Fall 2020

INTEL Dataset Classification by Neural Networks

EE285 Project 1

Rojin Zandi
Partner: Shifa Shaikh

1. Dataset Description

Intel image dataset contains three files: training, test and prediction. In this project we are going to use the training and test set. Each one of the sets contains six different files which are the labels(classes) for the images. These labels are Buildings, Forest, Glacier, Mountain, Street, and Sea.

	Buildings	**Forest**	**Glacier**	**Mountain**	**Sea**	**Street**	**Total**
Training Set	2191	2271	2404	2512	2274	2382	14034
Test Set	437	474	553	525	510	501	3000

Table 1: Number of images in each class of the training and test set.

As it can be seen in Fig.1 and Fig.2, Mountain and Buildings classes has the most and least number of training data, respectively. On the test set, Glacier with 553 images is the largest and Buildings with 437 images is the smallest class of data [Table 1].

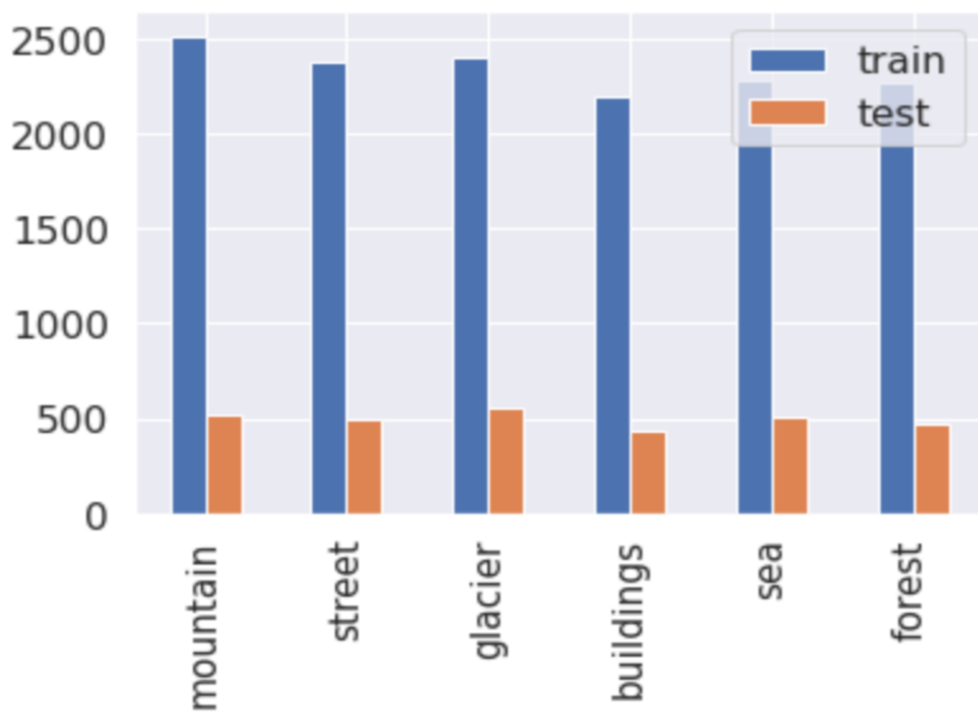

Fig. 1: Bar plot of the number of images in each class of the training and test set.

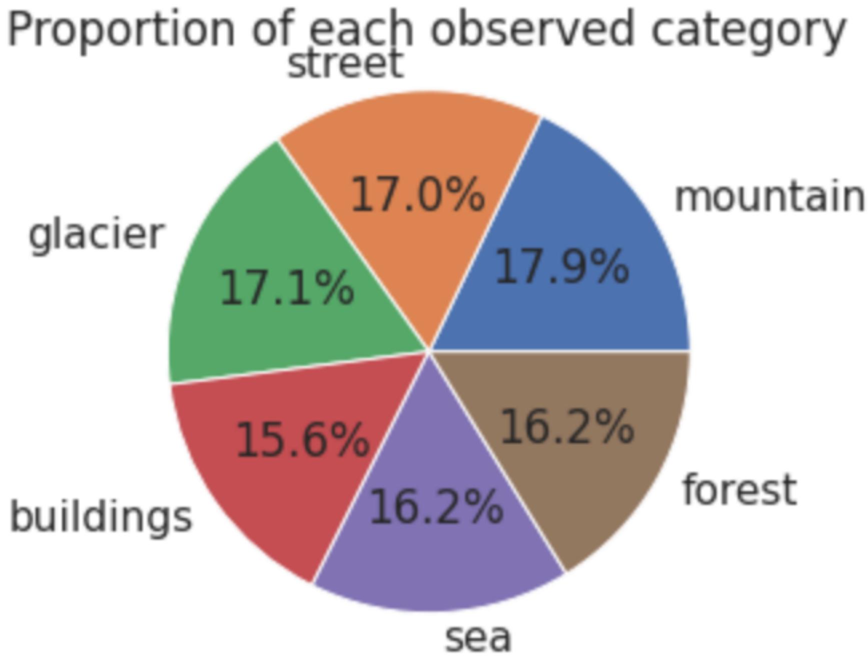

Fig. 2: Pie chart of the proportion of each observed category in the training set

2. Preprocessing

In Machine Learning, we need to provide clean and suitable data for feeding the algorithm. Preprocessing in deep learning is the process of transforming the raw data to a proper type which improves the accuracy of our models. Preprocessing includes data cleaning, transformation, reduction, integration which depends on our data and application. For example, in Intel image dataset we were successful to use all the data and data reduction was not necessary, but the type of data needed to be transformed. The raw data is RGB image which is not suitable for feeding our neural network models. So, we have implemented a function in our code, which takes some information about the data, such as data path, images, name of classes (labels), size of images. Notice that this function uses OpenCV library which the RGB images in this package are read BGR, so we need to change it. Finally, after resizing and calling the data, we have changed the type of images to `float32` and labels to `int32`. After implementing the function, we can load the data from our drive and start data analysis.

3. Models Utilized

In this project, we have implemented 11 Feed Forward Neural Networks (FFNN) without any backpropagation. In each model we have changed one parameter which affect the efficiency and may decrease or increase model accuracy and loss. The training and test accuracy of all utilized models are provided in Table 2.
In the provided code, we have used Sequential API which allows us to create our models layer by layer, but we only can use it when the model is not sharing layers. Considering our data which is image, we need to use Flatten API to convert our input to a one-dimensional array. There is also Dense layer in our code, which connects every input to every output. In other words, it fully connects the neurons of the previous layer to the neurons of the next layer.

	# Layers	Optim. Method	Learning Rate	# Epochs	Activation function	Regul.	Val. Split	Training Accuracy	Test Accuracy
1	3	SGD	0.1	10	same	No	0.1	66.72%	47.83%
1.2	4	SGD	0.1	10	same	No	0.1	65.86%	53.53%
2	3	SGD	0.001	10	same	No	0.1	29.65%	28.2%
2.1	3	Adam	0.001	10	same	No	0.1	68.7%	57.56%
3	4	Adam	0.001	10	same	No	0.2	67.92%	58.93%
4	4	Adam	0.001	10	Tanh*	No	0.2	67.28%	57.46%
5	4	Adam	0.001	15	same	No	0.2	72.86%	60.93%
6	4	Adam	0.001	15	same	L1	0.2	21.89%	17.49%
6.1	4	Adam	0.001	15	same	L2	0.2	33.65%	33.43%
6.2	4	Adam	0.001	15	same	L1.L2	0.2	31.43%	22.83%
6.3	4	Adam	0.001	15	same	Dropout	0.2	69.72%	58.17%

Table 2: Parameters and accuracy of utilized models

4. Model Improvements

As discussed in section 3, changing the parameters such as number of layers or learning, can affect the accuracy and loss of the model. To compare the efficiency of different models, we have provided the test and training accuracies in Table 2 and also confusion matrix of each model (Fig. 3 to Fig. 13). We have used heat map which shows the highest values by lighter colors, so in our figures we prefer to have a lighter diagonal which means there are more correctly predicted data.

- **Model 1**

  In model 1, we have three layers which are two hidden layers, and output with 512, 128, and 6 neurons, respectively. The activation function in hidden layers is Rectified Linear Unit (ReLU) and in output SoftMax has been used. There are 10 epochs and the applied optimization method is Stochastic Gradient Descent (SGD) with 0.1 as learning parameter. We have also applied cross validation with proportion of 0.1.

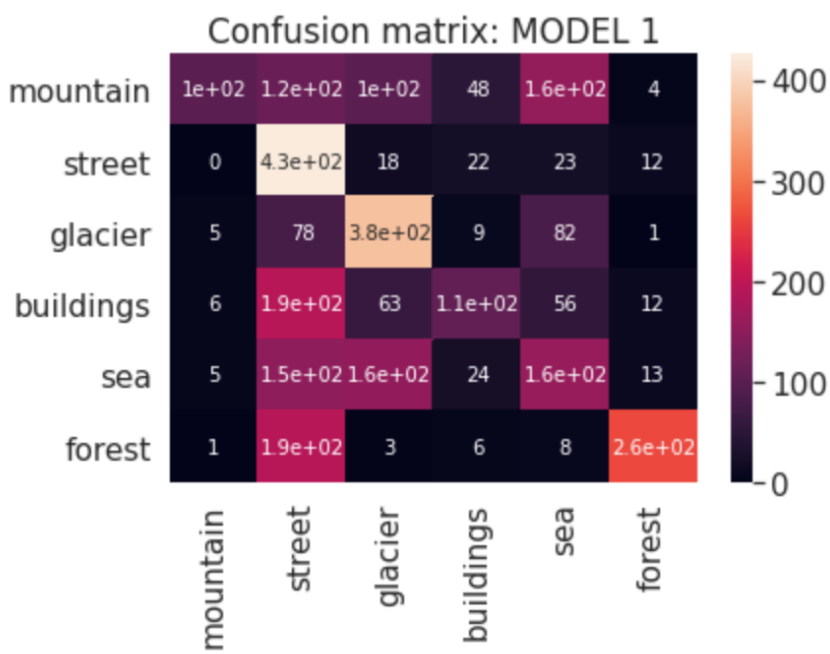

Fig. 3: Confusion matrix of model 1

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.19047619	0.8502994	0.6835443	0.24485126	0.31372549	0.55696203

Table 3: Recall of model 1

- **Model 1.2**

  Model 1.2 adds one hidden layer with same activation function (ReLU). So, the new model has three hidden layers and the test accuracy increases to 53.53%, comparing to model 1.

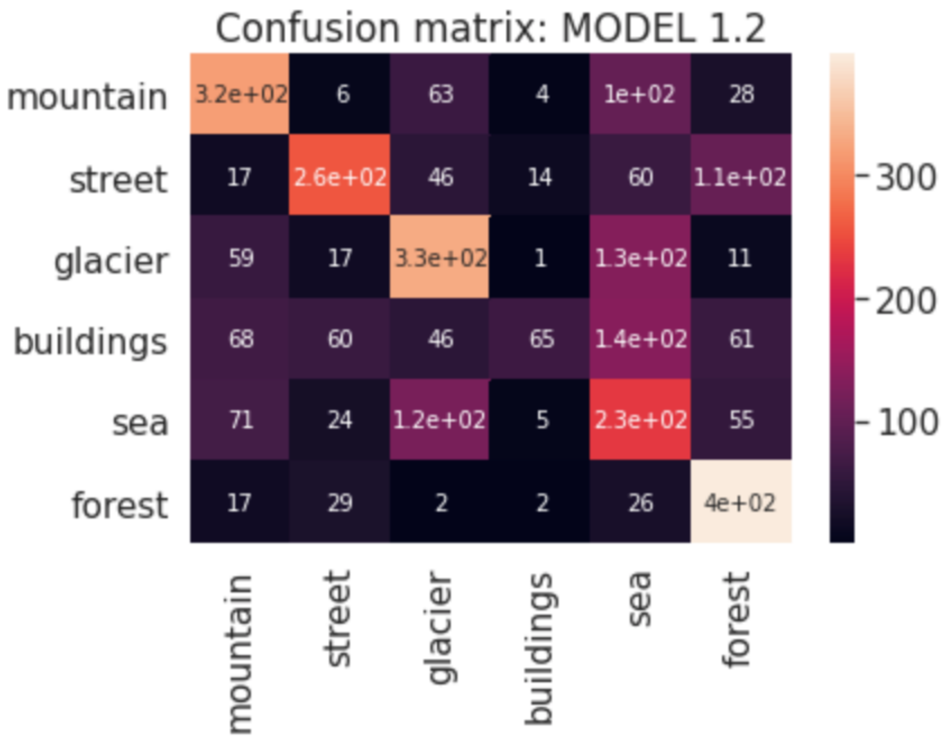

Fig. 4: Confusion matrix of model 1.2

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.60952381	.51497006	0.60036166	0.14874142	0.45686275	0.83966245

Table 4: Recall of model 1.2

- **Model 2**

  In model 2, all the parameters are the same as model 1, but we have decreased the learning rate from 0.1 to 0.001 with the same optimization method. Comparing to model 1 the accuracy of the test and training has decreased which is because of lower learning rate.

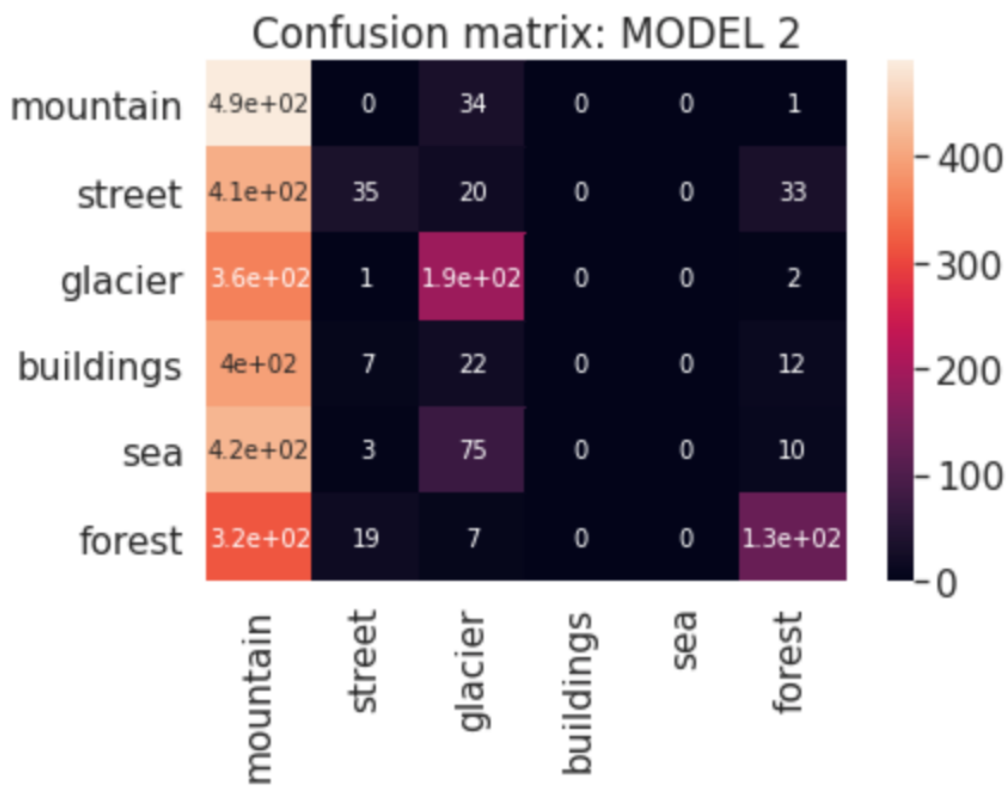

Fig. 5: Confusion matrix of model 2

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.93333333	0.06986028	0.34358047	0	0	0.27637131

Table 5: Recall of model 2

- **Model 2.1**
  In this model, we do not change the learning parameter but instead of applying SGD we will use Adam optimizer, which increases test and training accuracy to 57.56% and 68.7%, respectively.

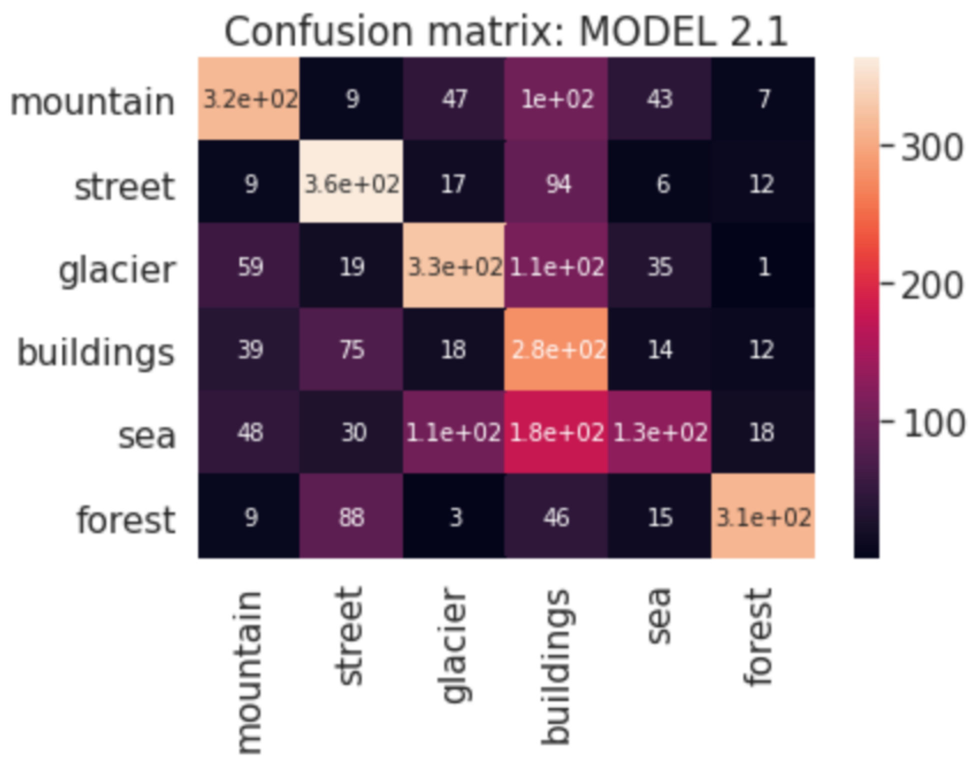

Fig. 6: Confusion matrix of model 2.1

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.6	0.7245509	0.59132007	0.63844394	0.25490196	0.66033755

Table 6: Recall of model 2.1

- **Model 3**
  Another parameter than can affect the model accuracy is proportion of validation split. In previous models we set it to 0.1, but in model 3 (with 3

hidden layers) it has increased to 0.2 and the new validation set increases the test error to 58.93% which shows less overfitting.

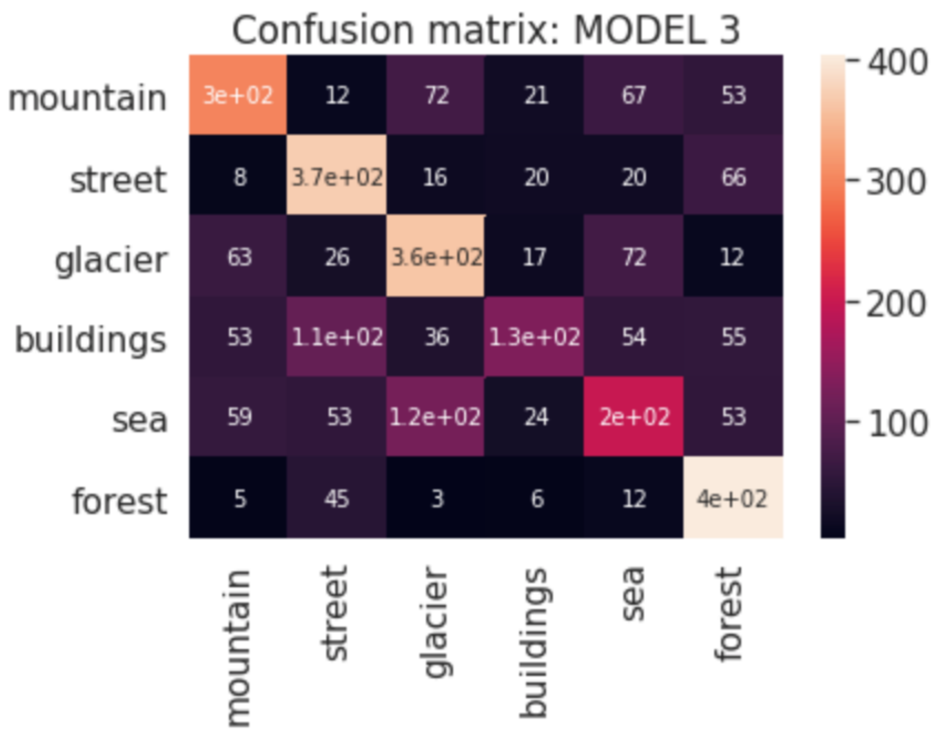

Fig. 7: Confusion matrix of model 3

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.57142857	0.74051896	0.656419	0.3020595	0.39019608	0.85021097

Table 7: Recall of model 3

- **Model 4**
  Activation functions are one of the most important factors in a neural network which are chosen based on the type of the problem, for example linear or non-linear. In model 4 we have changed activation function of last hidden layer from ReLU to tangent hyperbolic which is suggested for multi-layer neural networks but in this case, it does not improve the accuracy of the model.

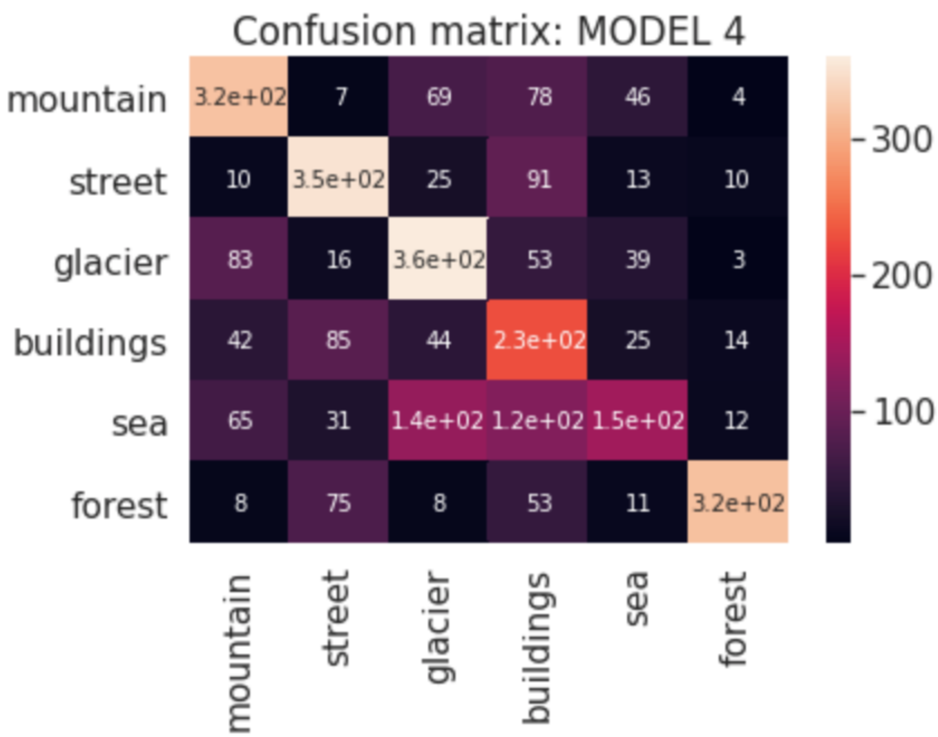

Fig. 8: Confusion matrix of model 4

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.61142857	0.70259481	0.64918626	0.5194508	0.28627451	0.67299578

Table 8: Recall of model 4

- **Model 5**
  In model 5, we have increased number of epochs from 10 to 15, so we had 15 complete passes through the training set, and it improved the training and test accuracy to 72.86% and 60.93%, respectively, which are the best gotten accuracies in implemented models.

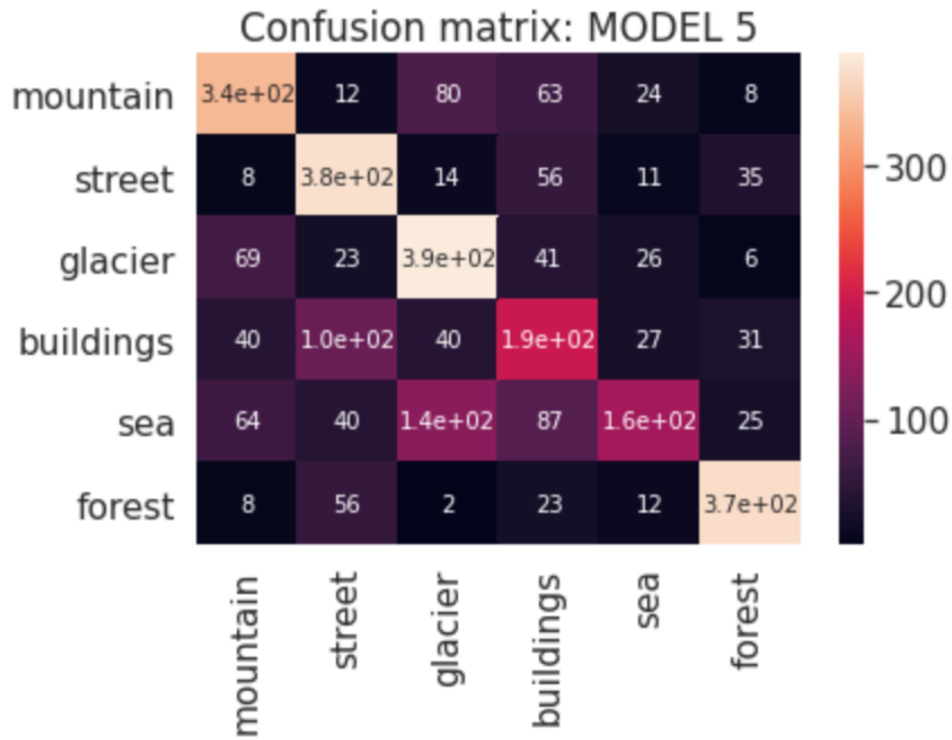

Fig. 9: Confusion matrix of model 5

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.64380952	0.75249501	0.70162749	0.44393593	0.30980392	0.78691983

Table 9: Recall of model 5

- **Model 6**
  Since we have got the best training and test accuracy, our goal is to decrease the overfitting. One of the methods to decrease the overfitting is L1 regularization which is also known as Lasso regression. By applying L1 regularization the accuracy of the model decreases dramatically.

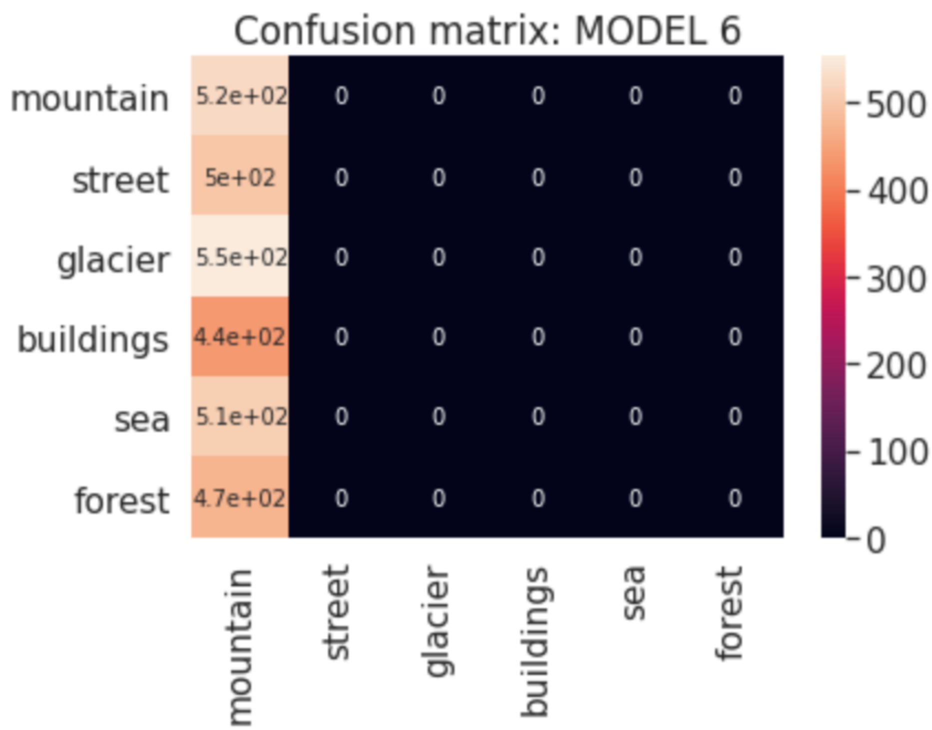

Fig. 10: Confusion matrix of model 6

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	1	0	0	0	0	0

Table 10: Recall of model 6

- **Model 6.1**
  Ridge regression or L2 regularization is another method to decrease overfitting and improve the model efficiency. In model 6.1, there is no overfitting, however the accuracy of the model is extremely low.

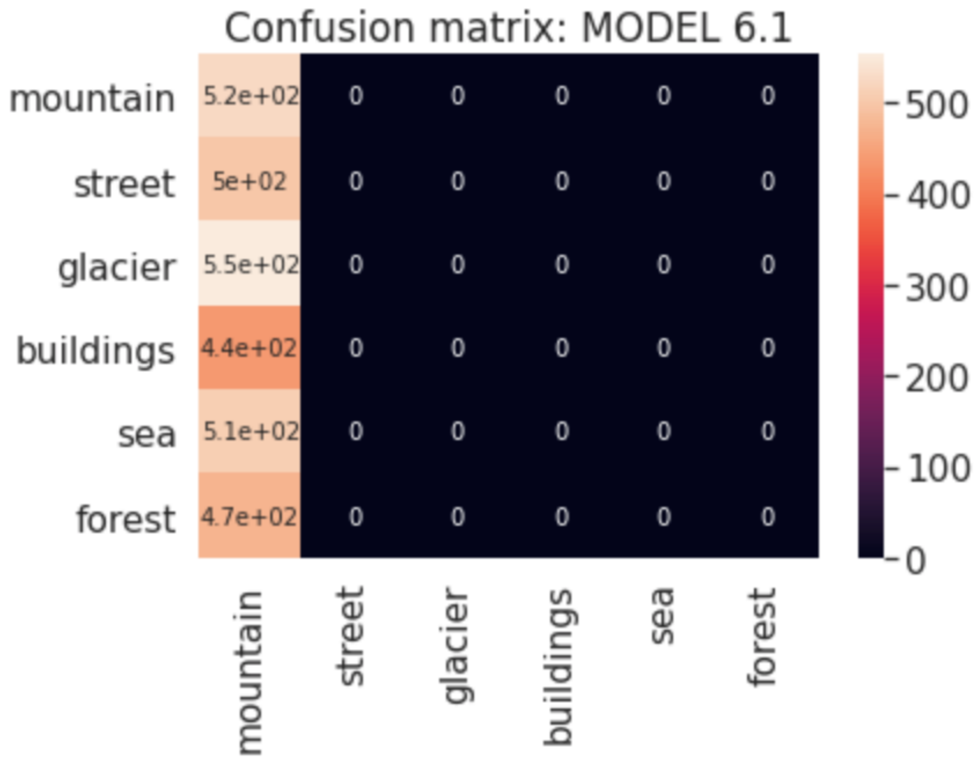

Fig. 11: Confusion matrix of model 6.1

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	1	0	0	0	0	0

Table 11: Recall of model 6.1

- **Model 6.2**
  Model 6.2 uses L1 and L2 regularization in two hidden layers which shows a reduction in training and test accuracy.

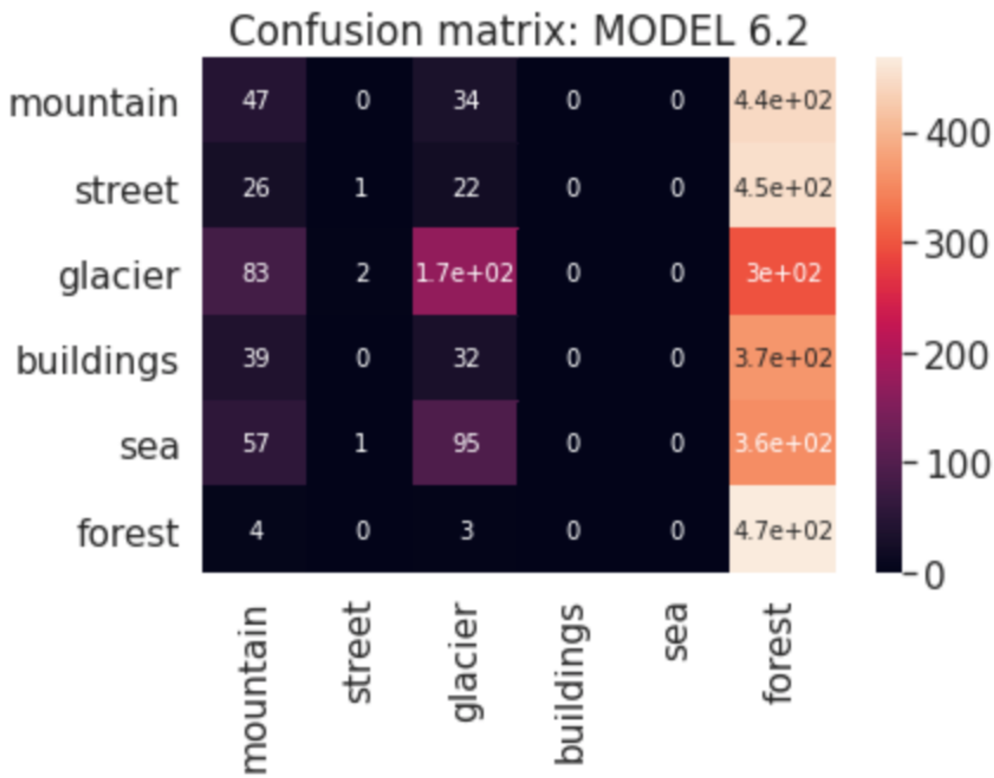

Fig. 12: Confusion matrix of model 6.2

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.08952381	0.00199601	0.3074141	0	0	0.98523207

Table 12: Recall of model 6.2

- **Model 6.3**
  Dropout is a common regularization method for neural networks to prevent overfitting. It sets the outgoing edges of neurons in hidden layers to 0 randomly. There is a dropout rate which as between 0 and 1, and in model 6.3 it is set to 0.5. Comparing to the other regularization methods, dropout works more efficiently on this model and the test accuracy is 58.17% which is one of the highest gotten accuracies.

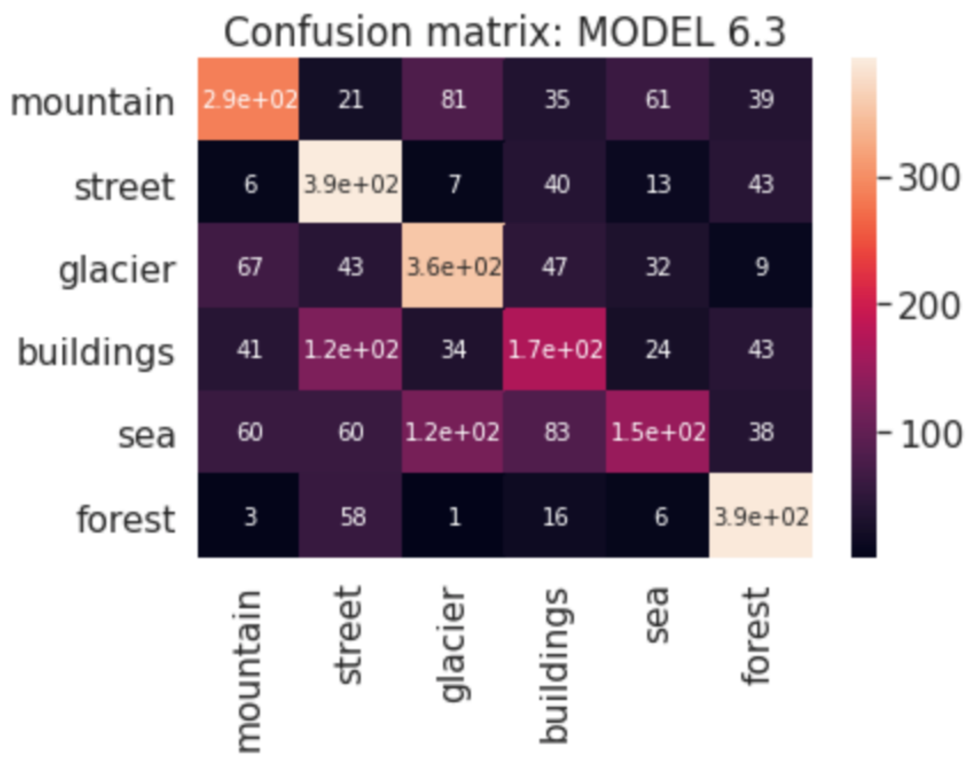

Fig. 13: Confusion matrix of model 6.3

	Mountain	Street	Glacier	Buildings	Sea	Forest
Recall	0.54857143	0.78243513	0.64195298	0.38901602	0.29411765	0.82278481

Table 13: Recall of model 6.3

5. Conclusions

Finally, after comparing different implemented models in section 4 we have observed:

1. Increasing number of hidden layers can increase the efficiency of the model, but we must also prevent overfitting.
2. Decreasing and increasing learning rate may decrease the accuracy of the model, so it is suggested to find the best one.
3. Adam optimizer works better than SGD in this study.
4. If the validation accuracy does not follow training accuracy, you can increase validation split proportion.
5. Increasing number of epochs will increase the training accuracy.

6. Dropout is more suitable than L1 and L2 regularization in this neural network.

We suggest applying CNN on this dataset and also increasing number of layers will increase the accuracy of this classification problem. Also, the regularization methods can be improved, and they can help to obtain a better model.