SIFT And CNN in Objects Classcification

COMP61342&41342 Cognitive Robotics And Computer Vision Assignment Report

Yuyi Yang
yuyi.yang@postgrad.manchester.ac.uk
p68393yy

I. Introduction

In recent years, equipping robots with reliable perception capabilities has become a foundation for autonomous manipulation, navigation, and human–robot interaction. Object recognition, identifying and localizing objects from camera data, remains especially challenging in unstructured, dynamic environments where lighting, occlusion, and viewpoint changes can drastically reduce accuracy.

Traditional computer-vision pipelines, based on handcrafted local features (i.e. SIFT) and classifiers (e.g. SVM, K-NN), offer interpretability and low inference latency; however, they may falter when faced with complex environment and large intra-class variability. Conversely, convolutional neural networks (CNNs) have revolutionized vision performance, automatically learning hierarchical features, but at the cost of greater computational demand and complex hyperparameter tuning. Despite extensive research in object recognition, systematic comparisons between traditional handcrafted methods and CNN-based approaches remain limited. Thus, this work aims to fill this gap by systematically benchmarking and analyzing these paradigms, providing practical insights for researchers and practitioners in robotic perception.

This work systematically compares CNN models and Local Feature methods (i.e., SIFT+BoVW) on the CIFAR-10 [1] and iCubWorld 1.0 [2] benchmark, researching how choice of feature extractor, classifier, network architecture, and training strategy affect performance of classification accuracy and inference speed, and model robustness to environmental changes.

The rest of the report is organised as follows: Section 2 makes a literature review on CNNs, Section 3 describes the methodology, detailing the model selection, design, and feature extraction process; Section 4 rpresents comprehensive experiments and results on both datasets; Section 5 offers an in-depth comparative discussion, critically evaluating accuracy, inference speed, interpretability, and robustness to environmental variations; and Section 6 concludes with future directions.

II. Literature Review

Computer vision technology plays a crucial role in robotic autonomy tasks, including target recognition, manipulation, navigation, and human-robot interaction. Early robotic vision methods typically relied on handcrafted localized features, such as by SIFT [3], [4], SURF [5] and ORB [6] extracting from the image feature points with interpretability, as well as certain invariance (to illumination, rotation and scaling) to achieve simple recognition and classification of objects. However, these methods rely too much on pre-defined features and lack the ability to represent more abstract features, making them difficult to extract effective features when facing more dynamic and variable complex real-world environments.

And over the past 15 years, especially after the successful founding of the ImageNet competition in 2012, deep learning, especially convolutional neural networks (CNNs), has dramatically reshaped computer vision research [7]. Its ability to automatically learn hierarchical feature representations of images through multilayer nonlinear transformations has led to significant advances in tasks such as image classification, target detection, and segmentation, enabling a leapfrog over traditional methods [8]. However, as the depth of the network increases, the problem of CNN models facing gradient vanishing or model performance degradation during training becomes more and more serious. In this context, the proposal of ResNet [9] provides a new idea for deep network training, which uses a residual learning mechanism to alleviate the degradation problem in deep network training by introducing skip connections (identity shortcut) to allow shallower features and the gradients flow directly to the deeper network, thus effectively alleviating the problem of gradient vanishing or explosion in deep network training.

And in recent years, the Attention mechanism and the Tansformer model [10] have made breakthroughs in the field of computer vision, which is able to capture long-distance dependencies in data through its efficient Self-Attention mechanism and demonstrates excellent scalability to ultra-large capacity networks and large datasets. These advantages have led to exciting advances in tasks such as image classification, target detection, and semantic segmentation using Transformer networks [11]. Vision Transformer [12] is the first to enable the use of Transformer on large-scale image data, which "completely" replace standard convolution in deep neural networks on large-scale image datasets and demonstrated performance that rivaled or even bettered CNNs. In addition, Swin Transformer [13] has further improved the computational efficiency of the Transformer model by introducing a window self-attention mechanism, achieving leading results in several visual tasks. Furthermore, recent researchers have proposed "Norm-free" Transformer [14], NoProp [15], and Model-Seeking [16] as an alternative to Transformer model and learning modalities such as direction propagation have been improved to provide new ideas for research in the field of Computer Vision.

III. Methodology

A. Dataset

CIFAR-10 and iCubWorld 1.0 are two datasets used in this project for object classification tasks. CIFAR-10 is a well-known dataset containing 60,000 32x32 color images (10,000 for test) in 10 classes, with 6,000 images per class. While, iCubWorld 1.0 is a dataset that objects are manually picked up for human and captured by iCub robot. It includes 10 categries, each categries contains 3 instances in the training set. Both dataset are famous benchmark in the field of computer vision, which is very suitable to compare the performance of local feature and deep learning methods. For each dataset, I applied corresponding data pre-processing methods separately, trying to improve the accuracy and robustness of the model, and compare the performance of different pre-processing methods.

In addition, iCubWorld 1.0 contains 4 different testsets to fit different scenarios, including: CATEGORIZATION, DEMONSTRATOR, ROBOT and BACKGROUND. In order to ensure the objectivity and precision of the experiments and to improve the efficiency of the experiments, this project did not use all four test sets. Only the CATEGORIZATION (marked as CATE) and DEMONSTRATOR (marked as DEMO) testsets are used in this project to evaluate the performance of the model. The CATE testset is used to evaluate the generalization ability of the model when facing new instances category, while the DEMO testset is used to evaluate the generalization ability of the model when facing same instances with different demonstrators (background). Thus, a comprehensive consideration based on these two test set is sufficient to assess the performance and generalization ability of the models.

B. SIFT+BoVW

In this section, the brief workflow of SIFT+BoVW method will be introduced.

1) Keypoints Detection: The first step is to detect interest points in the image, which are more likely to be stable and distinctive, making them suitable for matching across different images. In order to detect these interest points for the following SIFT descriptor extraction, in this preject, I tried two different methods: Difference of Gaussian (DoG) and Harris-Laplace.

In computer vision, scale invariance is one of the key requirements for feature detection. And DoG is an efficient scale-space keypoint detection method, which approximates the Laplace-Gaussian (LoG) operator by making a subtraction between different Gaussian smoothing scales and octaves, and searches for spatial-scale extreme points in the generated DoG pyramid, obtaining points of interest with invariance to scale, which in turn significantly reduces the amount of computation.

While, for the Harris–Laplace method, its central idea lies in effectively combining the spatially accurate Harris corner detector with the scale-invariant properties of the scale-normalized Laplacian-of-Gaussian (LoG) operator. This ensures that detected interest points not only have high spatial precision but are also stable across scales. Its brief workflow is as follows:

1. The Harris corner detection algorithm is applied independently to the input image at several scales (constructed via Gaussian smoothing). This step provides precise spatial localization of corner-like structures across each individual scale.
2. For each corner candidate detected from the previous step, the LoG response is computed across scales. The scale at which the candidate corner exhibits the strongest (local extrema) response is selected as its characteristic scale, ensuring robust scale invariance.

By this way, the Harris–Laplace method effectively emphasizes corner-type interest points, which typically convey richer structural information compared to edge points, while simultaneously preserving the desirable property of scale invariance inherent in LoG-based approaches. In addition, the traditional Harris algorithm is also tested in this project, and their results are compared in the Experiments section.

2) SIFT Descriptor Extraction: After gaining the interest points in Section III-B1, the project manually implements the various steps of SIFT descriptor extraction. The following will briefly describe the SIFT workflow in the project:

1. Gradient magnitude and direction are calculated at each keypoint neighborhood through a 16*16 sized region. Then, an orientation histogram with 36 bins (each covering 10°) is constructed. The peak direction from this histogram is selected as the keypoint' s main orientation, ensuring rotation invariance.
2. After aligning the neighborhood region with the keypoint' s main orientation (rotating the local patch accordingly), the region is divided into a 4×4 grid of smaller cells. Within each of these 16 cells, gradient orientations are quantized into an 8-bin histogram covering 360°. These histograms are then concatenated, forming a $4 \times 4 \times 8 = 128$-dimensional vector.
3. Applying L_2 normalization to the 128-D vector to obtain a final descriptor that is robust to make them independent of illumination and contrast variations, thus enhancing robustness.

3) Build Bag of Visual Words: After extracting SIFT descriptors (the 128-D vector) for all the training images, all the descriptors are clustered by kMeans method to get the cluster center (i.e. visual word) of each cluster. And the dimensionality of the final BoVW image coding vector is determined by the number of target clusters, presenting the expressive capacity of the BoVW.

For each image, the descriptor calls *kmeans.predict()* to assign each descriptor to the nearest visual word; then use *np.histogram* to count the number of descriptors that fall in each cluster, and obtain a one-dimensional histogram.

Finally, the histogram is normalized to convert the frequency into a probability distribution to form the final BoVW feature vector. And these BoVW vectors are utilized as feature inputs to a classifier (e.g. SVM) to perform the image classification task (target of this project).

4) Classifier: This preject test diffetent classifiers on the BoVW feature vectors, including: K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Softmax Regression. The performance of these classifiers will be compared with each other to explore the best classifier for the BoVW feature vectors.

C. CNN

CNN is selected as another approach to achieve object classification, which is more state of the art and widely used in the field of computer vision.

1) Network Architecture - Self-Designed Simple CNN: CNN Network Architecture is a key factor that affects the performance of CNN, which may influence the accuracy, speed, and robustness of the model. In this project, three different network architectures are tested:

- Self-Designed Simple CNN
- ResNet-18
- ResNet-50

All of them will implemented by self-coding, and the performance of them will be compared with each other.

The following Figure 1 is the brief architecture of the self-designed simple CNN model. It contains 2 stage, each stage contains 1 convolutional layers, 1 max pooling layer, and 1 ReLU activation layer. Then, the output of the last stage is flattened and passed to a fully connected layer. Its

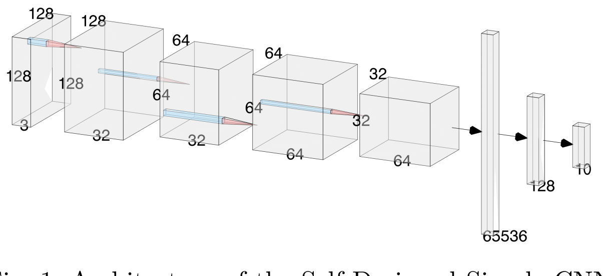


Fig. 1: Architecture of the Self-Designed Simple CNN

character is to play as the baseline model, and compare with the other two ResNet architectures (mainly ResNet-50) to explore the performance of different network.

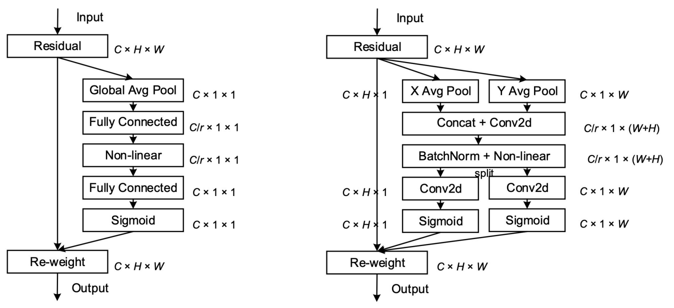
layer name	output size	18-layer	50-layer
conv1	112×112	7×7, 64, stride 2	
conv2_x	56×56	3×3 max pool, stride 2	
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax	
FLOPs		1.8×10^9	3.8×10^9

TABLE I: Architectures for ResNet (adapted from [9])

2) Network Architecture - ResNet: The Table I shows the architecture of ResNet-18 and ResNet-50, which are two popular architectures in the field of computer vision [9]. Compared with the self-designed simple CNNs, ResNets are more complex and deeper, consisting of a "stem" module followed by four sequential stages that progressively reduce spatial resolution while increasing feature depth. In addition, their most notable feature is the introduction of a residual learning mechanism to alleviate the degradation problem that occurs in the training process of deep networks.

In order to alleviate the degradation problem that arises in very deep networks, ResNets introduce skip (identity) connections in each residual block. Rather than directly learning an underlying mapping $H(x)$, each block learns a residual function $F(x) = H(x) - x$; its output is then $H(x) = F(x) + x$. This identity shortcut not only makes it easier for the optimizer to learn small perturbations around the identity (improving convergence), but also provides a direct pathway for both activations and gradients between shallow and deep layers, effectively mitigating vanishing or exploding gradients in very deep architectures.

3) Coordinate Attention: In the field of deep learning, attention mechanisms [10] have been widely used to enhance the model's performance. In this project, I also applied the Coordinate Attention (CA) mechanism [17] to the ResNets. CA is a lightweight attention mechanism designed to enhance the ability of CNNs to model spatial information while maintaining high efficiency. Unlike traditional channel attention mechanisms (e.g. SE) that focus on "channel dimensions", Coordinate Attention encodes both "spatial location information (coordinates)" and "channel dependencies". "channel dependencies". The specific schematic comparison in shown in Figure 2.



(a) The Raw Output (b) The Processed Output

Fig. 2: Schematic comparison of the proposed Coordinate Attention block (b) to the classic SE channel attention block [18] (a). (Adapted from [17])

IV. Experiments and Results

In the experimental section, this project will describe the experiments performed on each component and show the results of the experiments.

A. Local Feature Methods

This section will cover the experiments of grid search, classifiers, keypoint detection methods, and the final model selection.

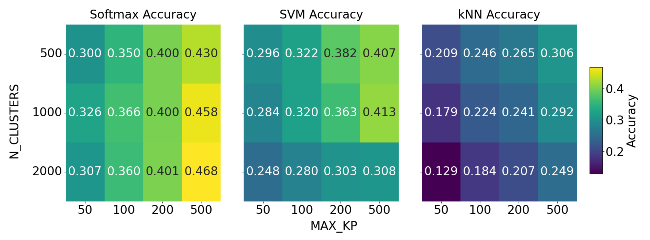


Fig. 3: Grid Search Results of SIFT on Different Classifiers

1) Grid Search: For Grid Search in the Local Feature section, this experiment mainly tests the following basic parameters:

- **MAX_KP**: The maximum number of keypoints to be detected in each image. The test range is: [50, 100, 200, 500].
- **n_CLUSTERS**: The number of clusters in the kMeans clustering algorithm, which determines the size of the BoVW dictionary. The test range is: [500, 1000, 2000].

This grid search is performed on both CIFAR-10 and iCubWorld 1.0 datasets, and the best performance is selected as the basic hyperparameters for the local feature methods. In this process, I downsampled both datasets separately to ensure efficiency in the grid search. A 30% downsampling was performed for the CIFAR-10 dataset and a 70% downsampling was performed for iCubWorld 1.0. In this case, there are 15000 images in CIFAR-10 and 4200 images in iCubWorld 1.0 being used for the grid search, which is representative. In addition, 80% of the data will be used as train set, while the rest of the data will be used as validation set which is used as the evaluation metric for the grid search.

Figure 3 shows the grid search results of SIFT+BoVW on CIFAR-10 dataset with different classifiers. It can be seen that three classifiers have different tendency with the hyperparameters. Additionally, the same phenomenon occurs on the iCubWorld 1.0 dataset. For the SVM and kNN classifiers, the accuracy of the model grows as MAX_KP grows, but this is not the case for n_CLUSTERS, where the best performance of both classifiers occurring at n_CLUSTERS=1000, and larger or smaller n_CLUSTERS both lead to a decrease in model performance. As for the softmax classifier, when MAX_KP is small, it also conforms to the above law; and when MAX_KP is large, the larger the n_CLUSTERS is, the more accurate the model is.

The possible reasons for this phenomenon could be the sensitivity of the three classifiers to the number of keypoints and BoVW size differs. First, as MAX_KP increases, the number of SIFT descriptors increases significantly, and the statistics of each visual word in the BoVW histogram is more stable, which enables both kNN based on the distance metric and SVM based on the maximal spacing to obtain more accurate and robust classification boundaries. Second, for the BoVW dictionary size n_CLUSTERS, both SVM and kNN degrade in performance when the dictionary is too small (underfitting) or too large (high-dimensional sparse/overfitting), but they reach a performance balance at a medium size (1000 words). Finally, Softmax classifier is with L2 regularization, which may help the model to suppress noisy words to take full advantage of the finer visual word segmentation. Therefore, for the comprehensive performance consideration of multiple classifiers, [MAX_KP=500, n_CLUSTERS=1000] will be chosen as the benchmark hyperparameters for Local Feature in the subsequent experiments.

2) Classifiers: Figure 3 also shows the performance difference between the three classifiers. Overall, the softmax classifier achieves the best performance, while the kNN performs the worst. And the SVM classifier is in the middle. In addition, the three classifiers differ significantly in terms of training time, with kNN being the most efficient on the same device (taking only 0.4s), softmax next (taking 1.8s for 10 epochs), and SVM (using LinearSVC) being the slowest (taking 11 mins). Therefore, for the combined consideration of model performance and training efficiency, SoftMax was used as the default classifier in all subsequent tests in this project.

3) Keypoints Detection: In the SIFT pipeline, I tested four different keypoint detection methods: *Difference of Gaussian (DoG)*, *Harris-Laplace*, *Harris*, and *LoG*. The results of these methods will be compared with each other to explore the best keypoint detection method for the SIFT pipeline.

Dataset	Harris	DoG	Harris-Laplace
CIFAR-10	45.4	40.2	46.5
CATE	23.8	21.6	25.7
DEMO	22.6	18.9	24.4

TABLE II: Accuracy (%) under different keypoint detection methods

In Table II, the experimental results demonstrate some differences in the performance of the three keypoint detection methods on different datasets. Harris-Laplace has the highest accuracy (achieves 46.5% on CIFAR-10, and 25.7% on CATE), followed by Harris (achieves 45.4% on CIFAR-10, and 23.8% on CATE), while DoG is the worst (achieves 40.2% on CIFAR-10, and 21.6% on CATE).

This performance is not expected. The performance ranking of the three was envisioned to be Harris < DoG < Harris-Laplace, but in fact Harris outperforms DoG on all test sets and closes to Harris-Laplace. Presumably, the possible reasons are, firstly, that the image resolution of CIFAR-10 is low, and thus there is limited spatial difference between different scales of blur images, making it difficult for DoG to find a truly stable blob structure; and secondly, the background of the shooting environment in the iCubWorld 1.0 dataset is relatively simple and consistent, and the surface of the object seldom shows obvious blob structures, but instead, it is dominated by edges and corner points. This makes Harris more able to obtain more stable and recognizable keypoints on both datasets, while DoG and Harris-Laplace have difficulties in obtaining better performance because of the difficulty in obtaining blobs.

Finally, Harris-Laplace was chosen as the keypoint detection method for this project in terms of overall accuracy.

B. CNN

1) Grid Search: For Grid Search in the Local Feature section, this experiment mainly tests the following basic parameters:

- **Learning Rate**: The test range is: [1e-3, 5e-4, 1e-4].
- **Batch Size**: The test range is: [32, 64, 128].
- **Epoch Number**: The test range is: [10, 20, 30].

Similarly, the two datasets used in the grid search were downsampled with the same downsampling strategy as the Local Feature grid search.

Figure 4 shows the results of the grid search on the CNN. In the CIFAR-10, the accuracy distribution is more regular basically following the law that the smaller the batch size, the larger the learning rate, the larger the epoch number, and ultimately the larger the accuracy. Therefore, [batch size=32, learning rate=1e-3, epoch number=30] is chosen as the basic hyperparameter combination for the subsequent experiments.

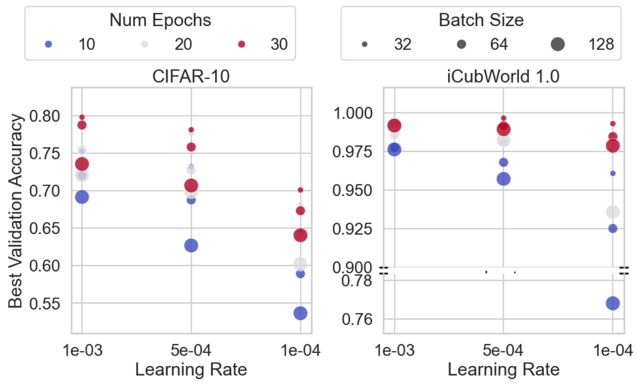


Fig. 4: Grid Search Results of CNN on CIFAR-10 and iCubWorld 1.0 (performed on Ranger optimizer)

And on the iCubWorld 1.0, all parameter combinations achieve a high level of fit, except for the case of epoch=10 where underfitting may occur. At a learning rate of 1e-3, all combinations achieve 97.5% accuracy. Also, it can be noticed that the batch size does not have a significant effect on this dataset (about 1%). Therefore, due to the concern of overfitting and comprehensive performance, [batch size=128, learning rate=5e-4, epoch number=30] is chosen as the basic hyperparameter combination on the iCubWorld 1.0 dataset.

2) Data Pre-Processing: This project applies various transformations to the original training images to synthesize training samples with higher diversity, thus expanding the data distribution and reducing the risk of model overfitting.

This experiment tests three different data augmentation strategies, each new strategy is based on the previous one for further processing as follows:

- **Strategy 1 - No Augmentation**: No data augmentation is applied to the original images.
- **Strategy 2 - Rotation and Cropping**: The images are rotated by 10 degrees and centeral cropped to 224x224 pixels.
- **Strategy 3 - ColorJitter and RandomPerspective**: The images are randomly adjusted in brightness, contrast, saturation, and hue, and the perspective is randomly changed.

Figure 5 shows the performance of the model with different data augmentation strategies on both datasets. By comparing the accuarcy line, it can be found that the model with data augmentation strategy 2 achieves the best performance (average increase of about 4.1% compared to Strategy 1); on the other hand, Strategy shows a negative impact in all conditions, with an average decrease in accuracy of 10.2% compared to Strategy 1. Therefore, the data augmentation strategy 2 is selected as the final data augmentation strategy for this project.

3) Model Architecture: As mentioned before, this project tests three different network architectures: *self-*

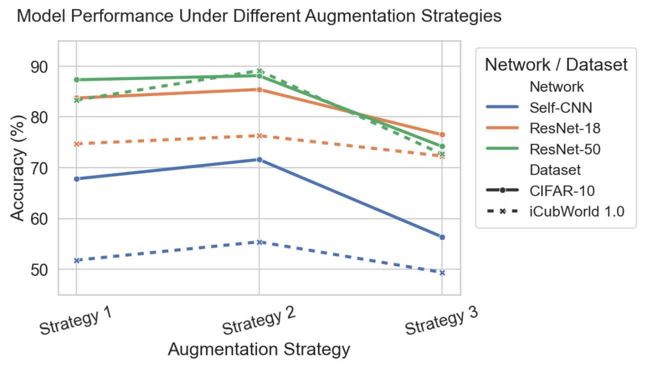


Fig. 5: Model Accuracy (%) Under Different Augmentation Strategies

designed simple CNN, ResNet-18, and ResNet-50. The following Table III shows their performance on the both datasets.

Dataset	**Self-CNN**	**ResNet-18**	**ResNet-50**
CIFAR-10	71.6	85.4	88.1
CATE	56.3	75.4	75.7
DEMO	55.4	76.3	89.1

TABLE III: Classification accuracy (%) of models on CIFAR-10 and iCubWorld 1.0 (categorization) test sets

It can be seen that on CIAFR-10, the performance of all models increases from 71% to 88% with model complexity, reflecting the positive correlation between model complexity and test set accuracy under the current hyperparameters. This phenomenon is even more pronounced for iCubWorld 1.0, where ResNet-18 generally improves accuracy by 10% compared to Self-CNN, but ResNet-50 does not improve as much compared to ResNet-18.

4) Attention Mechanism: Attention mechanisms are used as an important component of present-band machine learning, and this implementation conducts comparative experiments with several of the previously mentioned mechanisms. In Table IV, I present part of the experiment results about the attention mechanism on ResNet-50. The results show that the attention mechanism can significantly improve the performance of the model, and by comparing the training curves, it can be seen that the model with the attention mechanism tends to have a better fitting efficiency in the pre-training period. Especially when the training with smaller learning rate, the model can achieve a significant improvement of 7% on the CIFAR-10 dataset by CA. And compared with the SE and CEAM attention mechanisms, CA can achieve a better performance on both datasets. For example, on CIFAR-10, the accuracy of the model with CA is 88.1%, while the accuracy of the model with SE and CEAM are 85.4% and 86.2%, respectively. And on iCubWorld 1.0, the accuracy of the model with CA is 75.7%, while the accuracy of the model with SE and CEAM are 75.4% and 75.5%, respectively.

Dataset	**Original**	**SE**	**CA**
CIFAR-10	85.7	88.0	88.1
CATE	64.4	70.9	78.0
DEMO	81.0	84.7	89.1

TABLE IV: Accuracy (%) under different attention mechanisms

5) Optimizer: In the training process of CNN, the choice of optimizer is also a key factor that affects the performance of the model. This project mainly tests two different optimizers: *AdamW* [19], and *Ranger* [20]. By comparing the performance of the two optimizers on individual models and training sets, it was found that the final accuracy between the two was is similar (typically between 1 and 2%), but the Ranger optimizer had a better smoother fit, while the AdamW optimizer's fitting process had larger fluctuations. Therefore, the Ranger optimizer was chosen as the default option to ensure a smoother training process and facilitate the tuning of other parameters.

V. Discussion

This section will discuss the difference between the two methods from different perspectives, including the model performance, training time, and the interpretability of the model.

Overall, in terms of performance, the accuracy of the ResNet-50 model (with CA module) is significantly better than the Local Feature (SIFT+BoVW) method on either test set. This proves that the CNN model can still show good generalization performance on the image classification task, even with a smaller data volume and a test set that requires higher generalization ability of the model, such as iCubWorld 1.0. As for the SIFT method, its accuracy is backward, and it can only achieve an accuracy of 46.5% even on a simple dataset like CIFAR-10 with sufficient sample size, which illustrates the limitation of the SIFT method on the image classification task.

However, the performance comparison still reveals the essential difference in generalization performance between the two methods due to the different underlying logic. On the iCubWorld 1.0 dataset, the accuracy of SIFT+BoVW on CATE is higher than that on DEMO, while the accuracy of ResNet-50 is higher on DEMO than on CATE, which suggests that the SIFT+BoVW method relies more on low-level features such as color and texture of the image for feature extraction, and it can classify the same kind of different images in the CATE test set by learning the "local structure" of the image, and the model performs better when facing the same category of different instances in the CATE test set; however, in the DEMO test set, the performance of the model is degraded due to the drastic change of the background information by the change of the demonstrator. ResNet-50, on the other hand, achieves object classification by learning abstract semantic features of the image, which makes it more robust to changing backgrounds; however, since the CNN model is a Data-Driven approach, it is more dependent on sample size, which causes it to perform more poorly when confronted with small sample sizes of unseen examples in the CATE test set.

There is also a significant difference in the computational cost between the two methods. the computational time of the SIFT+BoVW method is mainly focused on feature extraction and BoVW dictionary construction, and the number of images that can be processed per second ranges from about 5it/s to 30it/s (using CPU only) after limiting the number of available keypoints. In contrast, the computational time of ResNet-50 (with a reference number of about 2.55×10^6) is mainly focused on model training, with its training time on the two datasets being 1.5 hours and 1 hour respectively (under the condition of using GPU acceleration). This reflects the huge difference in computation between the two and explains why the Local Feature approach has always been active on edge devices and has not been replaced by CNNs. Also the interpretability of the SIFT+BoVW method is significantly better than that of ResNet-50.Since the SIFT+BoVW method classifies images based on their local features, its classification results can be explained by visualizing the features in the BoVW dictionary; whereas, the training process of ResNet-50 is a black-box process, which makes it difficult to explain the reason. Therefore, the SIFT+BoVW method is easier to analyze and debug, while ResNet-50 can only rely on the loss function during the training process to evaluate the performance of the model.

VI. Conclusion

To summarize, this project has successfully implemented two different image classification methods: Local Feature (SIFT+BoVW) and CNN (ResNet-50). The experimental results show that the CNN model outperforms the Local Feature method in terms of accuracy and generalization performance. The project also explores the impact of various hyperparameters, data augmentation strategies, and model architectures on the performance of both methods.

Although this project has achieved good results, there are still some limitations and future work that can be done. For example, for the hyperparameter tuning, this project only performed a grid search on a few basic hyperparameters, and more automatical tuning methods such as Optuna can be used to find the optimal hyperparameters with more detailed search space. In addition, the techniques mentioned in the Section II, e.g. Vision Transformer, can be applied to further improve the performance of the model.

The practice and discovery of this project has deepened my understanding and inspired my interest in the field, and I look forward to further continuous improvement of the program and further study of the field afterwards.

References

[1] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images.(2009),” 2009.

[2] S. Fanello, C. Ciliberto, M. Santoro, L. Natale, G. Metta, L. Rosasco, and F. Odone, “icub world: Friendly robots help building good vision data-sets,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2013, pp. 700–705.

[3] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. Ieee, 1999, pp. 1150–1157.

[4] ——, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (surf),” *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[6] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[7] N. Robinson, B. Tidd, D. Campbell, D. Kulić, and P. Corke, “Robotic vision for human-robot interaction and collaboration: A survey and systematic review,” *ACM Transactions on Human-Robot Interaction*, vol. 12, no. 1, pp. 1–66, 2023.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[11] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, “Transformers in vision: A survey,” *ACM computing surveys (CSUR)*, vol. 54, no. 10s, pp. 1–41, 2022.

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[13] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[14] J. Zhu, X. Chen, K. He, Y. LeCun, and Z. Liu, “Transformers without normalization,” *arXiv preprint arXiv:2503.10622*, 2025.

[15] Q. Li, Y. W. Teh, and R. Pascanu, “Noprop: Training neural networks without back-propagation or forward-propagation,” *arXiv preprint arXiv:2503.24322*, 2025.

[16] K. Sargent, K. Hsu, J. Johnson, L. Fei-Fei, and J. Wu, “Flow to the mode: Mode-seeking diffusion autoencoders for state-of-the-art image tokenization,” *arXiv preprint arXiv:2503.11056*, 2025.

[17] Q. Hou, D. Zhou, and J. Feng, “Coordinate attention for efficient mobile network design,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13 713–13 722.

[18] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[19] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” *arXiv preprint arXiv:1711.05101*, 2017.

[20] L. Wright, “Ranger - a synergistic optimizer.” https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer, 2019.

Appendix