Name_Entity_Recognizer

Dataset

It is a dataset of clinical trial conducted by US-govt.

There are 4 files which contains almost the same data but there overlap between them is minimal. Dataset contains following columns:

ID	tags	text
NCT02105766	8:16:chronic_disease,20:32:treatment	portal fibrosis by liver biopsy

ID

It is a unqiue identifier for each row.

tags

It tells about the entity, where they start and end and what's category they comes under.

- For eg: **8:16:chronic_disease** this means that in corresponding text from 8th character to 15 character there is an entity and it's category is **chronic_disease**.

text

It is the actual sentence which has been collected from clinical survey. From this I have to first find which words belong to any of the pre-defined entites and then classify then under those category.

In all the three given files there is only 4 categories present which are: `treatement, chnoical_disease, cancer and allergy_name`. The distribution of these categories is same in all the given files. But the majority entity which is present in the data is `treatement and chronical_disease`. Lease present was `allergy_name`.

Data-Preprocessing

In tags column, there was some leading and trailing (commas), which I have removed. Then I made a new feature of `tokens` where I have splited the text into words using `nltk's word_tokenize`. So I got a list of the possible words and few special symbols also get splitted to individual category.

`ner_tags` which for each word present in the tokens, it mark it as `0.0` if it is not a part of any entity otherwise it maps as `{treatement: 1.0, chronical_disease: 2.0, cancer: 3.0, allergy_name: 4.0}`.

Introduction

My task is to find if in the given text there is any word which belong to these given enitities, if yes then under which category it belongs.

I used a fine-tuned DistilBERT model for this task. I have used `transformers` library for this task. I have used `DistilBERT` because it is a smaller version of `BERT` and it is faster than `BERT` and it is also giving good results.

Approach

I have fine-tuned the model in continous manner. First in the `G-1` dataset, then in `G-2` dataset and then in `G-3` dataset. For training, I used `Trainer` library of `HuggingFace`. Trained each task for 2 epoch.

Result.

	Performance on T_1 test set	Performance on T_1 and T_2 test sets	Performance on $T_1, T_2,$ and T_3
Precision	0.665049	0.679740	0.709221
Recall	0.732900	0.741376	0.757616
F1-Score	0.697328	0.709221	0.722247

- From above table, we can see that the continuous fine-tuning of the model is giving better results. But the last model which is trained on the combined dataset is giving the best results. It could be improved further by usingg more-complex model for fine-tuning.

Links

- Processed_Dataset
- T_1 Model
- T_2 Model
- T_3 Model
- Combined Model

What to do next?

Because of less time I was not able to format the model's output. Currently it gives the output for each of the tokenized word. But it should give the combined entity. For eg: `NB` is a `chrmonical_disease` and `NB` is a single word. But tokenizer tokenizes it as `N` and `B`. Hence currently it is giving output as `N` and `B` as `chrmonical_disease`. But it should give `NB` as `chrmonical_disease`.