Assignment 2: Semantic Analysis via Text Classification

In this assignment, you will need to re-investigate text classification problems by leveraging more sophisticated content embedding techniques, e.g., word2vec and BERT models. Before you start, please read the following tutorials carefully:

1. Text classification using word2vec (pretrain word embeddings by using word2vec model. If you need more assistance, please refer my slides and the class readings)
2. How to Fine Tune BERT for Text Classification using Transformers in Python (using BERT to train word and sentence embeddings, either just pretrain or pretrain + fine-tune)

Then, in this assignment, you will need to implement a review sentiment classification task via Amazon Fine Food Reviews. This dataset consists of reviews of fine foods from amazon. The data span a period of more than 10 years, including all ~500,000 reviews up to October 2012. Reviews include product and user information, ratings, and a plain text review. It also includes reviews from all other Amazon categories.

Note that, you don't need to use the whole dataset for this assignment. Using only 10-20% of the data is sufficient to complete all tasks. You can use Pandas Dataframe.sample() function for data sampling.

In this assignment, we need to use GPUs to speed up the training procedure. If you don't have GPU resources on your local machine, you are welcome to use Google Colab with a free one.

Overall steps: 1. Install/import corresponding packages. 2. Exploratory Data Analysis (EDA). 3. Create labels based on the **score**, e.g., score>3: positive, score<3: negative. 4. Resampling for imbalanced dataset if needed (Balancing positive and negative reviews). 5. Use 10-20% sub-dataset and split it into training and test. 6. Use **Text** and **created labels** as features and ground truth to complete the following tasks.

Task 1 – Re-do TFIDF approach for this task.

In this part, please implement the TFIDF based classification by using the method(s) that you proposed in assignment 1.

Task 2 – Review classification by using word2vec.
In this part, please implement the word2vec based classification by using the method(s) that you proposed in assignment 1.

Hint:

1. Instead of downloading the embedded file, you can load it by using the following command.

```
import gensim
import gensim.downloader as gensim_api

embeddings = gensim_api.load("word2vec-google-news-300")
```

Task 3 – BERT (without fine-tune) for review classification.

In this task, you will need to use the pretrained BERT pipeline('sentiment-analysis') for review classification. Note that you don't need to fine-tune the model for this dataset/task.

Hint:

1. You can find a tutorial with the example code here.
2. Using raw text data as inputs (No preprocessing required).
3. You might need to truncate each sentence to the maximum length the model can accept (<=512) before feeding it into the classifier.

```
result = classifier(i[0:max_length])
```

Task 4 – BERT (with fine-tune) for review classification.

By using the deliverable from task 3, please fine-tune the BERT model for this dataset/task.

Hint:

1. If you want to use the same Pytorch dataset class code from the tutorial, make sure your input format is the same as its. You need to convert your Pandas Dataframe to List or NumPy Array.
2. `RuntimeError: No CUDA GPUs are available`
   Solution: Go to Menu > Runtime > Change runtime type > GPU.
3. `RuntimeError: CUDA out of memory.`
   Solution: Reduce the batch size to solve GPU memory limitation, then restart the kernel.
   per_device_train_batch_size=8,
   per_device_eval_batch_size=10 ,
   Runtime > Restart runtime
   Check GPU memory status via `!nvidia-smi`

4. ValueError: --load_best_model_at_end requires the saving steps to be a round multiple of the evaluation steps, but found 500, which is not a round multiple of 200.
   Solution: reduce logging_steps, e.g., logging_steps=100

Task 5 – Results Analysis.

Please summarize all the results (from task 1 - 4) in the following table, and please try to interpret this result, e.g., why method X outperforms method Y?

Method	Precision	Recall	Accuracy	F1
TFIDF-model 1				
TFIDF-model 2				
TFIDF-model 3				
word2vec-mode 1				
word2vec-mode 2				
word2vec-mode 3				
BERT w/o fine tune				
BERT fine tune				