**Pritom Bhowmik**

**ABSTRACT**

Neural networks are used as a method of deep learning, one of the many subfields of artificial intelligence. They were first proposed around 70 years ago as an attempt at simulating the way the human brain works, though in a much more simplified form. Individual ‘neurons’ are connected in layers, with weights assigned to determine how the neuron responds when signals are propagated through the network. Previously, neural networks were limited in the number of neurons they were able to simulate, and therefore the complexity of learning they could achieve. But in recent years, due to advancements in hardware development, we have been able to build very deep networks, and train them on enormous datasets to achieve breakthroughs in machine intelligence.

These breakthroughs have allowed machines to match and exceed the capabilities of humans at performing certain tasks. One such task is object recognition. Though machines have historically been unable to match human vision, recent advances in deep learning have made it possible to build neural networks which can recognize objects, faces, text, and even emotions.

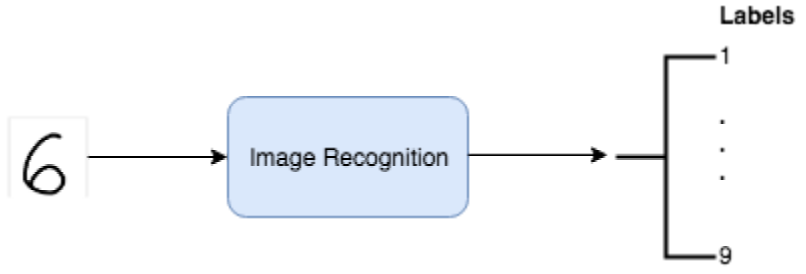
In this tutorial, you will implement a small subsection of object recognition—digit recognition. Using [TensorFlow](https://www.tensorflow.org/), an open-source Python library developed by the Google Brain labs for deep learning research, you will take hand-drawn images of the numbers 0-9 and build and train a neural network to recognize and predict the correct label for the digit displayed.

While you won’t need prior experience in practical deep learning or TensorFlow to follow along with this tutorial, we’ll assume some familiarity with machine learning terms and concepts such as training and testing, features and labels, optimization, and evaluation. You can learn more about these concepts in [An Introduction to Machine Learning](https://www.digitalocean.com/community/tutorials/an-introduction-to-machine-learning).

**BACKGROUND**

Handwritten character recognition is a field of research in artificial intelligence, computer vision, and pattern recognition. A computer performing handwriting recognition is said to be able to acquire and detect characters in paper documents, pictures, touch-screen devices and other sources and convert them into machine-encoded form. Its application is found in optical character recognition and more advanced intelligent character recognition systems. Most of these systems nowadays implement machine learning mechanisms such as neural networks. Machine learning is a branch of artificial intelligence inspired by psychology and biology that deals with learning from a set of data and can be applied to solve wide spectrum of problems. A supervised machine learning model is given instances of data specific to a problem domain and an answer that solves the problem for each instance. When learning is complete, the model is able not only to provide answers to the data it has learned on, but also to yet unseen data with high precision.

Handwritten character recognition can be thought of as a subset of the image recognition problem.



The general flow of an image recognition algorithm.

Basically, the algorithm takes an image (image of a handwritten digit) as an input and outputs the likelihood that the image belongs to different classes (the machine-encoded digits, 1–9).

We will look into the [Support Vector Machines](https://en.wikipedia.org/wiki/Support_vector_machine) (SVMs)  techniques to solve the problem.

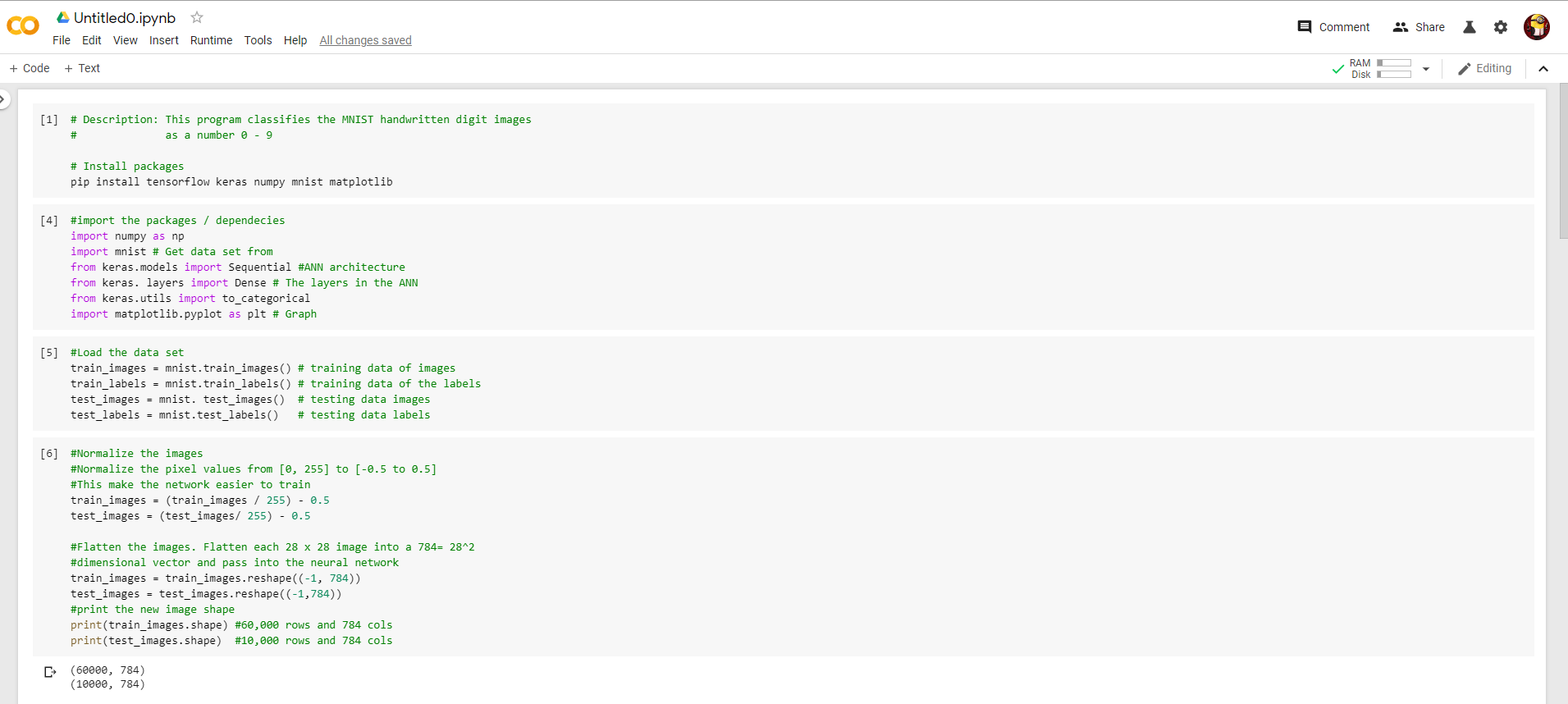
We will be using the **accuracy score** to quantify the performance of our model. The accuracy will tell us what percentage of our test data was classified correctly. The accuracy is a good metric choice because it will be easy to compare our model’s performance to that of the benchmark as it uses the same metric. Also, our dataset is balanced (equal number of training examples for each label) which makes the accuracy appropriate for this problem.

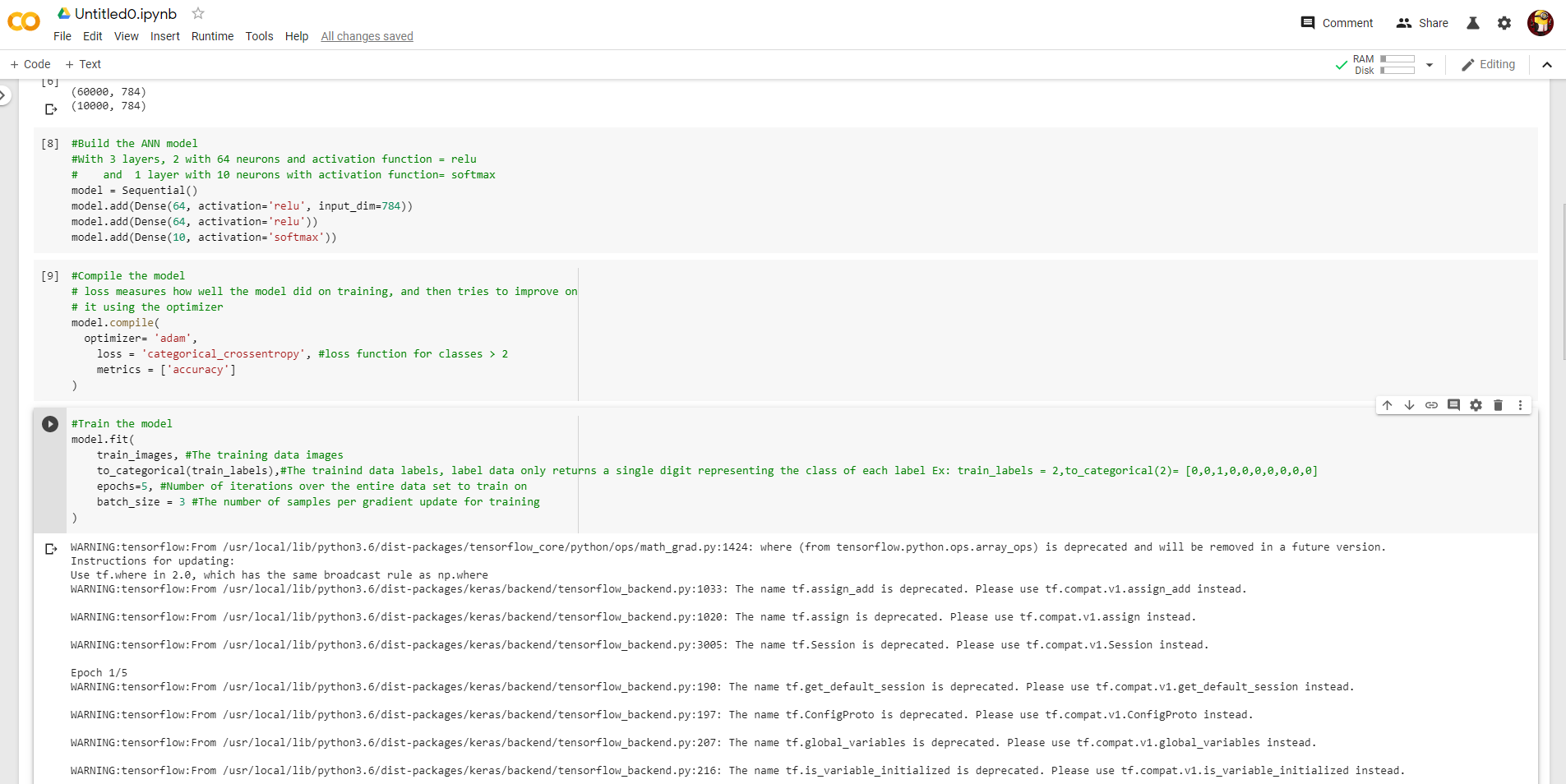
**Use of Database**: For pattern recognition related applications, data patterns are one of the most necessary requirements. If the data patterns for the particular recognition application is not available, then the first and foremost task in implementing the recognition system is to collect the data patterns. Data collection is one of the tedious task in most of the pattern recognition applications. The handwritten documents are collected and stored ..

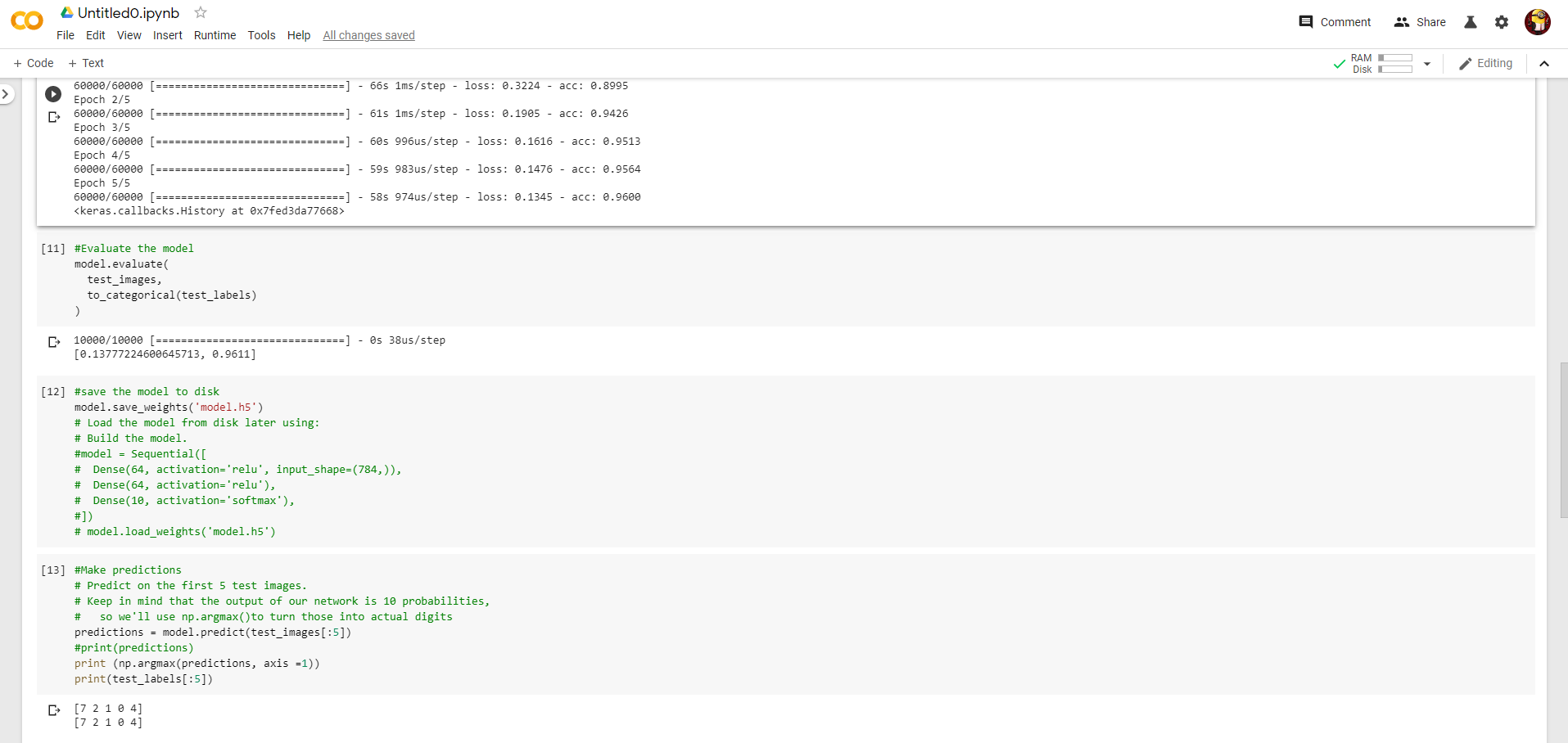
The present version of the character image database consists of binary isolated character images extracted from the collected handwritten data sheets using character segmentation algorithm.

The created character image database is available on request[1](https://www.sciencedirect.com/science/article/pii/S2215098618301447" \l "fn1), and is released in the form of comma separated values (CSV) files. Three CSV files representing training, validation and testing images are available. Each row in the CSV files, represents a character image.

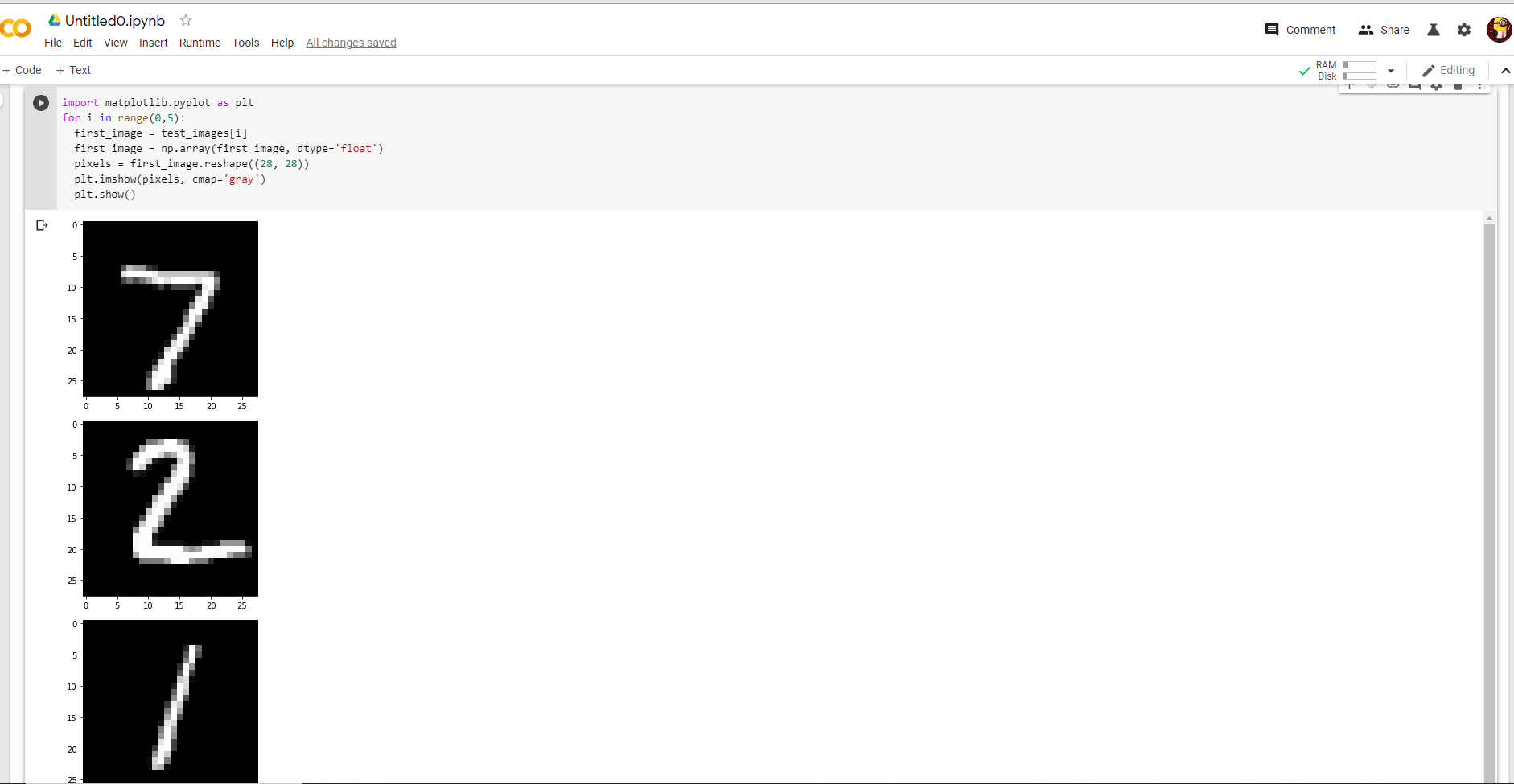
**SOURCE CODE:**

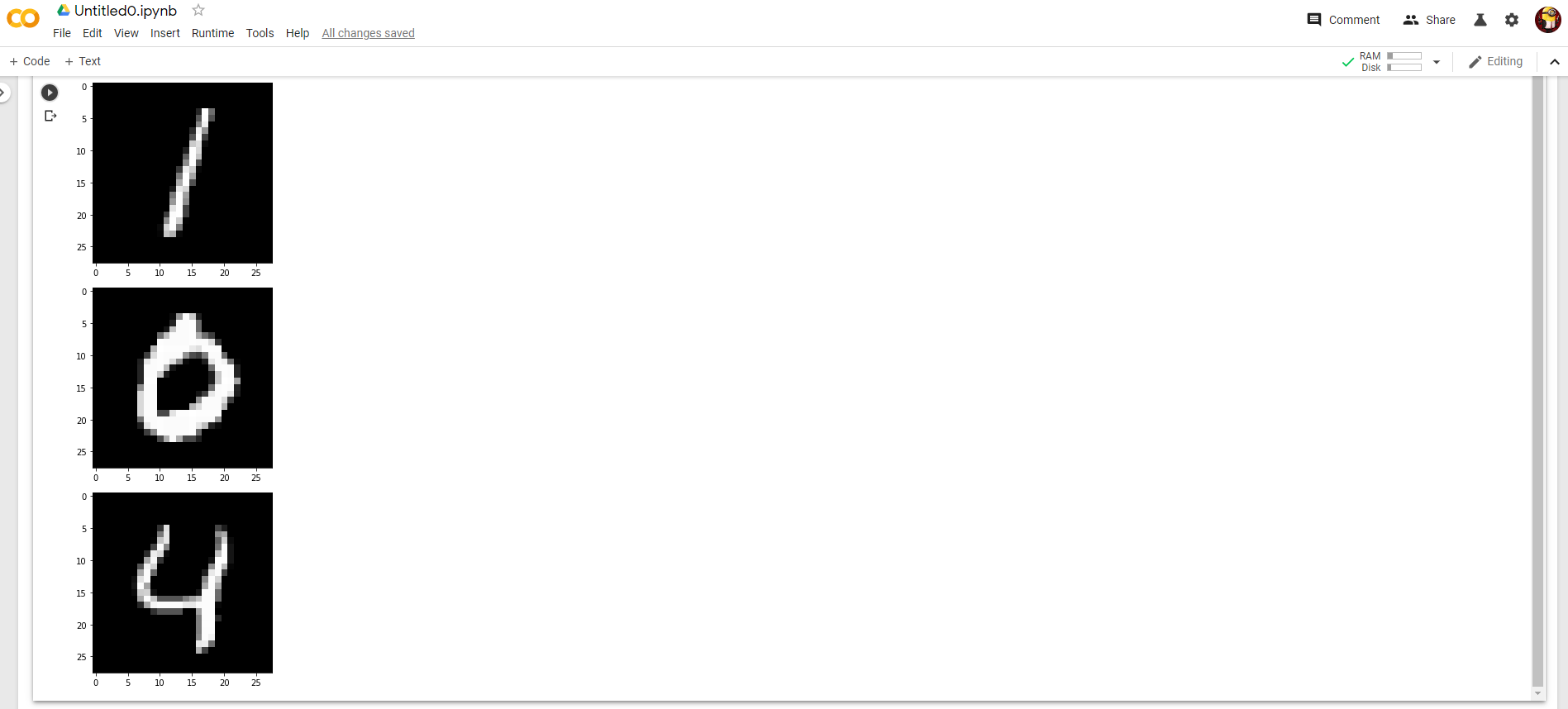
****

****

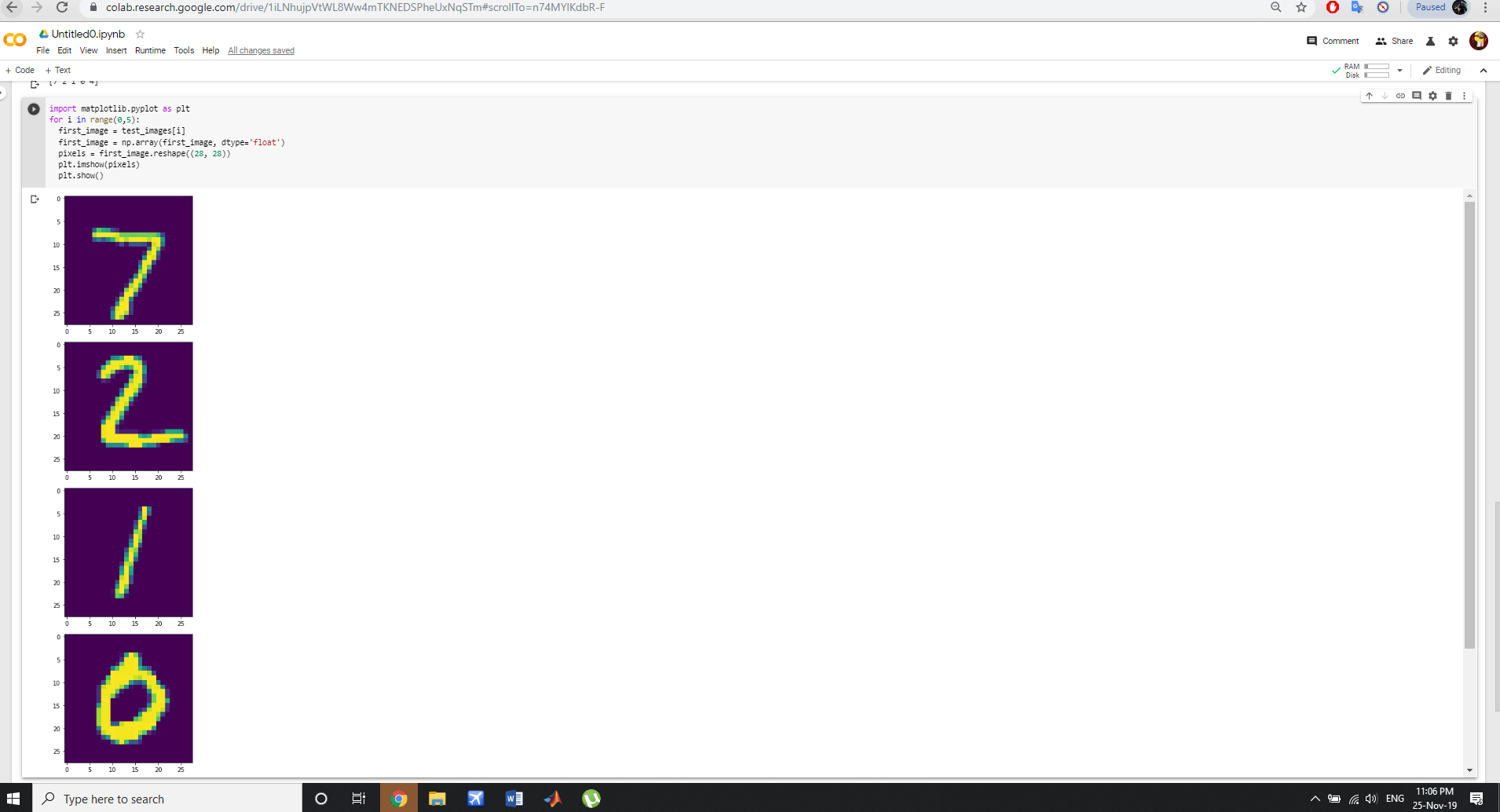


Experimental Results:





**Output in Pixel:**



**Test application Analysis:**

The test application accompanying the source code can perform the recognition of handwritten digits. To do so, open the application (preferably outside Visual Studio, for better performance). Click on the menu File and select Open. This will load some entries from the Optdigits dataset into the application. To perform the analysis, click the Run Analysis button. Please be aware that it may take some time. After the analysis is complete, the other tabs in the sample application will be populated with the analysis' information. The level of importance Experiments were performed on different samples having mixed scripting languages on numerals using single hidden layer.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data set** | **Training Set Size** | **Testing Set Size** | **Validation Set Size** | **Training Set Accuracy** | **Test Set Accuracy** | **Validation Set Accuracy** |
| Digit | 1778 | 6270 | 5430 | 96 | 97 | 96 |

**Table:** Detail Recognition performance of SVM

It is observed that recognition rate using SVM is higher than other model, i.e. Hidden Markov Model. However, free parameter storage for SVM model is significantly higher. The memory space required for SVM will be the number of support vectors multiply by the number of feature values. This is significantly large compared to HMM which only need to store the weight. HMM needs less space due to the weight-sharing scheme. However, in SVM, space saving can be achieved by storing only the original online signals and the penup/ pen-down status in a compact manner. During recognition, the model will be expanded dynamically as required. SVM clearly outperforms in all three isolated character cases. The result for the isolated character cases above indicates that the recognition rate for the hybrid word recognizer could be improved by using SVM instead of HMM.

**CONCLUSION**

In this tutorial you successfully trained a neural network to classify the MNIST dataset with around 92% accuracy and tested it on an image of your own. Current state-of-the-art research achieves around 99% on this same problem, using more complex network architectures involving convolutional layers. These use the 2D structure of the image to better represent the contents, unlike our method which flattened all the pixels into one vector of 784 units. You can read more about this topic on the [Tensor-Flow website](https://www.tensorflow.org/api_docs/python/tf/nn/convolution), and see the research papers detailing the most accurate results on the [MNIST website](http://yann.lecun.com/exdb/mnist/).

Now that you know how to build and train a neural network, you can try and use this implementation on your own data, or test it on other popular datasets such as the [Google Street-View House Numbers](http://ufldl.stanford.edu/housenumbers/), or the [CIFAR-10](http://www.cs.utoronto.ca/~kriz/cifar.html) dataset for more general image recognition.

**FUTUREWORK**

Future works on the database includes extending the character class collection by including all the presently used valid orthographic shapes for specific language script and creating word, line and page level collection of document images so that the researchers can focus on other stages of document recognition system as well.

It has been shown that Support Vector Machines (SVMs) can be applied to image and hand-written character recognition. However, SVMs don’t perform well in large datasets as the training time becomes cubic in the size of the dataset. This could be an issue as bigger datasets dataset containing thousands of samples which is quite large. To deal with this issue, a technique can be proposed,which is to train a support vector machine on the collection of nearest neighbors in a solution they called “SVM-KNN” . Training an SVM on the entire data set is slow and the extension of SVM to multiple classes is not as natural as Nearest Neighbor (NN). However, in the neighbor-hood of a small number of examples and a small number of classes, SVMs often perform better than other classification methods.

We can use NN as an initial pruning stage and perform SVM on the smaller but more relevant set of examples that require careful discrimination. This approach reflects the way humans perform coarse categorization: when presented with an image, human observers can answer coarse queries such as presence or absence of an animal in as little as 150ms, and of course, can tell what animal it is given enough time. This process of a quick categorization, followed by successive finer but slower discrimination was the inspiration behind the “SVM-KNN” technique.