32, 64, 128. The dimension of each feature is $K * D = 128K$, here K is the cluster number of K-means and D is the dimension of each descriptor. We use PCA to do dimension reduction and the new dimension is $4K$.

3.4 Fisher Vector

This algorithm has a large computation, so we run three small clusters $(4, 6, 8)$ for GMM to see the performance. We use the EM class of Opencv and delete the cluster whose weight is smaller than $1/K$. In this method, the dimension of each features is $(2 * D + 1) * (K' - 1) = 257(K' - 1)$, here K' is the rest cluster number of GMM and D is the dimension of each descriptor. We also use PCA to get features with smaller dimensions. The new dimension are 32, 64, 128 for the three cluster numbers. What's more, we calculate an additional feature set whose cluster number is 8 and the new dimension is 32. This will help us to compare the performance between VLAD and Fisher Vector.

3.5 SVM

The purpose for this project is to compare different encoding algorithms. So, we only use one setting for the SVM classifiers of different experiments. We use the SVM toolbox from sklearn. The parameters are $kernel = rbf$, $gamma = scale$ and $decision\_function\_shape = ovr$. The other parameters use the default value.

4 Experiment Result

4.1 Different Cluster Numbers

Figure 2 3 4 shows the accuracy on different cluster numbers of each algorithm. We can see the higher cluster number tends to get a higher accuracy. When the cluster number is large enough, the improvement will become small. Though we use PCA to do the dimension reduction, the new dimensions increase with the same ratio of the cluster number. So, we think the dimension factor can be ignored in this part.

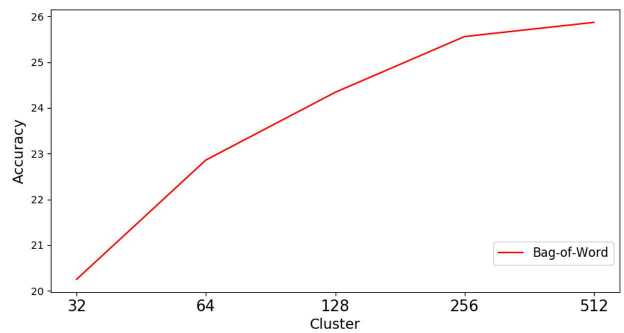

Figure 2: Bag-of-Word

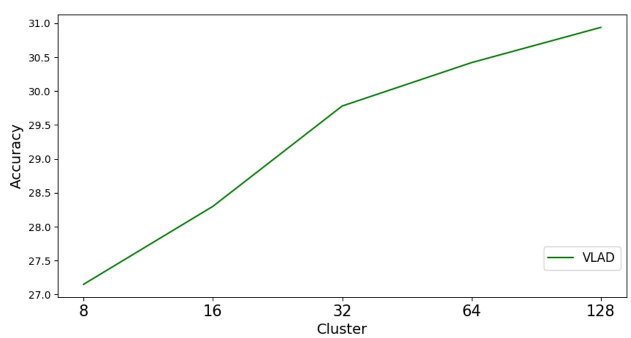

Figure 3: VLAD

4.2 Different Method & Same Dimension

Figure 5 shows the accuracy of different methods with same dimension. For a fixed dimension, the relation of cluster number among these three algorithms is

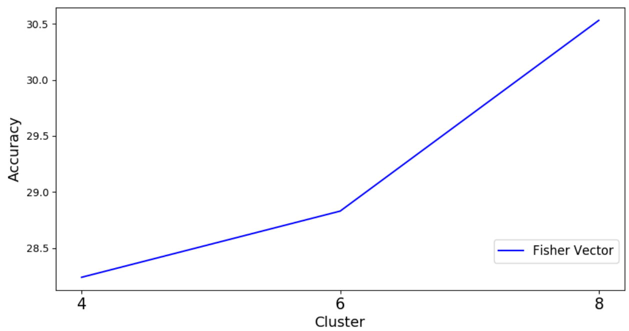

Figure 4: Fisher Vector

Bag-of-Word>VLAD>Fisher Vector. However, for the accuracy, we can see the relation becomes Fisher Vector>=VLAD>Fisher Vector. There is a huge gap between Bag-of-Word and the other two algorithms.

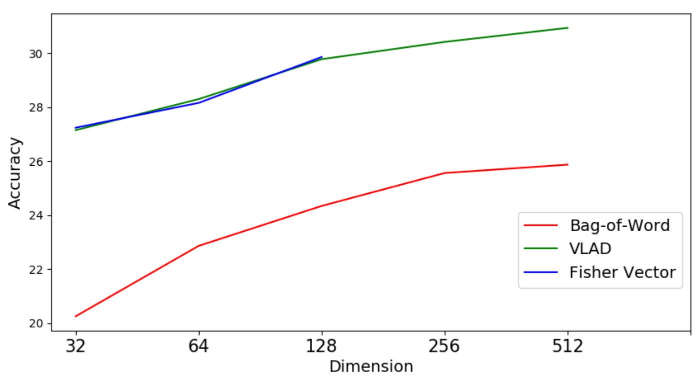

Figure 5: Different Method with Same Dimension

4.3 Same Cluster Number & Same Dimension

4.3.1 Bag-of-Word vs VLAD

Figure 6 shows the accuracy when the dimension and cluster number are all the same. In Bag-of-Word algorithm, the dimension equals to the clauster number. However, in VLAD, the dimension is larger. PCA is used to make the VLAD feature dimension equal to the cluster number. We can find that VLAD algorithm dominate the Bag-of-Word. As for speed, Bag-of-Word is much faster than VLAD.

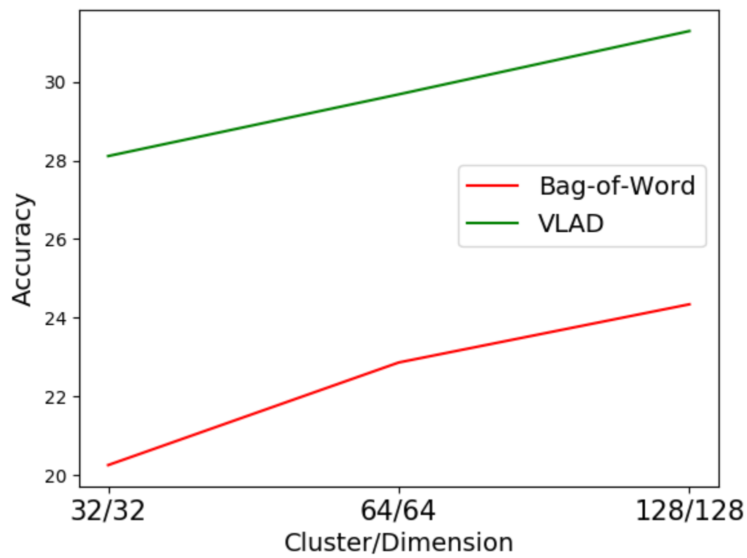

Figure 6: Bag-of-Word vs VLAD

4.3.2 VLAD vs Fisher Vector

Table 1 shows the result. When the cluster number and dimension are all the same, Fisher Vector is a little better than VLAD. However, VLAD is much faster over the speed.

Table 1: VLAD vs Fisher Vector

Method	Cluster Number	Dimension	Accuracy
VLAD	8	32	27.15
Fisher Vector	8	32	**27.40**

5 Conclusion

In this project, we use SIFT to get the keypoints and local descriptors of each image. Then, we implement three different encoding algorithms and try different cluster numbers & dimensions. We find that, for each algorithm, a larger cluster number tends to achieve a higher accuracy. If the cluster number is large enough, the improvement will become small. When fixing the parameters, on the accuracy, we can see Fisher Vector>VLAD>Bag-of-Word. However, for the speed, Fisher Vector<VLAD<Bag-of-Word. If we compare the accuracy between deep learning and these three algorithms, deep learning is much better

than these three traditional methods ($90+\%$ vs $30+\%$). But deep learning needs more computation. So computation means higher accuracy, there should be a trade-off between precision and computation in application.

6 Github Repository

We have put all the code we wrote for the assignments of this course on Github.
`https://github.com/Xiang-Gu/CS245_Principle_Data_Science`
Please feel free to check out the Github repo for any code-related issues.