Computer Vision Report

Yecheng Chu
10319044
May 2022

Introduction

Object recognition and classification is a popular task in computer vision. In this project, a classical computer vision algorithm, Bag-of-Visual-Words(BoVW), and a deep learning model, Convolution Neural Network(CNN) is implemented for image classification using the dataset CIFAR10. The performance of the two methods is analysed and discussed in this report. All the measurements were taken using a MacBook Pro with 2 GHz Quad-Core Intel Core i5 processor.

BoVW

To implement the BoVW method, feature extraction is first performed. Then the extracted local features are clustered to form visual vocabulary. After that, for each image, a frequency histogram is plotted. Finally, classification is performed to classify test images to different classes. There are several choices when implementing this method and their pros and cons as well as their effects on the performance of BoVW is discussed below.

Local Features

The student uses two main approaches for feature detection and representation. The first is using SIFT descriptor. It has the advantage of being able to handle viewpoint change up to some degree and being able to handle image translation, rotation and scale change. It also copes well with the change in illumination. In addition, individual feature extracted by SIFT has very distinctive descriptor, which allows a single feature to find its correct match with good probability in a large database of features. Moreover, it is a fast and efficient approach. However, there are also disadvantages such as it will lose some spatial information and is not performed well for images been blurred. In addition, the student also performed SIFT on the interesting points detected by Harris detector. Harris is a corner detector, it can provide informative features and could handle intensity change as well as translation and rotation, but it cannot deal with scale change. Another approach adopted is representing images as patches using extract_patches_2d function in sklearn library, it is a simple approach but have many parameters to specify such as patch size and maximum number of patches extracted. Also, it will extract huge number of features and would be very slow for clustering algorithm afterwards. Due to the huge computation required, the comparison for the two local feature extraction approaches is only performed when cluster number is 90 using the two classifiers, KNN and SVM respectively. The results are shown below.

Classifier	Method	Accuracy	F1 Score
KNN	SIFT	0.165	0.15866
	Harris + SIFT	0.153	0.14756
	Patches	0.235	0.23422
SVM	SIFT	0.263	0.25522
	Harris + SIFT	0.256	0.25073
	Patches	0.316	0.31141

Table 1 Accuracy and F1 score of SIFT and Patches

It is obvious that for both KNN and SVM classifier, the feature extraction using patches performs better than SIFT in terms of both accuracy and F1 score. The confusion matrix for SVM using patches is plotted in Figure 1.

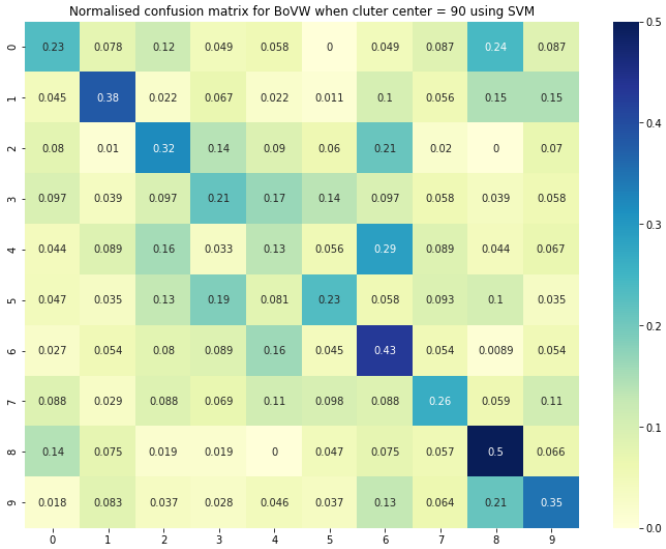


Figure 1 Confusion matrix for Patches with SVM

The reason for the low accuracy using SIFT could be the relatively low number of features extracted from images. However, for the patches, there are too many features been extracted. Although they provide better matches due to the more features extracted, they could be extremely slow for further processing. As a result, for other choices, SIFT is chosen compromising between the accuracy and time consumption.

Visual dictionary

This section mainly concerns the number of cluster centers for the clustering algorithm. The cluster center number decides the vocabulary size. If the number is too small, the visual words may not be representative for all patches. If it is too large, it could lead to quantisation artifacts and overfitting. As a result, the size of the visual dictionary needed to be carefully chosen. In the experiment, the student chosen to use SIFT with KNN and vary the number of cluster centres and observe the results.

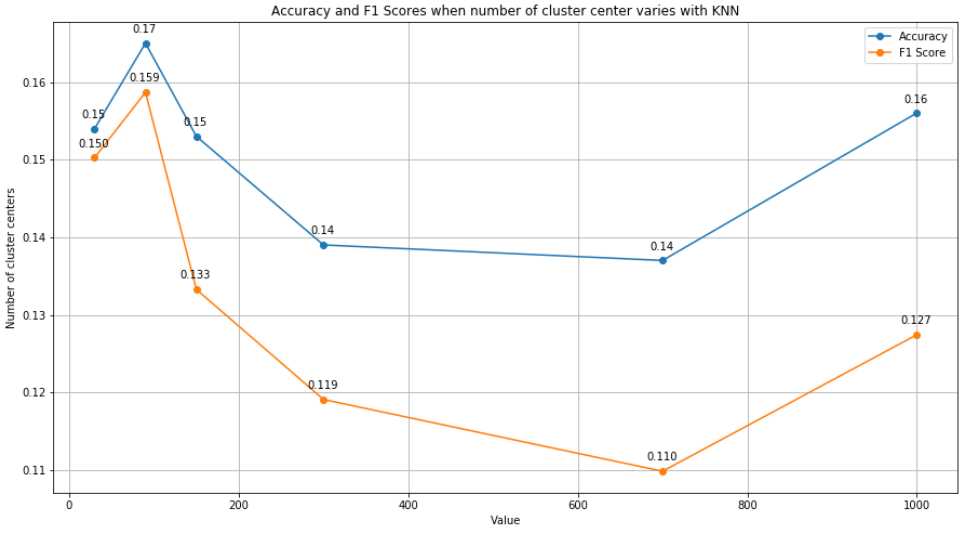


Figure 2 Comparison between different number of cluster centers

According to the experiment, the optimal number of cluster centres occurs at 90. It is clearly shown that the accuracy and F1 score follow a similar trend, they first show an increase and then decrease when the number of cluster centres increases. After reaching a minimum, it shows another rising trend when the number of centre increases. It is possible that an increase in number of clusters could lead to better performance, 90 is still chosen due to computational efficiency consideration. It could take several hours for the clustering algorithm, such as K means, to run when the number of cluster centres is greater than 1000.

Classifier

The last choice is the classifier used for test image classification. Three different classifiers are used. Two of them are discriminative classifiers, KNN and SVM. KNN is a simple and instance-based learning algorithm. It has the advantage of relatively faster in training and it is useful for both linear and non-linear datasets. It could also handle multi-class cases and can have good performance if there are enough representative data. However, it has a comparatively low testing speed and is not very suitable for high dimensional data. It also has a huge data storage requirement. SVM offers high accuracy and has a satisfactory prediction speed. Unlike KNN, it could handle high dimensional data and it is also less affected by outliers. Nevertheless, it is not suitable for large datasets due to the long training time. The last classifier used is a generative classifier called Naïve Bayes Classifier (NB). NB is simple to implement and is very fast and accurate. It can work on huge datasets efficiently. However, NB needs the hold of assumption that each feature is independent. Also, it could have zero probability problem. A comparison of the accuracy and F1 score with different number of cluster centres for the three classifiers is shown below.

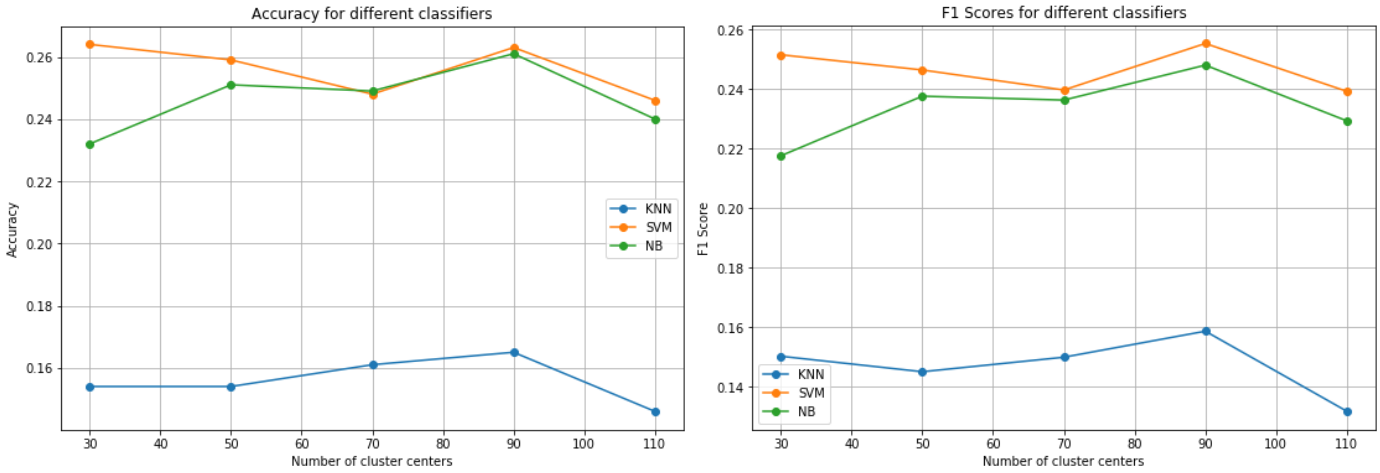


Figure 3 Accuracy (Left) and F1 Score (Right) for different classifiers

Classifier	Measure	K = 30	K = 50	K = 70	K = 90	K = 110
KNN	Accuracy	0.154	0.154	0.161	**0.165**	0.146
	F1 Score	0.15026	0.14506	0.14993	**0.15866**	0.13180
SVM	Accuracy	**0.264**	0.259	0.248	0.263	0.246
	F1 Score	0.25144	0.24633	0.23961	**0.25522**	0.23916
NB	Accuracy	0.232	0.251	0.249	**0.261**	0.24
	F1 Score	0.21753	0.23753	0.23623	**0.24796**	0.22926

Table 2 Comparison between different classifiers

From the experimental results, it is observed that SVM and NB performs much better than KNN in both accuracy and F1 score. It could be due to the implementation of KNN. The student set the number of neighbours for KNN to be 1. The accuracy and F1 score should be higher if the number of neighbours is increased. However, due to the influence of outliers, it would still get a lower performance compared with SVM and NB. Overall, considering both accuracy and F1 score, the SVM classifier with number of clusters equal to 90 could get a best performance when using SIFT as the local feature extraction method. However, just as shown in the Local Features section, both the accuracy and F1 score could be further improved at this number of cluster centres when using extract_patches_2d for feature extraction.

CNN

The BoVW method is an order-less representation of data that means it would lose spatial information. In order to deal with this problem, we could use CNN. The topology of my CNN network is shown below.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	896
activation (Activation)	(None, 32, 32, 32)	0
conv2d_1 (Conv2D)	(None, 32, 32, 32)	9248
activation_1 (Activation)	(None, 32, 32, 32)	0
max_pooling2d (MaxPooling2D)	(None, 16, 16, 32)	0
dropout (Dropout)	(None, 16, 16, 32)	0
conv2d_2 (Conv2D)	(None, 16, 16, 64)	18496
activation_2 (Activation)	(None, 16, 16, 64)	0
conv2d_3 (Conv2D)	(None, 16, 16, 64)	36928
activation_3 (Activation)	(None, 16, 16, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 8, 8, 64)	0
dropout_1 (Dropout)	(None, 8, 8, 64)	0
conv2d_4 (Conv2D)	(None, 8, 8, 64)	36928
activation_4 (Activation)	(None, 8, 8, 64)	0
conv2d_5 (Conv2D)	(None, 2, 2, 64)	36928
activation_5 (Activation)	(None, 2, 2, 64)	0
max_pooling2d_2 (MaxPooling2D)	(None, 1, 1, 64)	0
dropout_2 (Dropout)	(None, 1, 1, 64)	0
flatten (Flatten)	(None, 64)	0
dense (Dense)	(None, 512)	33280
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5130
activation_7 (Activation)	(None, 10)	0

Figure 4 CNN Topology

It follows a similar structure for the first three blocks, 2 convolution layers with activation layers using ReLU as the activation function followed by a maxpooling layer with a dropout layer to prevent overfit. The final block consists of a dense layer with ReLU activation followed by a dropout layer and a dense layer with Softmax actionvation to classify the data into different categories. The kernel size in the convolution layers is all set to be 3 and that of the maxpooling layer is all set to be 2. The dataset used in the CNN is CIFAR 10, which is same as that in BoVW. The data was first converted to categorical and then normalised. The hyperparameter selection is separate into two stages. In the first stage, the student tends to find the best batch size, optimiser, learning rate for optimiser and epoch pair which gives best performance. With this best hyperparameter pair, the student carries the second stage to modify the hyperparameters in the model and change the model structure to achieve a better result.

The first hyperparameter to choose is batch size for this model. It is an important hyperparameter as if the batch size is too small, the model may not converge to the global optimal. While if it is too large, the quality of the model would also be affected as it decreases the ability to generalise. The RMSprop optimiser and epoch size 20 is chosen as a start, and the batch size being testing is 32, 64, 128, 256, 512. The result is shown in the figure below. It is clearly shown that the batch size 128 is the one with lowest loss and highest accuracy.

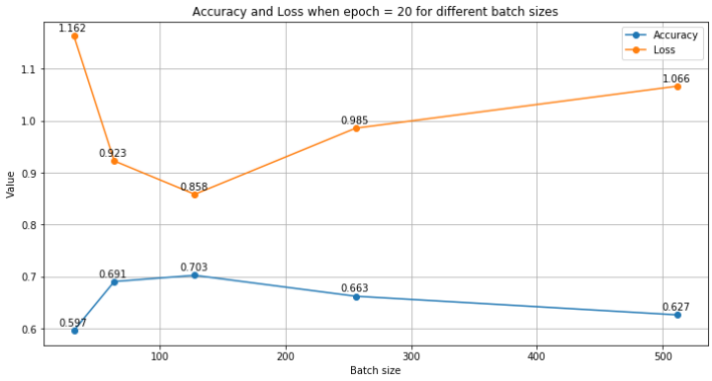


Figure 5 Batch size selection

Then it comes to optimiser and their learning rate selection. Using the optimal batch size 128, several optimisers are adopted, and their performance is compared using their default learning rate. From the experiment shown in left figure in Figure 6, Adam and NAdam are the two best optimisers, and they are further compared using different learning rates. Learning rate controls the amount to change according to the

estimated error each time the model weight is updated. A small learning rate may result in a long training process and is not efficient. On the other hand, a large learning rate could lead to an unstable training process. Also, using a large learning rate could potentially skip the optimal weight as the weight adjustment is too large. By comparing the result get in the right figure in Figure 6, the student decided that Adam optimiser with the learning rate 0.001 is the optimal choice for the learning rate.

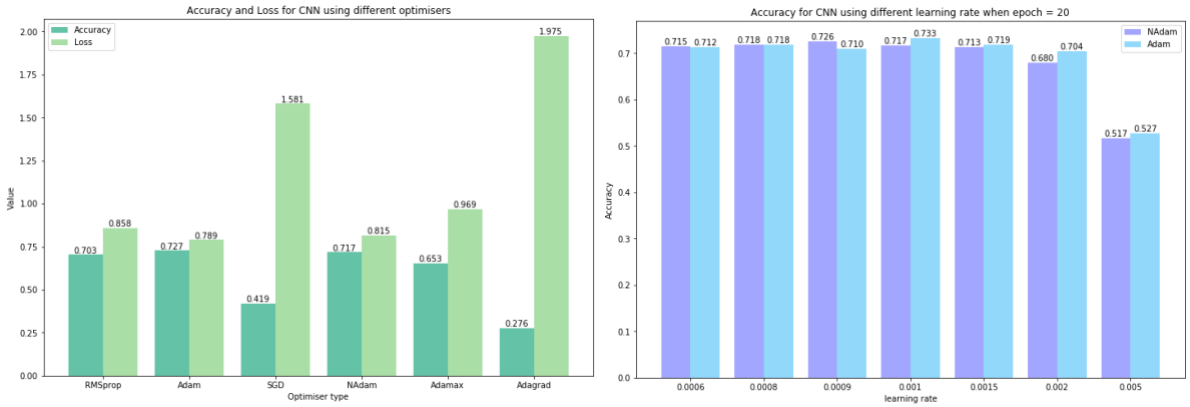
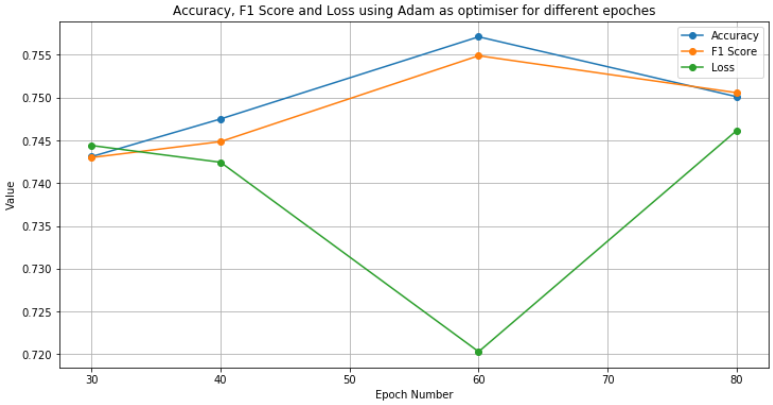


Figure 6 Optimiser selection (Left) and Learning rate selection (Right)

The next hyperparameter to choose is the epoch number. A high epoch number will result a very long training process. Also, it could lead to overfitting where the trained model fits the training sample too well that it loses the ability for generalisation. While if the epoch number is not large enough, it could cause underfitting. The training process stops before an optimal model is trained. Apparently, the epoch number 20 used before is a compromise between accuracy and time consumption, it is not large enough for the best performance. As a result, during the optimal epoch selection, the epochs used starts from 30. And they are 30, 40, 60 and 80 respectively. The results are shown in Figure 7.



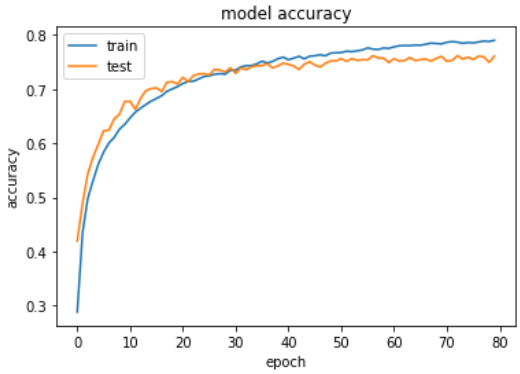


Figure 7 Epoch Selection (Left) *Figure 8 Epoch Selection(Right)*

It demonstrates that the epoch number 60 is the best choice, where the model achieves best accuracy and F1 score while having the minimum loss. At epoch 60 batch size 128 using Adamax with learning rate 0.001 as optimiser, the testing accuracy is 0.757. The model accuracy for training and testing for the epoch 80 is shown in Figure 8. It illustrates that the testing accuracy is no longer increased after 60 epochs while the training accuracy is still increasing suggesting the overfitting has happened.

In stage two, some modification on the CNN model is performed. The first is changing the dropout rate from [0.25, 0.25, 0.25, 0.5] to [0.2, 0.3, 0.4, 0.5]. Another improvement is using weight decay to reduce overfitting by add L2 regularisers on dense layer. The third improvement is using learning rate decay, it is empirically observed to help both optimisation and generalisation. By using Keras's callback function LearningRateScheduler, the student kept the learning rate for the first 30 epochs the same and start to decrease the learning rate afterwards. The last improvement is using batch normalisation. It is a technique

that can improve model's fitting ability, ensure the stability and accelerate the training process. During the implementation, the student chosen to add a batch normalisation layer before each activation layer. Table 3 summarises the accuracies when those improvements are applied.

Method	Accuracy
Changing dropout + Weight decay	0.78961
Changing dropout + Weight decay + Learning rate decay	0.81438
Changing dropout + Weight decay + Learning rate decay + Batch normalisation	0.83340
Changing dropout + Learning rate decay + Batch normalisation	0.83960

Table 3 Accuracies for the improvement

It is shown that with all those improvements, the accuracy could further increase to 0.833. The reason for the lower accuracy with weight decay could because the epoch number 60 is not large enough for overfitting to appear after all the improvements being applied. And the batch normalisation also has a modest regularisation effect. Finally, the accuracy and loss of the model as well as the confusion matrix is plotted using the pair achieving best accuracy, 0.83960. Meantime, the F1 score achieved for this setting is 0.83965.

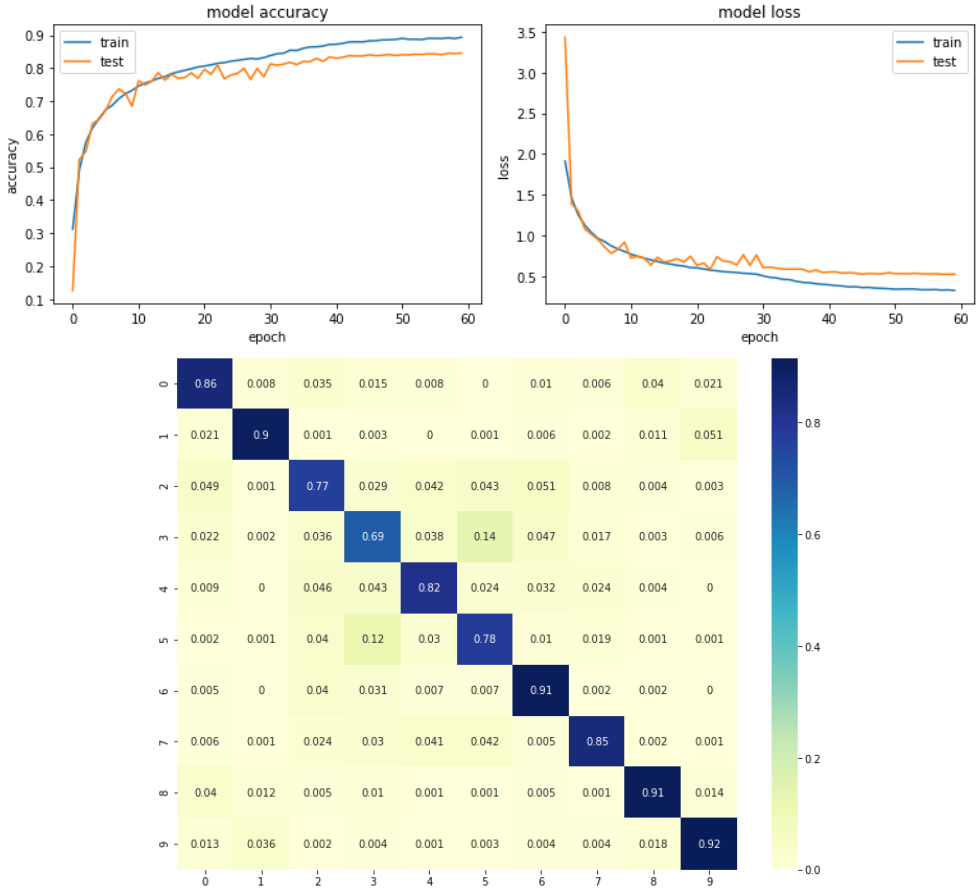


Figure 9 Accuracy (Upper Left) and Loss (Upper Right) for the model and Confusion matrix (Lower)

As a further exploration, the data augmentation could be used to increase the number of training samples without collecting new images. By training with a larger dataset, the CNN has the potential to achieve a better performance. In addition, the kernel size in the convolution layer could be varied to explore for better result. To further increase the accuracy, ResNet [1] could be adopted to replace the current CNN model defined.

Comparison, Discussion and Conclusion

The comparison of the highest achievements from the BoVW and CNN is summaried in Table 4. From the table and the confusion matrix shown in Figure 1 and Figure 9, it is clearly shown that, for this image classification

task using CIFAR10 as dataset, CNN performs much better in terms of both accuracy and F1 score. The poor performance of BoVW could due to several drawbacks of this classic computer vision algorithm.

Method	Accuracy	F1 Score
BoVW	0.316	0.31141
CNN	0.840	0.83965

Table 4 Comparison between BoVW and CNN

First, a large number of key points and features are detected but a huge proportion of them is from background, so they are not very representative. Second, the clustering algorithm, K-means, used may not perform well [2]. The third issue is that BoVW considers all the features to be same weight while some features may play a more important role than others. In addition, the optimal vocabulary size is not clear. Although attempts have been made to find the best setting, due to efficiency, an exhaust search could not be feasible. The last drawback which can be solved well in CNN is the losing of spatial information. In BoVW, when collecting features, the spatial information is lost. However, BoVW also has its advantages. It is able to deal with viewpoint change and is a compact summary of image content. And it has fixed length vector irrespective of number of detections. In addition, it is simple to build compared with a deep convolutional neural network and has fewer hyperparameters to choose. On the other hand, by using the 2D convolution and pooling modules, CNN preserves the spatial information of the input images. And it closely resemble mammal visual cortex. Also, CNN can automatically detect important features without human interventions. Due to the distinguishing valuable features detected, CNN usually gives high accuracy. It is very effective for image classification tasks as it uses dimensionality reduction and suits well for huge number of parameters in images. While there are also drawbacks for CNN. First of all, it takes a long time to train a deep CNN since some operations such as maxpooling are slow. And there are many hyperparameters to choose, each of them could have an effect on the testing result. As a result, finding an optimal hypermeter set is time-consuming. In addition, CNN is not that robust with rotation and reflection, so a large dataset is needed to train the model. If the dataset for training a deep CNN is not large enough it may lead to overfitting. Moreover, CNN is not good at handle adversarial attacks [3]. From the experimental results, apparently CNN is more suitable for the classification task on this dataset. Although, it could be due to the images in CIFAR10 has some blur that creates difficulty in feature extraction in BoVW such as using SIFT. And BoVW may also achieve a better result than CNN when the dataset is smaller, as the deep CNN may have overfitting issues. It still suggests that CNN is better at image classification tasks. The findings of the student's experiment are the same as the conclusions made from relevant studies comparing BoVW and CNN [4][5]. It illustrates that the better performance of CNN over BoVW is not due to specific dataset and CNN is more suitable for the computer vision domains such as feature extraction, image classification and recognition.

Reference

[1] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[2] Chanti, Dawood Al, and Alice Caplier. "Improving bag-of-visual-words towards effective facial expressive image classification." arXiv preprint arXiv:1810.00360 (2018).
[3] Yang, Tao Andy, and Daniel L. Silver. "The Disadvantage of CNN versus DBN Image Classification Under Adversarial Conditions."
[4] Okafor, Emmanuel, et al. "Comparative study between deep learning and bag of visual words for wild-animal recognition." 2016 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 2016.
[5] Kumar, Meghana Dinesh, et al. "A comparative study of CNN, BoVW and LBP for classification of histopathological images." 2017 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 2017.