LLM

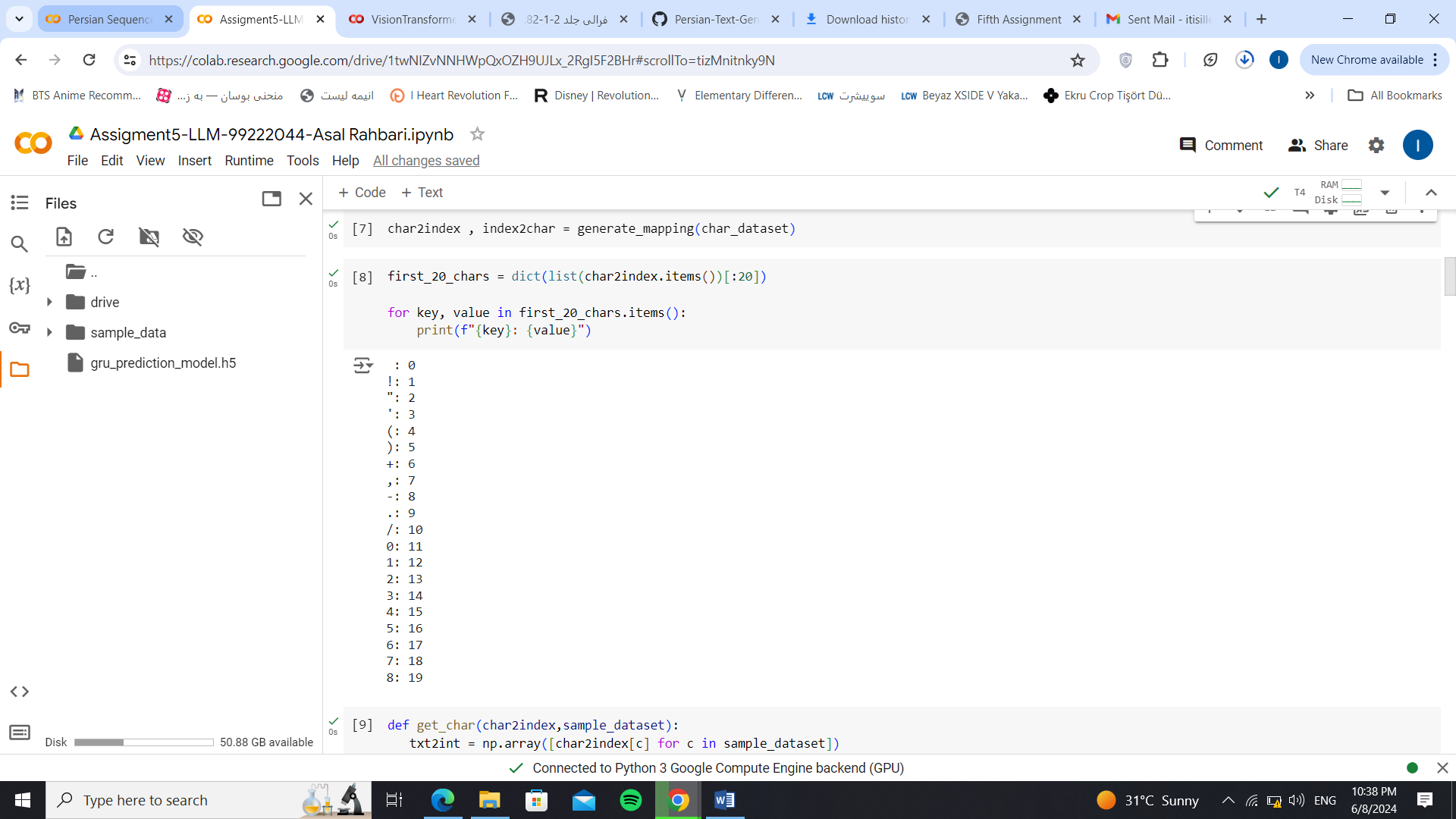
In this task we’ll have an LLM trained on the Persian Wikipedia dataset. This LLM given a word will predict some words (as many as we want) coming after that.

1. Loading Dataset and Pre-processing: Since this dataset is quite large we will work with a sample of it. First I defined a function for loading the dataset and taking sample from it. This function will take the path of the data’s file, load it and then put all sentences together in a sequence (replacing ‘\n’ with space). After that a sample of size 1000000 is taken from the data and then it creates a list of unique characters of the data named ‘char\_dataset’.

‘char\_dataset’ contains all of the unique characters of the sample taken from data. In this case, the number of unique characters of file is 224.

The next function I defined is for generating two mappings for the char\_dataset, one from indices to characters another for characters to indices. These mappings will act as a dictionary that’ll help us with processing and accessing characters later.

As you can see the first 20 characters of the char\_dataset are as follows:

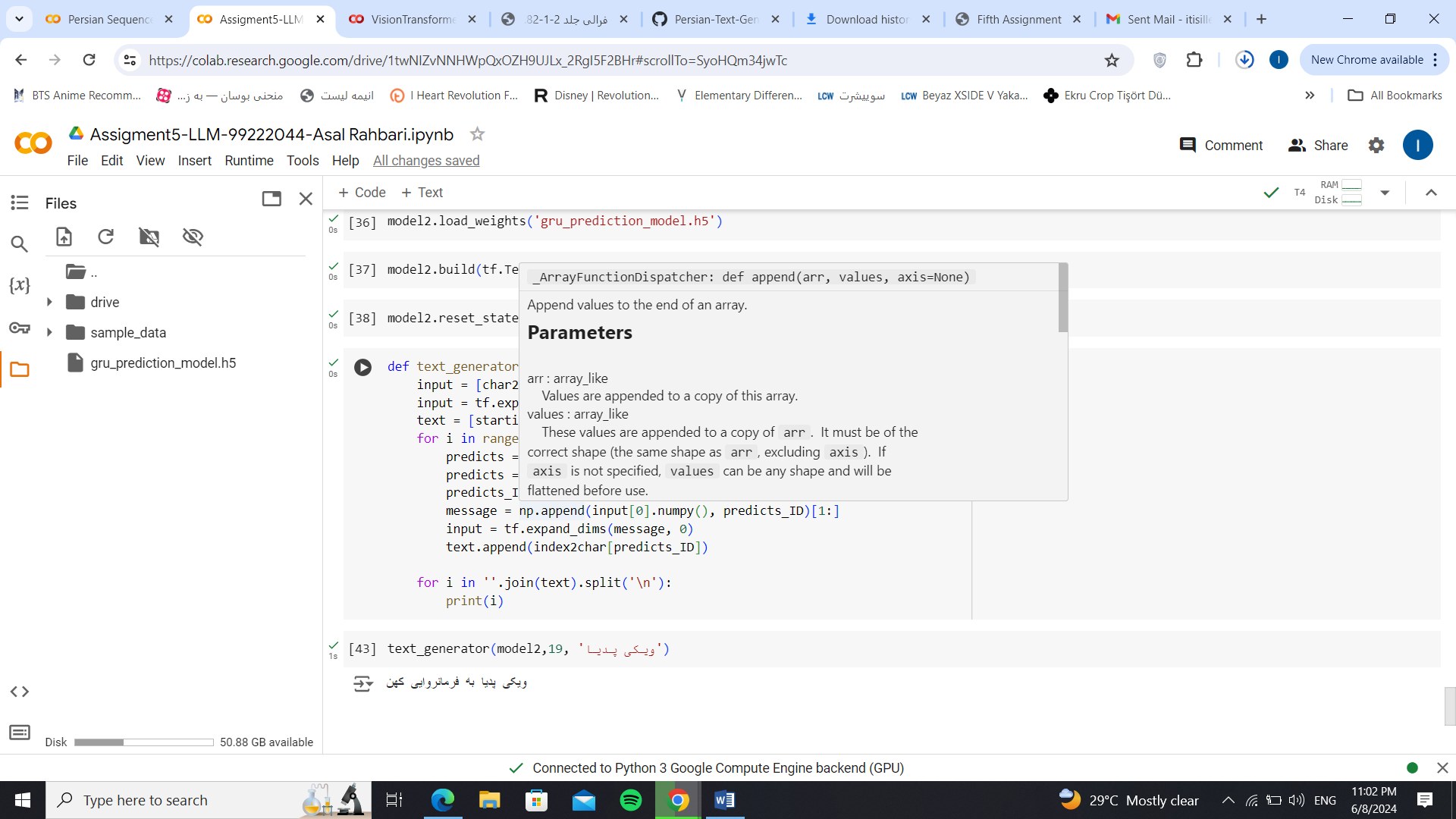


The next function defined will give us a tensor flow dataset that contains integer indices corresponding to the characters in the sample of dataset.

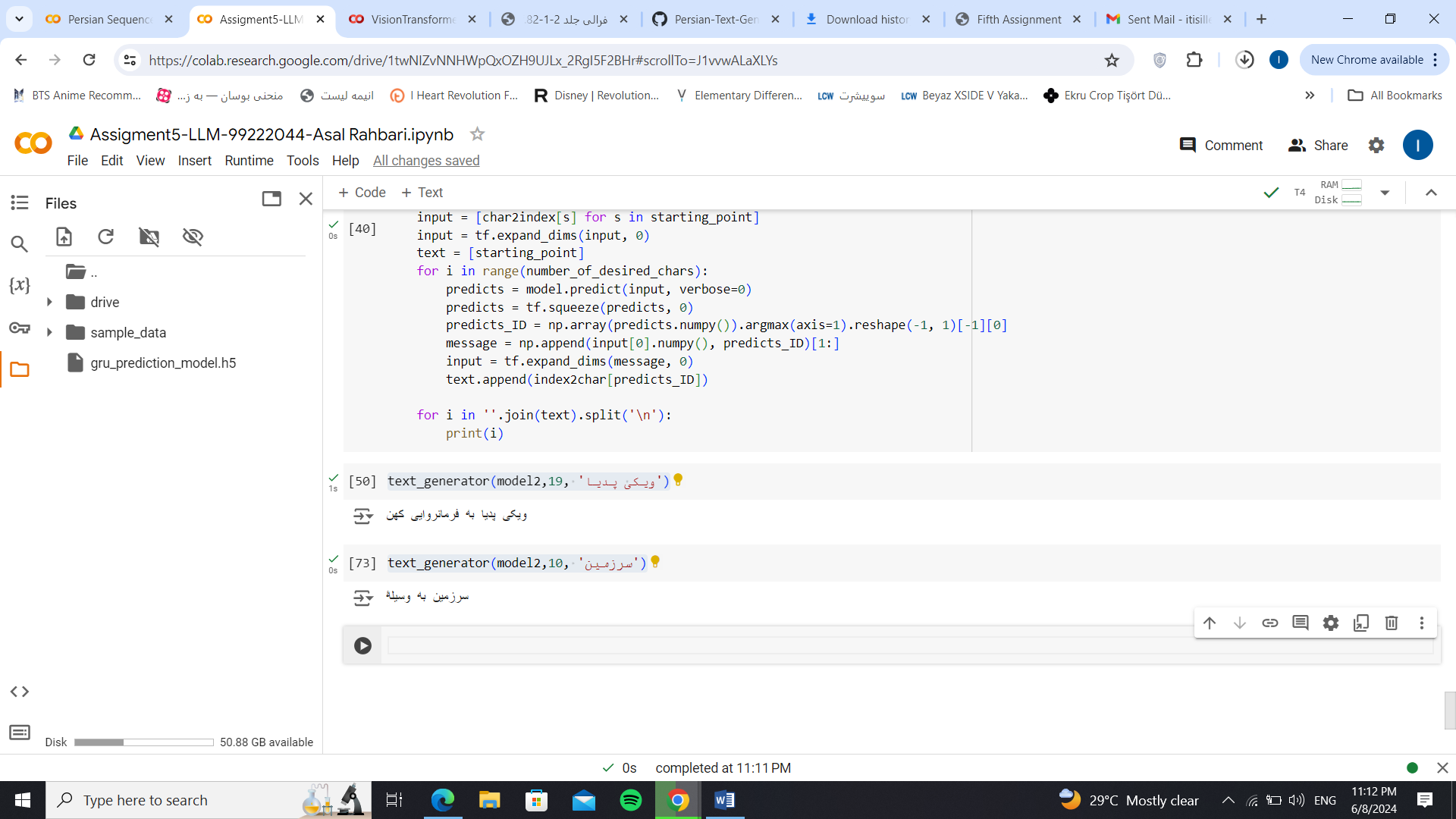
Now that we have our final desired dataset to work with, we split data into batches of size 100. (We drop the last remaining part of data if it has fewer samples than the batch size).

Next, a function is defined for splitting each batch of data into input and output text. The whole batch except for the last element is considered as the input text and the last element is the output text. We later apply this on the final dataset.

1. Model: In this stage, we will choose the model for our dataset and train it on the pre-processed data for some number of epochs. I’ve chosen a sequential model for this task. The first layer which is an embedding layer will convert the integer indices into dense vectors. GRU is a type of recurrent neural networks that processes sequences and captures dependencies. At last, a fully connected layer feed forward layer that will reduce the output’s dimensionality to 224 (number of unique characters). Since there are multiple unique characters in the dataset there exist multiple classes for prediction. So the CE loss is a good choice for this model. Adam optimizer is also the best choice for this task. I trained the model on the data for 40 number of epochs. I also used the concept of checkpoint to save the weights of the model during training.
2. Evaluation: Now we have to test the model we have trained to see if it’s really capable of generating text with different lengths for a given text.



In this example, we asked for the next 19 characters to be generated.



In this one we asked for 10 characters to be generated.

The End