Action, Object and Location Extraction with Roberta

Dataset

The dataset consists of a sentence and we are supposed to find the action, object and location from that sentence. This dataset seems to be like the AI used in smart home assistants in order to identify the actions and the objects to which that action should be done in a particular location. There are a total of 6 actions, 14 objects and 3 locations (kitchen, washroom, bedroom). There are sentences which doesn't have a location and for these location is represented in None. There are 11566 unique train examples and 3118 validation examples. But all these are duplicates of each other. The sound input must be different but there are only 248 unique sentences in train and validation datasets combined. So I divided it into 200 in train set and 48 in validation set. The train set size is very small. Also I made sure that none of the elements in my validation set are there in the train dataset.

Model

I used Roberta model. The given dataset size was very small so it would be very difficult for the model to learn from its own. So I searched for some pretrained state-of-the-art models and I selected Roberta for my experiments. Used pretrained Roberat model from huggingface library. Roberta is a transformer based model. Transformer models are mainly based on the idea of self attention. It wont process the sequence in order one by one as done by RNNs and LSTMs but take the whole input together as one and does the learning. The model was trained for 20 epochs with Adam optimizer and a learning rate of 1e-5. It achieved an accuracy of 100% on all the three labels namely – actions, objects and locations. The model was created in such a way that the backbone of Roberta was used and we get the Roberta maxpooled output from it. This output is passed through 3 dense layers parallelly so that we can predict for each of the three labels simultaneously and by just using the Roberta layer once. The model was trained using Sparse Categorical Cross Entropy loss.

Training

Training was done with the help of colab TPUs. The code can detect the resources available and can strategize accordingly. If no TPUs are available and multiple GPUs are available it performs a MirrorredStrategy such that the training task gets split between all the GPUs. If both TPUs and GPUs are not available training takes place in CPUs. The training process is done inside saarthi_train.py file. You should also pass --config argument along with this so that it can read the parameters/hyperparameters from the external.json file.

Needs the following as input files.

- train dataset
- validation dataset

Outputs the following files in the same folder.

- model.h5 – the weights of the final trained model (about 400MB in size)

- action, object, location encoders – dictionary which gives a mapping of each of actions objects and locations to corresponding label used for training.
- log.csv – gives the logging of the whole training process

Training was done with a batch size of 32 and for a total of 20 epochs. Got final accuracies of 100% for each of actions, objects and locations.

Inference

For inferencing you need to pass the path of the test dataset in the external.json file. Once you give the path it takes the file loads the model and it weight which was previously saved after training and after that do model.predict in order to get the output logit scores. These logit scores are then converted to labels which are then encoded back to the original actions, objects and location. These are compared with the original values and F1 scores are calculated. The F1 score comes out to be 100% when we compute it for the whole 248 examples in the train and validation dataset combined.

external.json file

It consists of all the parameters and hyperparameters required for training and inference.

- BATCH_SIZE → batch size to be used in training. Currently set to 32 batch size. When increasing batch size increase the number of epochs as well because now it will take more time to train. Make sure the batch size is less than the train dataset. Here the size of train dataset that we used is 200.
- MAX_LEN → refers to the length of the sequence that is getting input into the Roberta model. Suppose a given sentence has 5 words and MAX_LEN = 13 it will be padded to 11 words and the remaining two tokens are for representing the start and the end. I am using 13 here because I found out that 11 is the maximum number of words in a string in the given dataset.
- EPOCHS → number of epochs for training. It should be increased as batch size is increased.
- FOLDER_PATH → This is the path of the folder where all input is present for running the given.py file. All the outputs from that file would be dumbed to this folder as well.
- TEST_PATH → Path of the test file which is to be run during inference. Make sure this file is of the same format as train and validation csvs that are given to us. (should have the path column as well.. other wise gives error)