EECS569 Hw5

Bumsik Kim

April 16, 2023

1 CNN Training on LeNet-5

1.1 Motivation

Nowadays, deep learning and AI are the trending topics that will lead the future of technology. In this problem, we will learn to train a simple convolutional neural network (CNN) called LeNet-5 and apply it to three datasets MNIST, Fashion-MNIST, and CIFAR-10.

1.2 Approaches

1.2.1 LeNet-5

LeNet-5 is designed for handwritten and machine-printed character recognition. We have implemented the LeNet-5 using the neural net library from Pytorch based on the explanation of structures in the assignment documentation. The only variation I have applied is that on the number of padding in first convolutional layer and its input channel. It is because CIFAR-10 has 3 channels unlike MNIST and Fahsion-MNIST and it also has a different image size. When training MNIST and Fashion MNIST, we use the padding of the first convolutional layer as 2 and the input channel as 1. But for the CIFAR-10, we use an input channel of 3 and the number of padding to 0.

1.2.2 Training

When constructing training and testing data, we have applied transformation on every dataset using normalization that works for each particular dataset. We have set the training batch size to 64 and the testing batch size to 1000. For every training and testing, we used a fixed number of epochs, which is 10.

1.3 Result

1.3.1 Problem 1-(b)-1,2

We have conducted training on the LeNet5 with MNIST data under three different yet representative hyper-parameter settings. The three hyper-parameters

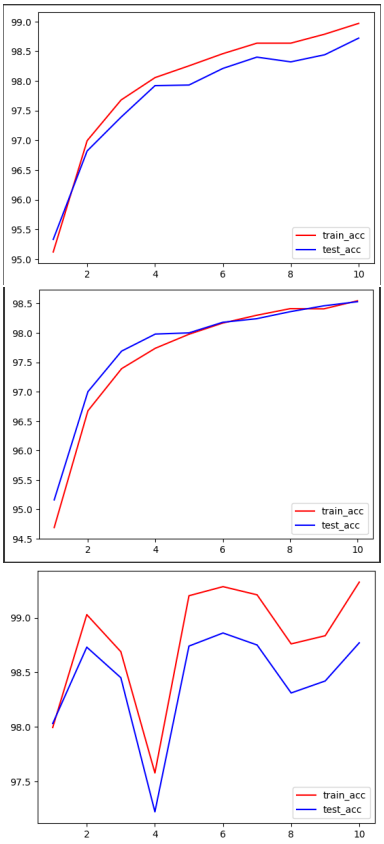
we have selected are learning rates, gamma(decay of the learning rate), and optimizer. The below table is the details of each setting.

Setting No.	Optimizer	Learning Rates	Gamma(Decay)
1	SGD	0.001	0
2	SGD	0.001	0.9
3	Adam	0.01	0

For each setting, we ran 5 runs to report the best test accuracy among five runs, the mean test accuracy of five runs, and the standard deviation of test accuracy among five runs to evaluate the performance. The below table is the result of runs for each setting. The training and testing were held under images of MNIST.

Setting No.	Best Test Accuracy	Mean Test Accuracy	Standard Deviation of Test Accuracy
1	98.720	98.662	0.059
2	98.530	98.402	0.120
3	98.770	98.622	0.184

The below figures are the plot of train accuracy and test accuracy of each epoch that holds the best test accuracy with setting No.1, 2, and 3.



1.3.2 Problem 1-(b)-3, 4

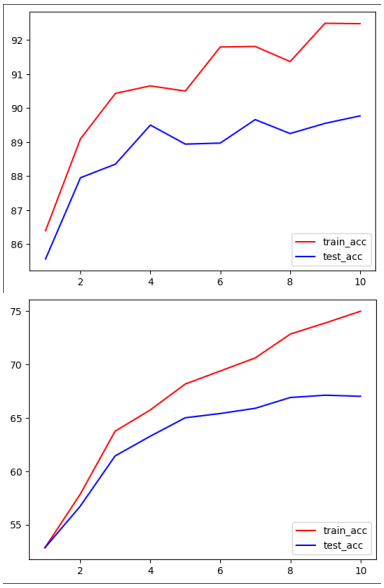
The below table lists details of the best parameters we obtained for MNIST, Fashion-MNIST, and CIFAR-10, respectively.

Image Data	Optimizer	Learning Rates	Gamma(Decay)
MNIST	Adam	0.01	0
Fashion-MNIST	Adam	0.01	0
CIFAR-10	SGD	0.01	0.9

We run training and testing five times and report the performance on Fashion-MNIST and CIFAR-10 image datasets, respectively.

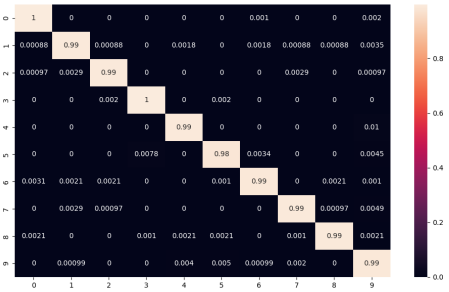
Image Data	Best Test Accuracy	Mean Test Accuracy	Standard Deviation of Test Accuracy
MNIST	98.770	98.622	0.184
Fashion-MNIST	89.770	89.332	0.290
CIFAR-10	67.000	66.534	0.345

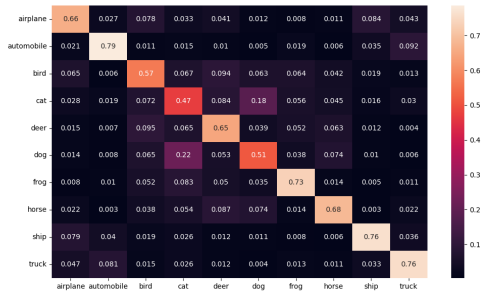
The below figures are the plot of train accuracy and test accuracy of each epoch that holds the best test accuracy for Fashion-MNIST and CIFAR-10, respectively.



1.3.3 Problem 1-(c)-1

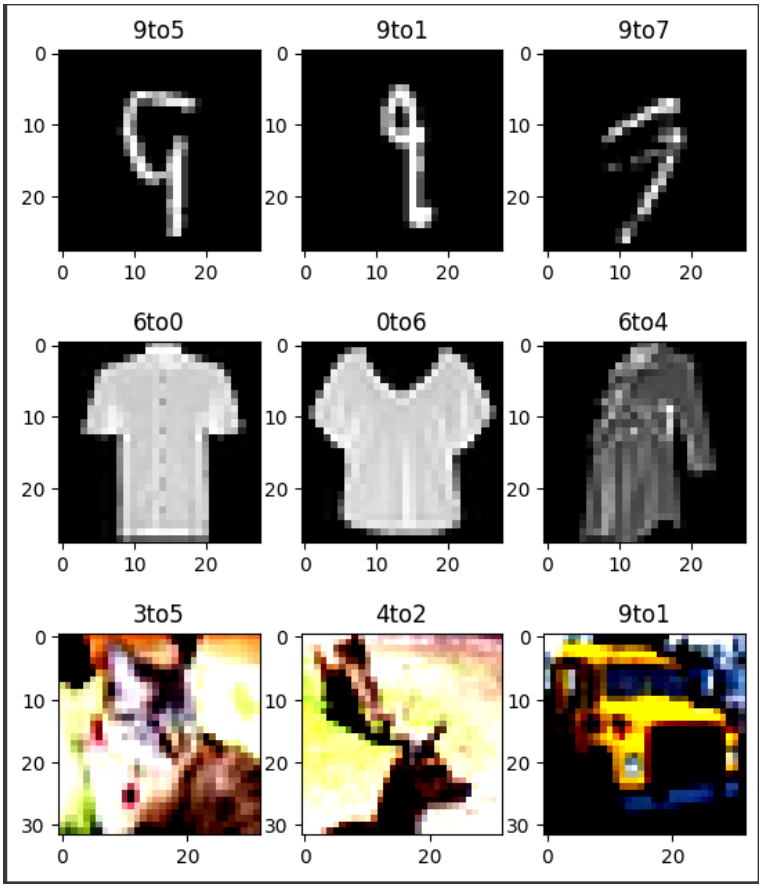
The below figures are the normalized confusion matrix for the ten classes on the MNIST, Fashion-MNIST, and CIFAR-10, respectively. The x-axis of the plot tells the expected labels from the test images and the y-axis of the plot tells the predicted labels from the test images using LeNet5.





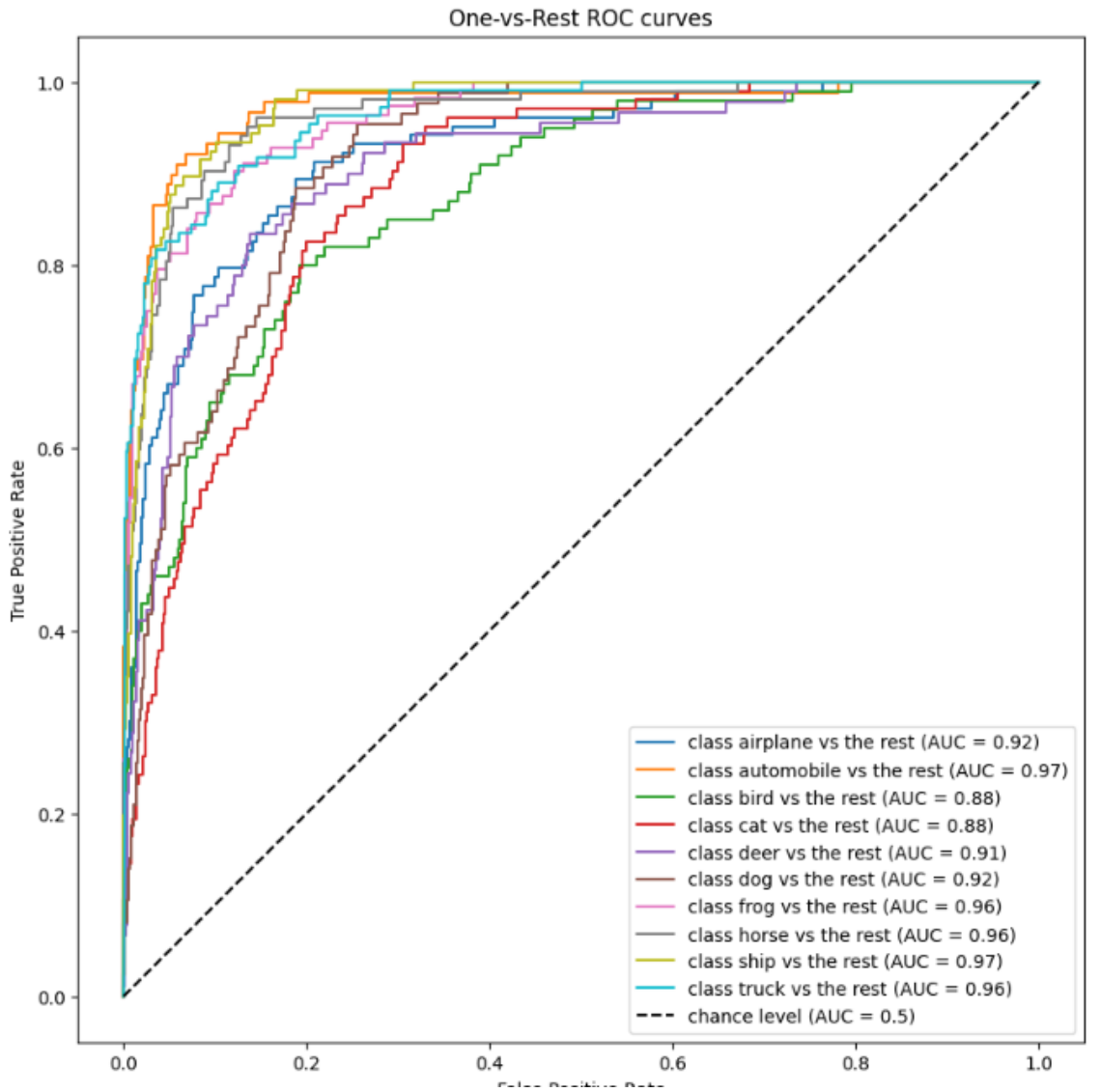
The below plot is the demonstration of examples of the top three confused pairs for MNIST, Fashion-MNIST, and CIFAR-10, respectively.

Each title is written in a form of "9to5". For instance, if the following title for an example is "9to5", it means the true label for the corresponding image is "9" but the prediction from LeNet5 under our best parameter is "5".



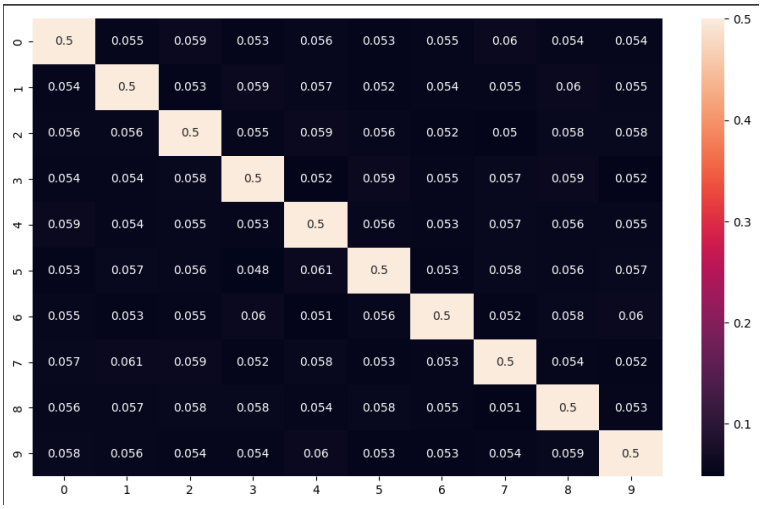
1.3.4 Problem 1-(c)-2

The below plot is the ROC curve for each class for CIFAR-10 in a one-vs-rest manner. The x-axis of the plot represents the false-positive rate and the y-axis of the plot represents the true positive rate.



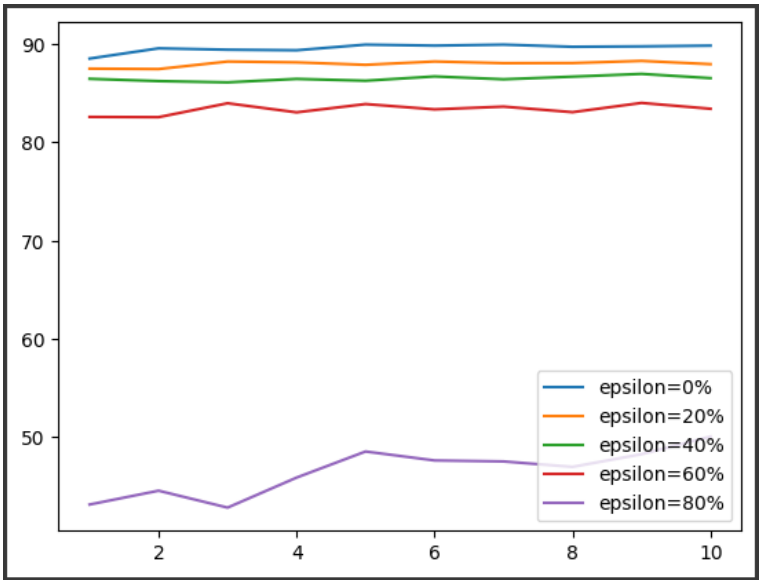
1.3.5 Problem 1-(d)-1

The below plot is the normalized confusion matrix for epsilon = 50% between true labels and noisy labels of the Fashion-MNIST training set.



1.3.6 Problem 1-(d)-2

The below plot is the curve of testing accuracy versus epsilons for 0%, 20%, 40%, 60%, and 80%. The x-axis represents the epoch and the y-axis represents the mean test accuracy for each epoch under 5 runs.



1.3.7 Problem 1-(d)-3

The below chart shows the mean and standard deviation of the testing accuracy under five runs for each setting.

Epsilon	Mean Testing accuracy	Standard deviation Testing Accuracy
0%	89.589	0.404
20%	87.975	0.277
40%	86.478	0.241
60%	83.352	0.509
80%	46.552	2.2667

1.4 Discussion

1.4.1 Problem 1-(a) CNN Architecture

Question 1.

In this part, we will describe the definition of some components that constitute CNN.

The fully connected layer is a layer that applies a linear transformation to a given input tensor using a weight matrix. The fully connected layer connects every input neuron to every output neuron.

The convolutional layer is the main building component of a CNN. It contains a set of filters. The filter includes parameters that are to be learned throughout the training. A convolutional layer transforms the input image in order to extract features from it by convolving with a filter.

The max-pooling layer applies a pooling operation that calculates the maximum value in each patch of each feature map. The results are pooled feature maps that highlight the most present feature in the patch. The main purpose of pooling is to reduce the size of feature maps which helps increases computation speed.

The activation function applies nonlinearity that allows the neural networks to develop complex representations and functions. Examples of activation functions would be ReLu, Leaky ReLu, and so on.

The softmax function is an activation function in the final layer of a neural network to classify multi-class data. It returns the vector that has the same length as the number of classes in the data. Each element in the vector represents the probabilities of the input data to be classified as the index of itself. The probabilities of the categories must sum up to 1.

Question 2. Overfitting makes the model relevant to its training set only. It may have a high accuracy on the training set, but the overfitted model will have a low accuracy on the testing set. Cross-validation is one of the methods that help to avoid overfitting. Cross-validation is a technique for evaluating the model by training several models on subsets of the available input data and evaluating them on the complementary subset of the data. This helps to detect whether the model is overfitted or not.

Question 3. ReLU, Leaky ReLU, and ELU are non-linear activation functions these are the difference between them.

ReLU is a function that returns the input itself if the input is bigger than 0, 0 otherwise. The advantage of using ReLU is that it is easy to compute compared to other activation functions.

Leaky ReLU is a function that returns the bigger value between a*x and x, where x is the input and a is a constant smaller than 1. This allows a small, non-zero output for negative inputs and the gradient to flow for negative inputs.

ELU is a function that is defined as below.

$$\mathrm{ELU}(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha * (\exp(x) - 1), & \text{if } x \leq 0 \end{cases}$$

From the function definition above, we could infer that ELU handles the negative input by transiting into an exponential function. ELU has a similar positive effect as Leaky ReLU has compared to ReLU where it allows a small, non-zero output for negative inputs and the gradients to flow for negative inputs. Furthermore, using ELU as an activation function will converge faster and have better generalization performance compared to using ReLU and Leaky ReLU.

Question 4.

L1 loss is used when we need to predict a continuous target variable like the regression model. Predicting the price of the apple based on the weather change may be one example to use L1 loss. L1 loss measures the absolute difference between predicted and ground truth.

MSE loss is also used for a regression problem which is similar to L1 loss. We can both apply L1 loss and MSE loss at the same regression problem. While L1 loss uses the absolute difference, MSE loss uses square loss between prediction and ground truth. Using MSE Loss will allow the model to be penalized harder for smaller errors compared to using L1 Loss when training.

Other than L1 and MSE losses, Binary Cross Entropy(BCE) Loss is used in binary classification problems. Binary cross entropy is a metric that classifies the data into two classes. Spam detection is one example that uses BCE loss to define whether the email is spam or not.

1.4.2 Problem 1-(b) Compare classification performance on different datasets

Question 1. From the above tables and plots in Section 1.3.1, we could observe that under the SGD optimizer, when applying decay on learning rates, the mean test accuracy may decrease but can obtain more variations of test accuracy with each run. Since the setting with the Adam optimizer got the highest test accuracy, we could observe that the Adam optimizer works slightly better compared to the SGD optimizer. With a higher learning rate, we could obtain better test accuracy. The best test accuracies for three settings placed between 98-99%, we could assume that those three settings are all appropriate hyper-parameter settings for LeNet5 with MNIST image data.

Question 2. The best parameter setting to achieve the highest accuracy on the MNIST test set is setting No.3, which is Adam optimizer, with a learning rate of 0.01, and a gamma value of 0(no decay). The performance plot is in Section 1.3.1, the last plot among the plots in that section. The best test accuracy is 98.770, the mean test accuracy is 98.622, and the standard deviation of test accuracy is 0.184.

Question 3. The best parameter we obtained from Section 1.3.1 was Adam Optimizer with a learning rate of 0.01 and gamma value of 0. Using this parameter, as we did above, we run training and testing five times and report the performance on Fashion-MNIST. The best test accuracy is 89.770, the mean test accuracy is 89.332, and the standard deviation of test accuracy is 0.290.

Question 4. The best parameter we obtained from Section 1.3.1 was SGD Optimizer with a learning rate of 0.01 and gamma value of 0.9. Using this parameter, as we did above, we run training and testing five times and report the performance on CIFAR-10. The best test accuracy is 67.000, the mean test accuracy is 66.534, and the standard deviation of test accuracy is 0.345.

Question 5. From the performance results, we could observe that LeNet5 performs the best on MINST image data, followed by Fashion-MNIST and CIFAR-10 in order. The difference in terms of best test accuracy can be clearly observed when running the training and testing among three datasets, MNIST, Fashion-MNIST, and CIFAR-10. The test accuracy using CIFAR-10 was significantly low compared to MNIST and Fashion-MNIST. It is because images from CIFAR-10 have more variation compared to ones from MNIST and Fashion-MNIST. In terms of variation, each image from CIFAR-10 has various scales, orientations, colors, backgrounds, and so on. However, images from MNIST and Fashion-MNIST only have an object inside the image. Furthermore, we have trained and tested on a selected number of epochs, which is 10, for every dataset. Since the CIFAR-10 has more complicated image data compared to others, the test accuracy may enhance when having more epochs.

1.4.3 Problem 1-(c) Evaluation and Ablation Study

Question 1. Based on the normalized confusion matrices in Section 1.3.3, we could infer the top three confused pairs of classes for each dataset by selecting the three pairs of expected labels and (mis)predicted labels from the confusion matrix that has the highest score. For MNIST, we have selected (9,5), (9,1), and (9,7) as the top three confused pairs of classes. For Fashion-MNIST, we have selected (6,0), (0,6), and (6,4) as the top three confused pairs of classes, where 0 represents T-shirts/tops, 4 represents Coats, and 6 represents Shirts. For CIFAR-10, we have selected (3,5), (4,2), and (9,1) as the top three confused pairs of classes, where 1 represents automobiles, 2 represents birds, 3 represents cat, 4 represents deer, 5 represents a dog, and 9 represents trucks. For the explanation of the representation of confused pairs, the first element of each pair is the true label and the second element of each pair is the predicted label.

We could observe that confusion has generated from the network when the image looks similar to the images from the mispredicted label. However, this kind of confusion would not happen if the human is the one who classifies the image. I would assume this happens due to the limitation of the neural nets.

Question 2. From the plot in Section 1.3.4, we could observe the ROC curves for each class in a one-vs-rest manner. The ROC curves are computed based on a confidence score for each class, which is generated by the final softmax function. We could infer that the class with high accuracy and AUC score corresponds to the curve that is structured in a way that has a high area under the ROC curve.

Question 3.

The overall AUC score in a multiclass classification problem is typically calculated as a weighted average of the ROC AUC scores for each class. Since CIFAR-10 is a multi-class image dataset, we used micro-average when computing the overall AUC score, which is 0.94.

1.4.4 Problem 1-(d) Classification with noisy data

Question 1 In this section, we will describe the implementation details of Symmetric Label Noise. Symmetric Label Noise (SLN) is the type of labeling noise where a% of the data with the true label of class I is labeled as other classes J that is not the same as class I with uniform probability. For the implementation, we first count the value distribution of labels. For Fashion-MNIST, the total number of image data is 60000 and there exist 6000 images of labels 0 to 9, respectively. Using the given epsilon, we could obtain the number of images that will be label-noised. For epsilon = 50 %, 3000 images per label will be noised. To noise the label, we iterate the image label and changed the label with a uniformly random label that is not the same as the original one. The result after applying the Symmetric Label Noise is in section 1.3.5.

Question 2

The curve of testing accuracy versus epsilons for 0%, 20%, 40%, 60%, and 80% is in section 1.3.6. For each epsilon, five runs have been run to calculate the mean and standard deviation of the testing accuracy.

Question 3

From the curves in section 1.3.6, we could infer that the testing accuracy decreases as the value of epsilon grows. The table that represents the mean and standard deviation of the testing accuracy is in section 1.3.7. Based on the results and the curves, we could infer that when the epsilon value is at 80%, the testing accuracy has decreased dramatically. Furthermore, the standard deviation is also high compared to other results with different epsilon values. It is because testing accuracy was extremely low when it trains for the first epoch.