Assignment3: Report

Ronak Dedhiya Dinesh (ronakdedhiya@iisc.ac.in; SR No: 06-18-05-19-52-21-1-20116)
Prashanth N Bhat(prashanthbn@iisc.ac.in; SR No: 04-03-06-19-52-21-1-20082)

September 28, 2024

1 Problem Statement

Self Supervised Learning

1. Consider the CIFAR-10 dataset CIFAR-10. Construct a CNN-based classifier (call it the base-model) and report the accuracy. Now change the definition of the classes in two different ways (one binary classification problem and one 5-class classification problem). Use the pretrained base-model with an additional classification head at the last layer in accordance to the new class definitions. Retrain only the final classification heads and report the accuracies. Plot the t-SNE graphs for all three cases and record your observations.

2. Implement MOCO for CIFAR-10. Consider 5 different types of augmentations (rotations, blur, color distortion, cropping and resizing) for defining the positive samples. Use the entire training data to learn the representations. Once the representations are learned, use a linear and logistic layers and retrain with 10-50% of supervised training data and compare the results with the base CNN model in question (1). Experiment with two different sizes for the encoder dictionary.

2 CNN based classifier

2.1 Dataset Preparation

The CIFAR-10 dataset has 10 classes: Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship, and Truck. In addition to the original labels of the dataset, two more labels are added:

- **Binary classification** For Binary classification, assign 0 to Vehicle and 1 to Animal
- **5 class classification** For 5 class classification, define the 5 classes as below
    1. Airplane or Bird
    2. Automobile or Truck
    3. Ship or Frog
    4. Cat or Dog
    5. Deer or Horse

2.2 Model architecture

The base CNN model architecture used is depicted in Figure 1. Figures 2 and 3 use the base model in 'model' and a classification head.

2.3 Results

The accuracy results with Training and Test data are as in Figure 4

2.4 T-SNE plots

The T-SNE plots were created using T-SNE library from sklearn.manifold. The T-SNE plots are as in Figures 5, 6, and 7.

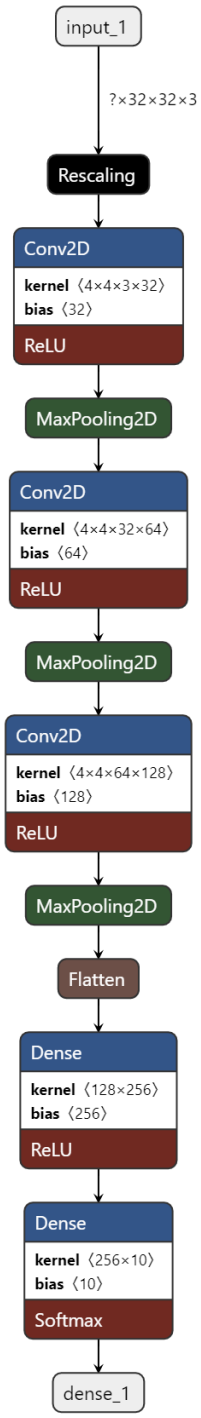

Figure 1: Base CNN classifier

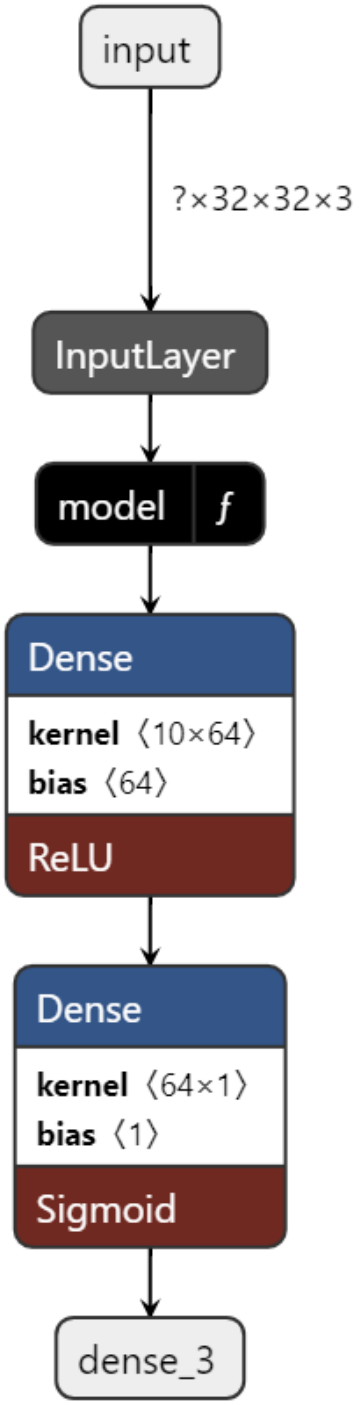

Figure 2: CNN classifier for binary

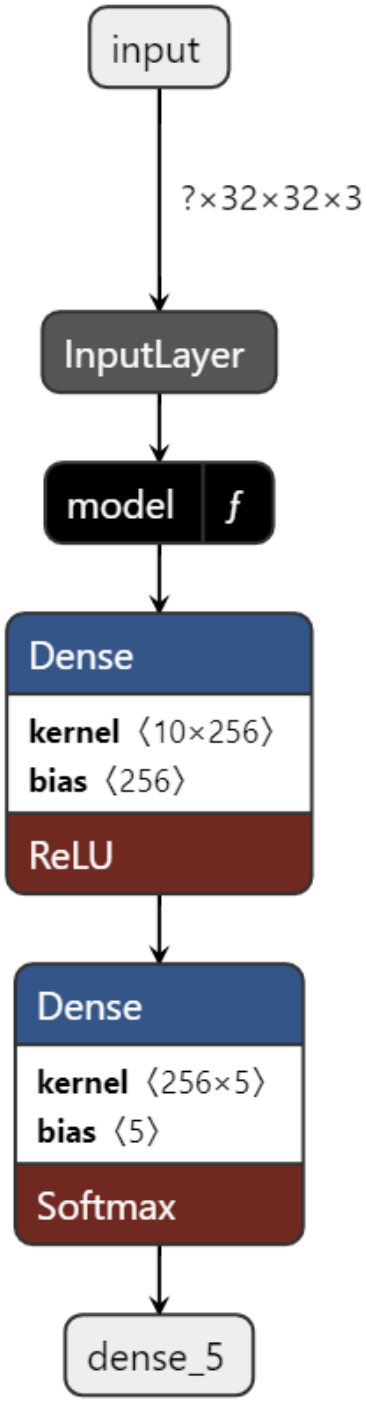

Figure 3: T-SNE plot for 5-class classification data prediction

	Accuracy (%)	
	Training	Testing
Base CIFAR-10	84.27	67.19
Binary Classifier	98.18	67.19
5 Class Classifier	92.19	75.29

Figure 4: CNN results

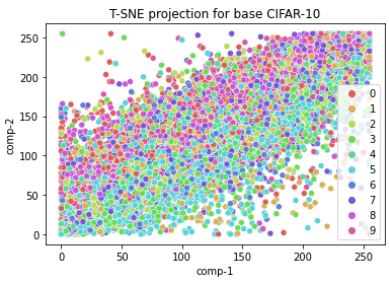

Figure 5: T-SNE plot for base CIFAR-10 data prediction

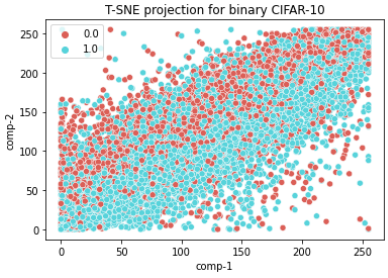

Figure 6: T-SNE plot for binary classification data prediction

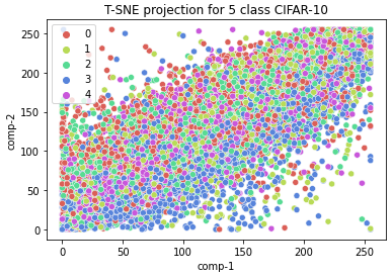

Figure 7: T-SNE plot for 5-class classification data prediction

3 MOCO

3.1 Description

- Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys (k0, k1, k2, ...) are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the current mini-batch enqueued and the oldest mini-batch dequeued, decoupling it from the mini-batch size. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations.

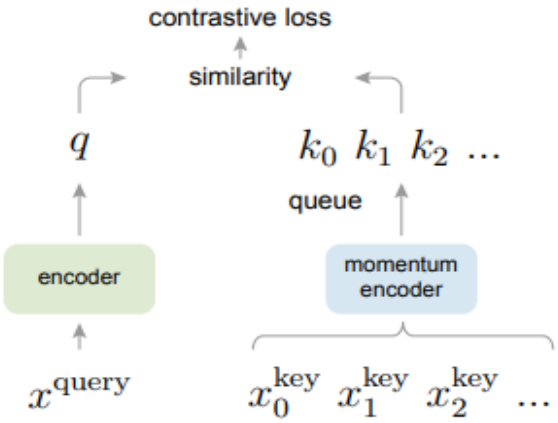

Figure 8: Momentum Contrast Methodology

3.2 Loss

- Similarity is measured using the dot product, a form of contrastive loss function, called InfoNCE given by:

$$L_q = -log\frac{exp(q.k_+/t)}{\sum_{i=0}^{K} exp(q.k_i/t)} \tag{1}$$

- Failure occurs due to rapidly changing encoder affecting key representations consistency. The paper proposes a momentum update to address this issue which is given by:

$$\theta_k = m\theta_k + (1-m)\theta_q \tag{2}$$

3.3 Results

Experimentation with Embedding size and data split ratio for classifier training

We experimented with two embedding sizes, 128 and 512. It is observed that self-supervised training for 512 takes longer to converge than with 128. We also experimented with the classifier training with different data ratios, i.e., training the classifier with only 50%, 30%, and 10% data, where we observed that the lesser the data, the more training accuracy deters. The accuracy on cifar 10 test set for different configurations is shown in Fig 12, we see that embedding size 128 performs better than 512 and testing accuracy drops as the supervised data is reduced from 50% to 30% to 10%. Max accuracy obtained is 44.3%

Experimentation with momentum parameter

Here we change the momentum parameter, which decides the exponential averaging of the weights between the models. The paper's optimal value is 0.99; we experiment with the same and 0.999,0.9,0.95, as shown in the figure below. We observe that performance sharply drops when a momentum parameter of 0.9,0.95 is used; the model hardly learns anything. Performance with 0.999 is similar to 0.99 but still not better. On the test set, we obtain only 37% accuracy with 0.999 and around 10% for 0.9,0.95. We can infer that momentum is an important hyperparameter, and 0.99 is the optimal value.

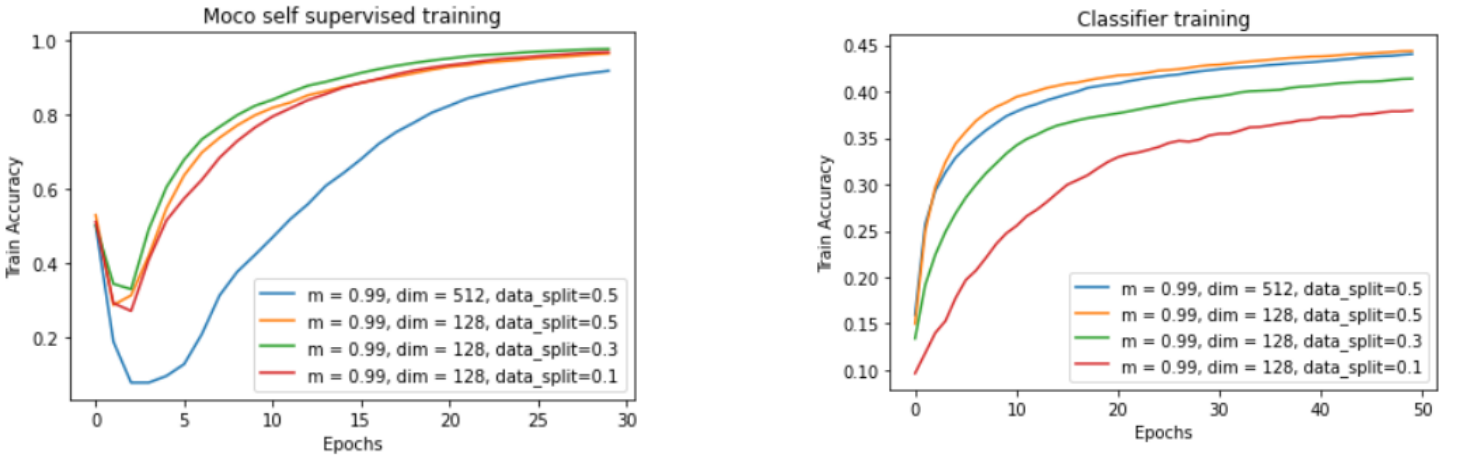

Figure 9: Effect of Embedding size and available classifier training data

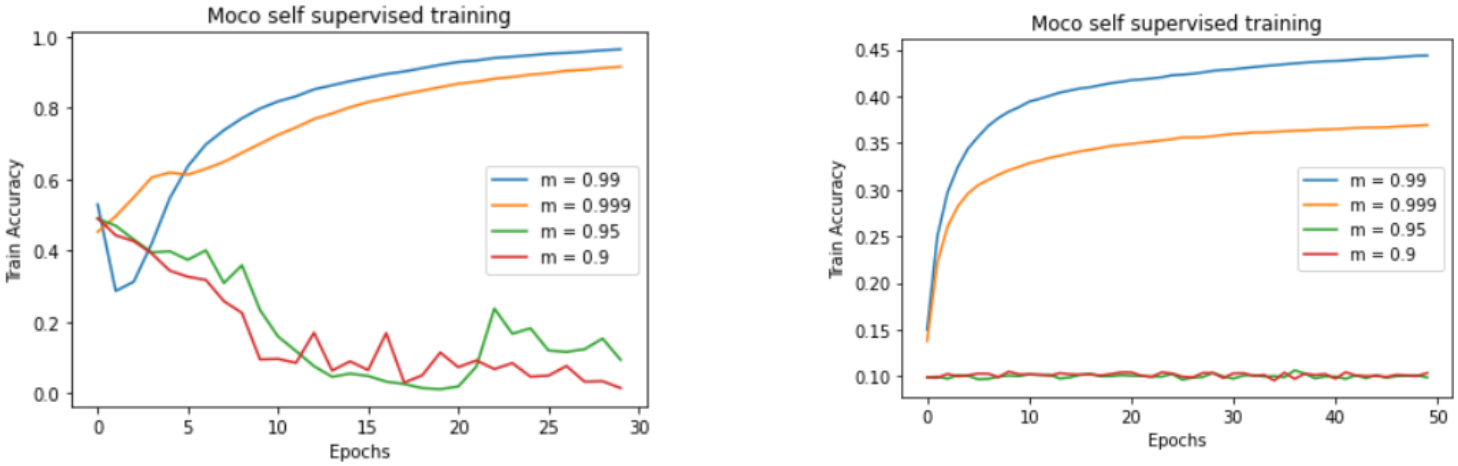

Figure 10: Effect of momentum parameter

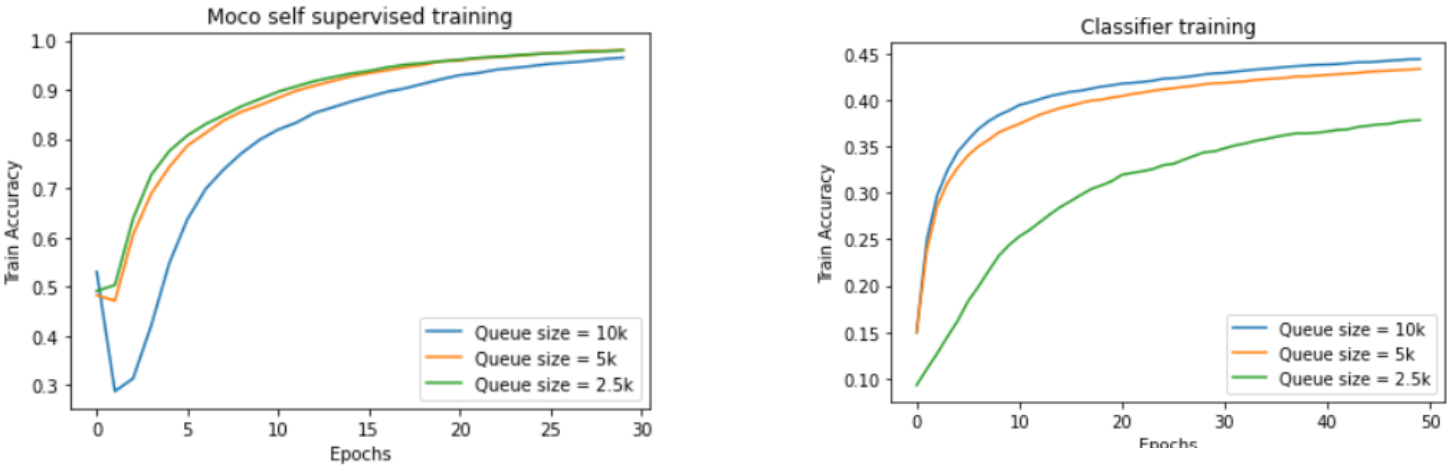

Figure 11: Effect of queue size

Effect of queue size

We change the queue size and selected from 10000,5000,2500. We saw almost similar performance with 10000 and 5000, and slight decrease in accuracy with 2500 queue size as shown in Fig 12

Final result

The final best result from the table 12 infers that queue size = 10000, momentum = 0.99, embedding size = 128, available data = 0.5 results in the best accuracy on the cifar 10 test dataset of 44.3%.

Queue size	momentum	Emb_size	datasplit ratio	Test set accuracy
10000	0.990	512	0.5	0.422903
10000	0.990	218	0.5	0.443771
10000	0.990	218	0.3	0.409334
10000	0.990	218	0.1	0.369860
10000	0.999	218	0.5	0.372327
10000	0.950	218	0.5	0.099404
10000	0.900	218	0.5	0.100637
5000	0.990	218	0.5	0.439453
2500	0.990	218	0.5	0.369038

Figure 12: Result of Momentum contrast with various hyperparameters