Report for Data Science Project 3
Feature Encoding

Cen Xinxin, cenxinxin@sjtu.edu.cn Chen Ruizhao, stelledge@sjtu.edu.cn Li Tong, 2017lt@sjtu.edu.cn

Abstract

In this project, we tested and evaluated the performance of three feature encoding methods on image classification under Animals with Attributes(AwA2) data set. Three feature encoding methods included are Bag of Word, VLAD and Fisher Vector. We also compare the performance of SIFT features and deep learning proposals features. With this comparison, benefits and limits of different models can be revealed.

Figure 1. Target of different scales

1. Dataset and Experiment Overview

We split it into train set, consisting of 22390 images(60%), and test set, consisting of 14932 images(40%). For each feature encoding method, the best k is chosen by cross-validation and clusters are trained with SIFT features or deep learning features of both training set and test set. For BOW and VLAD, we choose C for a linear SVM based on former experiments. For Fisher vector, we choose C by validation.

More details about experiments are shown in relevant sections.

2. Local Descriptors

2.1. SIFT

2.1.1. Introduction

Scale-invariant feature transform aims to find features of key points.[1] It comes from an intuition that key points we human used to recognize a target is invariant from scales.

It first build a scale-space to mimic the different images of the target in the retina when people are at different distance from the target. The larger scale the image is, the blurrier it is. To vision, images of different scale varies in gray resolution and contrast resolution. Also, the scale-space mimics different images from different views. However, two images different from each other in mentioned perspectives should have the same key points.

The generation of scale-space needs to use Gaussian blur. It uses the normal distribution to calculate the blur template, and uses the template to do convolution operation with the original image to achieve the purpose of blur image.

Then, Gaussian pyramid is constructed. Gaussian blur is applied to the image of some size. Several blurred image sets form an octave. Then the most blurred image of this octave is sampled – the length and width are shortened by twice. The reduced image will be the initial image of the next octave. The difference between the two adjacent images of the same octave is used to get the interpolation image. The set of all the interpolation images of octave constitutes the difference of Gaussian(DoG).

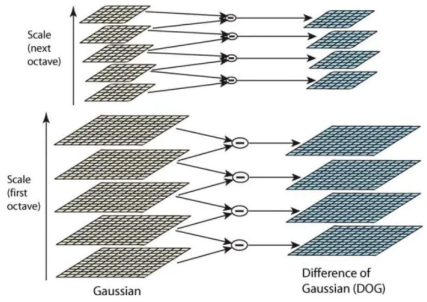


Figure 2. Construction of DoG

Search the scale space and we can find extreme points, which are the key points we are looking for. Then, we calculate the dominant direction and rotate to that direction. Finally, we split this district as 4×4 sub-districts and count up gradients in 8 direction. So, each of our SIFT features

has 128 dimensions.

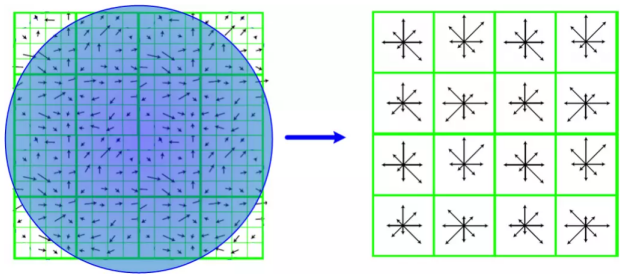


Figure 3. SIFT features

2.1.2. Implementation

We use *cv2* library to extract SIFT features.

We first convert the original images to gray space to reduce computing time.

While calculating key points, we find that the number of points does varies with its size, which seems contrary to theory. This is because some key points may be merged and some too small district may vanish when we reduce the size.

To control the number of key points, we set contrast threshold to 0.02 so that each image have over 10 key points. Also, considering limited storage space, we resize each image with size over 100000 to lessen the number of key points as in Figure 4

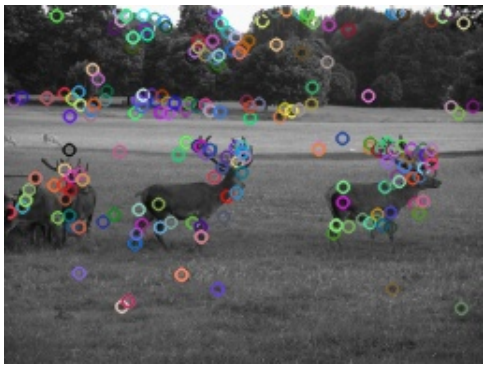


Figure 4. SIFT key points of deer 10048

2.2. Deep Learning Features of Proposals

2.2.1. Introduction

We use Selective Search(SS)[2] to extract proposals from images. SS can get target region of different scales and is more efficient than exhausting all regions. It is of great use when there are multiple targets in a single image. Another advantage of SS is that its diversification. It use color, texture, size and other strategies to merge the segmented regions.

SS first use graph-based image segmentation[3] to get region candidates. Then, it uses the greedy strategy to calculate the similarity of every two adjacent regions, and merge the most similar pair each time, as in Figure 5.

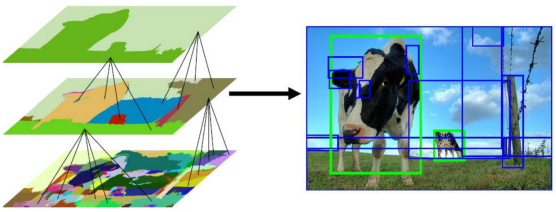


Figure 5. Hierarchical Aggregation of Regions

SS has four measures of similarity. The first one is color similarity. Using L1-norm normalization to obtain the histogram of 25 bins of each color channel of the image. Then, the colour information of each region is a vector of 25×3 dimension. The similarity is the intersection of these histograms.

$$S_{colour(r_i,r_j)} = \sum_{k=1}^{n} min(c_i^k, c_j^k)$$

The second one is texture similarity. Here adopts SIFT-like features. The Gaussian derivative with variance $\sigma = 1$ is calculated for 8 different directions of each color channel, and the histogram of 10 bins in each direction of channel is obtained by L1-norm. The texture information of each region is a vector of 10 dimension. The similarity is the intersection of these histograms.

$$S_{texture(r_i,r_j)} = \sum_{k=1}^{n} min(t_i^k, t_j^k)$$

The third measure is about the region size. To avoid the merged area continuously engulfs its surrounding area, SS endow smaller regions with higher weights.

$$S_{size}(r_i, r_j) = 1 - \frac{size(r_i) + size(r_j)}{size(im)}$$

The fourth measure is whether the two regions are more consistent. If they are closer or even included, they are more likely to be merged. SS calculate the bounding box's(BB_{ij}) area of the merged region to quantify the consistence.

$$S_{fill}(r_i, r_j) = 1 - \frac{size(BB_{ij}) - size(r_j) - size(r_j)}{size(im)}$$

The four measures are combined.

$$S = a_1 \cdot S_{colour} + a_2 \cdot S_{texture} + a_3 \cdot S_{size} + a_4 \cdot S_{fill}$$

2.2.2. Implementation

We use *selectivesearch* library[4] to perform SS.

We set $scale = 500$, for a larger scale causes a preference for larger components. We choose a dynamic minimum size of the regions $min\_size = w \times h/200$. A sample is shown in Figure 6.

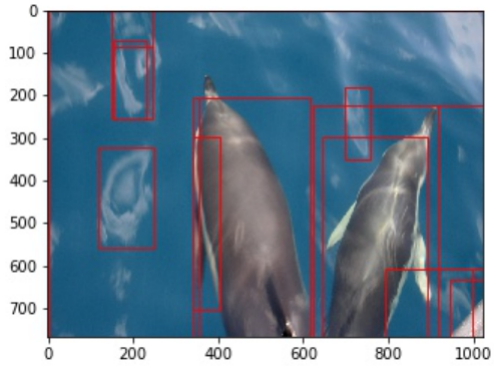


Figure 6. Proposals of dolphin_10105

One thing worth-mentioning is that there's an image in the data set who has one channel (in gray space), corresponding transformation is conducted.

To get the deep learning features of proposals, we use a pre-trained ResNet101 model from the *torchvision.models* library. The deep learning features are of 2048 dimensions.

3. Feature Encoding

3.1. Bag of Words

3.1.1. Discription

The bag-of-words model (BOW) is a way of representing local features by extracting 'words' from all local features and count the frequency of certain words appeared in a sample. Words are decided with K-means algorithm with parameter $n\_clusters = k$. Then the presence time of all words are counted to find corresponding global feature, as is shown in Figure 7.

3.1.2. Configuration

In order to speed up clustering task, Mini-batch K-means algorithm is applied. 10^6 SIFT or 5×10^6 Proposal local descriptors are sampled from original dataset, which are divided into batches including 10^4 samples. These samples are used to get codebook for the following jobs.

Considering that embedded global descriptors are high dimensional and vary in ranges of features, Z-norm and PCA are applied. We use z-norm to standardize global descriptors at first, then use PCA for dimension reduction. PCA is configured with $n_components = 0.95$, which decides how many dimensions to preserve automatically.

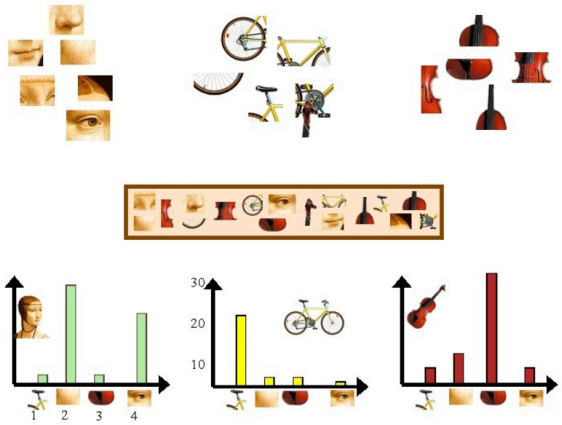


Figure 7. BOW:local features to words, words to frequency

SVM's parameter is configured as $c = 0.1$.

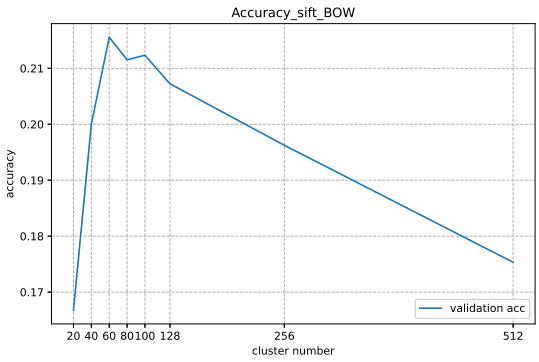
3.1.3. Result

According to our earlier preparation experiments, accuracy on validation varies prominently in low dimensional cased. Thus final cluster numbers k are chosen from 20, 40, 60, 80, 100, 128, 256 and 512. Comparison between two types of features is shown in Figure 8.

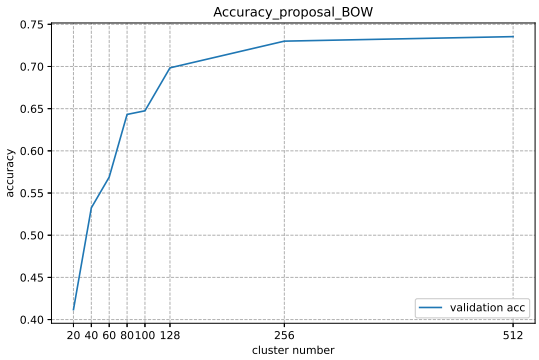
k	SIFT	Proposal
20	0.1668	0.4118
40	0.2000	0.5324
60	**0.2155**	0.5689
80	0.2115	0.6432
100	0.2123	0.6475
128	0.2072	0.6984
256	0.1962	0.7300
512	0.1753	**0.7354**
test	0.2024	0.7495

Table 1. BOW Accuracy

SIFT features have shown a peak value on $k = 60$,where we have a 0.2024 accuracy. Increasing k over 60 only damages performance greater when it's far from the peak. However in proposal features case we didn't witness this peak. Relatively, we find the accuracy simply grows with increasing cluster number.



(a) SIFT



(b) Proposal

Figure 8. BOW results: (a) SIFT (b) Proposal

Theoretically this is because of the underlying real distribution of 'words' in local descriptors' space. There is high probability that only around 60 types of words exists in SIFT features. Thus bigger k induces more noise, which leads to bad results. Proposal method produces 2048-dim features and brings more latent words into consideration, so larger cluster number works better.

Also, an obvious different between two types of features is that SIFT has lower accuracy and performs worse faster as k grows. Further test on Proposal features shows an accuracy of 0.7469 on $k = 1024$ (see in Figure 9), which has no sign of losing effectiveness.

We consider this as the different representing ability of chosen global features. In BOW algorithm, the frequency of local descriptors are evaluated. Relation between the existence of local feature and final class label decides the presentation ability of local features. Proposal features, which involves object detection task originally, has a close tie to the classification result, encoded more information of animal into the presence of it. However, SIFT feature makes relatively small contribution to the classification result.

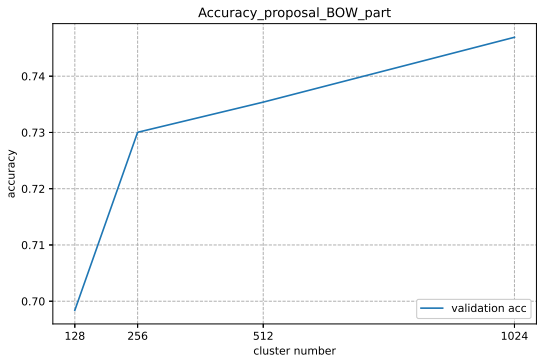


Figure 9. Proposal+BOW accuracy: Additional experiment

3.2. VLAD

3.2.1. Discription

VLAD (Vector of Locally Aggregated Descriptors) is an improved algorithm of feature encoding that encodes first order statistics into the extracted features. This is accomplished by adding up the differences between local descriptors and its closest word. Procedure shown in Figure 10.

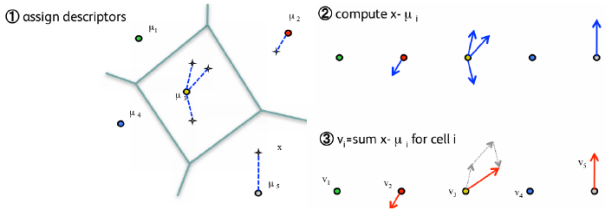


Figure 10. VLAD:local features to words, words to sum of differences

3.2.2. Configuration

Clustering configuration, sampling method and data preprocessing method is the same as BOW.

3.2.3. Result

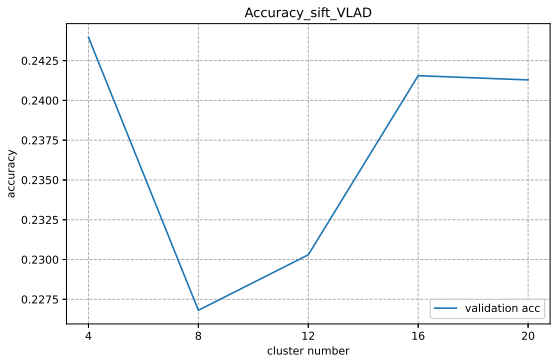
VLAD descriptor is of $k \times N$ dimensions, in which N represents the dimension number of local features. Thus we used smaller number 4, 8, 12, 16 and 20 for SIFT, 2, 4, 8, 12 and 16 for Proposal.

VLAD method surely showed its advantages against BOW method by having higher accuracy. Encoding first order statistics benefits algorithm obviously.

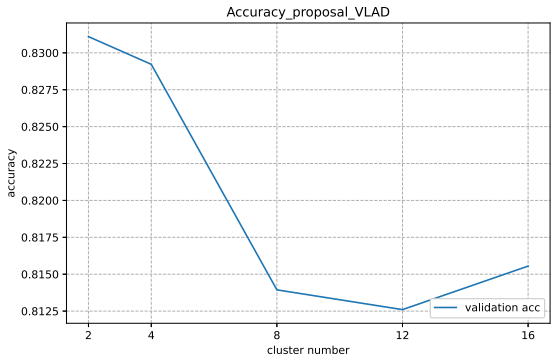
Although Proposal still outperformed SIFT this time, a new phenomenon appeared. There is a canyon in accuracy under

k	SIFT	Proposal
2		**0.8311**
4	**0.2440**	0.8292
8	0.2268	0.8139
12	0.2303	0.8126
16	0.2416	0.8155
20	0.2413	
test	0.2460	0.8227

Table 2. VLAD Accuracy



(a) SIFT



(b) Proposal

Figure 11. VLAD results: (a) SIFT (b) Proposal

both types of feature. In SIFT it's $k = 8$ while in Proposal it's $k = 12$. This may due to two conflicting source of VLAD 's effectiveness. One is the correctness of clustering, the other is the distribution of local features around words. When k is small, words are badly chosen but the sum of differences can compensate for this. When k is larger, words are chosen correctly but the sum of differences induces noise.

Thus we can assume that when k is a lot more larger, there will be another peak of accuracy. However we can't afford to calculate that scale of k, since VLAD descriptor's dimension is $k \times N$.

3.3. Fisher Vector

3.3.1. Discription

Fisher vector is essentially a gradient vector by likelihood function to express an image. The meaning of this gradient vector is to describe the direction in which parameters should be modified to best fit the data. For an image, if there are T local features, this image can be expressed as $X = \{x_t, t = 1 \dots T\}$. These local features x_t conform to a certain distribution and these distributions are independent of each other. So $p(X|\lambda) = \prod_{t=1}^{T} p(x_t|\lambda)$, where $\lambda = \{\omega_i, \mu_i, \Sigma_i, i = 1 \dots K\}$. After taking the logarithm,

$$\mathcal{L}(X|\lambda) = \Sigma_{t=1}^{T} \log p(x_t|\lambda) \tag{1}$$

Now we need a set of linear combinations of K Gaussian distributions to approximate these independent identical distributions. Assuming these Gaussian mixture distribution parameters are also λ, then

$$p(x_t|\lambda) = \Sigma_{i=1}^{N} \omega_i p_i(x_i|\lambda). \tag{2}$$

where p_i represents the Gaussian distribution

$$p_i(x|\lambda) = \frac{exp\{-\frac{1}{2}(x-\mu_i)'\Sigma_i^{-1}(x-\mu_i)\}}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \tag{3}$$

ω represents the coefficient of the linear combination $\Sigma_{i=1}^{N}\omega_i = 1$. D is the dimension of the feature vector, and the covariance matrix Σ_i^{-1} calculates the relationship between different dimensions. Σ_i^{-1} is diagonal matrix, that is, different dimensions of the feature are independent of each other.

We define $\gamma_t(i) = \frac{\omega_i \mu_i(x_t)}{\Sigma_{j=1}^{K}\omega_j\mu_j(x_t)}$ as occupancy probability, that is, the probability of feature x_t generated by the i-th Gaussian distribution.

Then according to formula 1,2,3, calculate the partial derivative, which is the gradient, as the fisher vector.

$$\begin{aligned}
\frac{\partial\mathcal{L}(X|\lambda)}{\partial\omega_i} &= \Sigma_{t=1}^{T}[\frac{\gamma_t(i)}{\omega_i} - \frac{\gamma_t(1)}{\omega_1}] \text{ for } i \geq 2, \\
\frac{\partial\mathcal{L}(X|\lambda)}{\partial\mu_i^d} &= \Sigma_{t=1}^{T}\gamma_t(i)[\frac{x_t^d - \mu_i^d}{(\sigma_i^d)^2}], \\
\frac{\partial\mathcal{L}(X|\lambda)}{\partial\sigma_i^d} &= \Sigma_{t=1}^{T}\gamma_t(i)[\frac{(x_t^d - \mu_i^d)^2}{(\sigma_i^d)^3} - \frac{1}{\sigma_i^d}].
\end{aligned} \tag{4}$$

In order to normalization, the three variables of formula 4

are introduced by three corresponding fisher matrix:

$$
\begin{aligned}
f_{\omega_i} &= T(\frac{1}{\omega_i} + \frac{1}{\omega_1}) \\
f_{\mu_i^d} &= \frac{T\omega_i}{(\sigma_i^d)^2} \\
f_{\sigma_i^d} &= \frac{2T\omega_i}{(\sigma_i^d)^2}.
\end{aligned} \tag{5}
$$

Therefore, the normalized fisher vector is

$$
\begin{aligned}
& f_{\omega_i}^{-1/2} \partial \mathcal{L}(X|\lambda)/\partial \omega_i \\
& f_{\mu_i^d}^{-1/2} \partial \mathcal{L}(X|\lambda)/\partial \mu_i^d \\
& f_{\sigma_i^d}^{-1/2} \partial \mathcal{L}(X|\lambda)/\partial \sigma_i^d
\end{aligned} \tag{6}
$$

Since each feature is d-dimensional, K linear combinations of Gaussian distributions are required. According to formula 6, the dimension of a Fisher vector is $(2 * d + 1) * K$ dimension.

3.3.2. Result

Since our dataset is large, we sample 10000 local features uniformly in dataset to generate GMM and learn a codebook. And then we use fisher vector to encode feature of each image. Finally we use linear SVM to classify each image. Since the dimension of features may be large, we use PCA to reduce dimension. We compare the performance of SIFT descriptors and proposals, and test different cluster numbers.

Firstly, we use cross-validation within the training set to determine hyper-parameters C in SVM. The validation results of SIFT is shown in Figure 12. The validation results of proposal is shown in Figure 13.

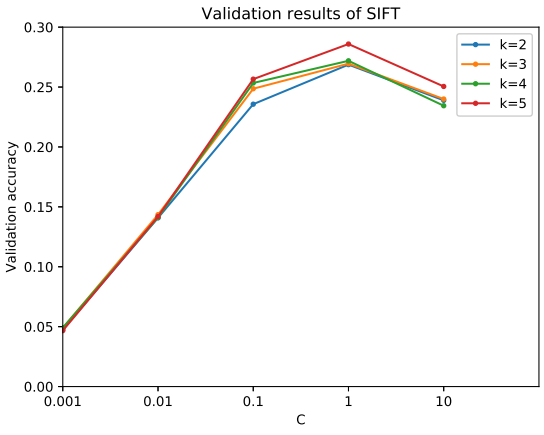


Figure 12. Validation results of SIFT

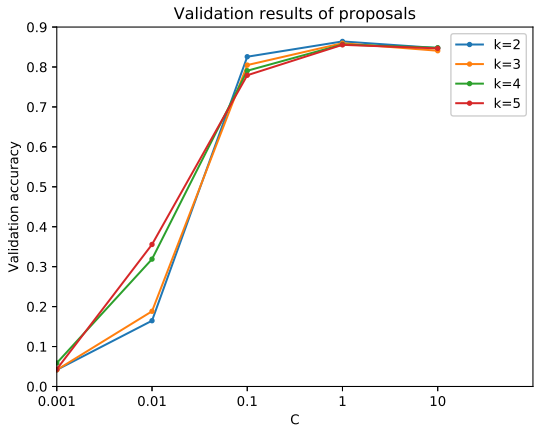


Figure 13. Validation results of proposal

We can see that $C = 1$ get the highest accuracy respectively in both SIFT and proposal cases. So we set $C = 1$ in the following testing section.

Then we set $K = 2, 3, 4, 5$ to generate GMM and use PCA to reduce feature dimension for image classification. The testing accuracy results of SIFT are shown in Table 3. And the testing accuracy results of proposals are shown in Table 4.

k	original	PCA_512	PCA_256	PCA_128	PCA_64
2	0.2589	0.2584	0.2562	0.2446	0.2361
3	0.2668	0.2638	0.2581	0.2475	0.2408
4	0.2691	**0.2699**	0.2616	0.2540	0.2459
5	0.2687	0.2649	0.2604	0.2553	0.2513

Table 3. Fisher Vector Accuracy by SIFT

k	PCA_2048	PCA_1024	PCA_512	PCA_256
2	0.8394	0.8425	0.8427	0.8391
3	0.8467	0.8456	0.8453	0.8431
4	0.8442	**0.8457**	0.8432	0.8415
5	0.8450	0.8449	0.8438	0.8366

Table 4. Fisher Vector Accuracy by proposals

From the results, we can see that SIFT and proposals both get the highest accuracy when $k = 4$. One possible reason is that 4 Gaussian distributions may fit the data well. We also observe that accuracy decrease slightly, even increase in some cases, by using PCA. It confirms that PCA can hold the main features and remove noise to reduce dimension and speed up calculation. Besides, we can obviously see that by using deep learning proposals, the accuracy performance compared to SIFT has been greatly improved. We think there are two reasons:(1)proposals can filter the candidate

regions. By this way, 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 in SIFT descriptors, there are many irrelevant descriptors. (2)ResNet101 model we used is very powerful. The features extracted by it are more expressive and more distinguishable than the SIFT features.

4. Conclusion

In this project, we tried to extract local descriptors from images and used different feature encoding methods to change local features to global feature. Firstly, we used SIFT algorithm and selective search to extract descriptors and proposals for each image respectively. Then we used three different feature encoding methods (BOW, VLAD, Fisher Vector) to convert the descriptors or proposals to feature vector. Finally, we feed the feature vectors to SVM for image classification. Table 5 summarizes all our experiments' results.

Features	Model	Cluster	Accuracy
SIFT	BOW	60	20.24%
SIFT	VLAD	4	24.60%
SIFT	Fisher Vector	4	**26.99%**
proposals	BOW	512	74.95%
proposals	VLAD	2	82.27%
proposals	Fisher Vector	4	**84.57%**

Table 5. Our experiments' results

From the perspective of accuracy, Proposals $>>$ SIFT and Fisher Vector $>$ VLAD $>$ BOW. Because there are less noise points and more distinguishable features in deep learning proposals than SIFT. And Fisher Vector converts more information to feature vector than other two methods. From the perspective of speed, SIFT $>$ Proposals and BOW $>$ VLAD $>$ Fisher Vector. Because extracting proposals from each image needs more computing resource than SIFT algorithm. And the dimension of BOW feature vector is equal to the cluster number (k), which is much faster than other two methods ($k \times d, 2k \times d + k$). As a result, if the computing resource is enough, we think Fisher Vector is the best feature encoding method. If not, we think VLAD is the best choice.

Acknowledgements

We thank Prof. Niu Li and Cong Wenyan, the teaching assistant, for helping us in the experiments.

Reference

[1] Scale-Invariant feature transform

https://www.jianshu.com/p/e25562a87cca

[2] Uijlings, Jasper RR, et al. "Selective search for object recognition." International journal of computer vision, 104(2) (2013): 154-171.

[3] P. Felzenszwalb, D. HuttenlocherInternational Journal of Computer Vision, Vol. 59, No. 2, September 2004

[4] selectivesearch library

https://github.com/AlpacaDB/selectivesearch