# SP25 IS567 Text Mining

Assignment 6: Deep Neural Networks; Large Language Models

­­­

Due Date: 4/27/2024, 11:59PM CT

**Goal:**

The goal of this assignment is to help you gain hands-on experience with using deep neural networks and large language models for text classification tasks. Specifically:

1. You will use a tweet sentiment classification dataset (same as the dataset in assignments 1-3 & 5).
2. You will explore Bidirectional Encoder Representations from Transformers (BERT) for text classification using Google Colab and write about your findings.
3. You will explore Fine-tuned LAnguage Net Text-To-Text Transfer Transformer (FlanT5) for text classification using Google Colab and write about your findings.

**Deliverables:**

The submission must include **a narrative document as a PDF file** and **a jupyter notebook that you export from Google Colab** and must be uploaded to Canvas.In your narrative document, include screenshots and figures as necessary to answer the questions. Make your writing informative!

**Suggestions on technical resources:**

The following web resources provide the text mining fundamentals that you may need to complete this assignment.

**Pre-trained word embeddings:**

* BERT-base-uncased model: <https://huggingface.co/google-bert/bert-base-uncased>
* FLAN-T5-base: <https://huggingface.co/google/flan-t5-base>

**PyTorch:**

* Data loading: <https://pytorch.org/tutorials/beginner/data_loading_tutorial.html>
* Training a classifier: <https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html>
* Optimization: <https://pytorch.org/docs/stable/optim.html>

**BERT:**

* BERT-related methods/functions on HuggingFace:

<https://huggingface.co/transformers/v3.2.0/model_doc/bert.html>

* BERT paper: Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019. <https://aclanthology.org/N19-1423/>

**FLAN-T5:**

* FLAN-T5-related methods/functions on HuggingFace: <https://huggingface.co/transformers/v3.2.0/model_doc/t5.html>
* FLAN-T5 paper: Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., ... & Wei, J. (2024). Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, *25*(70), 1-53. ﷟HYPERLINK "https://www.jmlr.org/papers/volume25/23-0870/23-0870.pdf"https://www.jmlr.org/papers/volume25/23-0870/23-0870.pdf

**BERT Implementation**

*Data and Google Colab setup*

1. Download two data files (*train.txt, test.txt)* from the Canvas assignment page: <https://canvas.illinois.edu/courses/52836/assignments/1166359>.
2. Open <https://colab.research.google.com>. Open “TextClassification\_BERT.ipynb” on the Canvas assignment page using Colab.
3. Change your Runtime setup to GPU: click on Runtime → Change Runtime Type → click “T4 GPU” option as shown in the screenshot below (Note: training neural networks with CPU will be extremely slow, so we are using GPU). T4 is the only choice for the free plans. More powerful GPUs (e.g., A100 GPU) require paying for an upgrade, but you should be able to do the assignment using T4 GPU.

A screenshot of a computer program

Description automatically generated

1. Start from the beginning of the notebook, run each section of code, and answer the following questions. Make sure to run the code following the order in the notebook and do not skip any section.

**Task 1 - corresponding to section 1**

* Briefly explain the section. In particular:
  + What are the inputs and outputs of the tokenizer?
  + What does the tokenizer function do?
  + What do “truncation” and “padding” mean in tokenizer?
  + What is the vocabulary size of the tokenizer?
  + What is the difference between the tokenizers used in CNN and in BERT?
* Add code to print out the ids of the following words in the final vocabulary: "people", "smile", "amazing", and "time".

**Task 2 - corresponding to section 2**

* Briefly explain the section. In particular:
  + What do TensorDataset and DataLoader do?
  + What are the inputs and outputs of TensorDataset and DataLoader?
  + What do “input\_ids” and “attention\_mask” of train\_encodings/test\_encodings denote?
  + What does “batch\_size” mean?
  + What does “shuffle” mean? Why set “shuffle” of train\_loader as True while set “shuffle” of test\_loader as False?

**Task 3 - corresponding to section 3**

* Briefly explain the section. In particular:
  + What does “AdamW” do?
  + What is a scheduler?
  + What does “num\_warmup\_steps=100” mean?
  + Will the learning rate set with the optimizer affect the trained model performance?

**Task 4 - corresponding to section 4**

* The BERT model used in the code is BertForSequenceClassification from Transformer. You can check the source code for this function at:

<https://github.com/huggingface/transformers/blob/v4.39.3/src/transformers/models/bert/modeling_bert.py#L1512>

* + What are the inputs and outputs of the BertForSequenceClassification forward function? How does BertForSequenceClassification calculate the loss?
* Run the code and interpret the results. In particular:
  + What is the best F1 score on the test set?
  + Do you find any relation between the training loss and test F1 score?

**Task 4 - corresponding to section 5**

* Briefly explain the section 5. In particular:
  + How does the function assign probability to each label for an input sentence?
  + Can we change the “max\_len” parameter in the function?

**Task 5 – Hyperparameter tuning**

* Change the hyperparameters to the values listed below:
  + Learning rate = 1e-3, 5e-3, 1e-4, 5e-4, 1e-5
* Run the train and evaluate code again and interpret the results. In particular, do you see any improvements/degradation in the performance? If yes (or no), what do you think are the potential reasons?

**FLAN-T5 Implementation**

Follow the same instructions above but this time open “TextGeneration\_FlanT5.ipynb” on the Canvas assignment page.

**Task 6 - corresponding to section 1**

* Briefly explain the section. In particular:
  + What is the vocabulary size of the tokenizer?
  + Why is the model adding 1 to the end of each sequence?
  + What is the difference between the encoded labels in BERT and in FLAN-T5?
* Add code to print out the ids of the following words in the final vocabulary: "people", "smile", "amazing", and "time".

**Task 7 - corresponding to section 2**

* Briefly explain the section. In particular:
  + What does “DataCollatorForSeq2Seq” do?
  + What does the “padding=True” mean? How does it pad sequence?
  + What does the “batched=True” mean?
  + Why is label\_pad\_token\_id set to -100?

**Task 8 - corresponding to section 3**

* We train the FLAN-T5 model using the Seq2SeqTrainer, which provides a complete training and evaluation loop for PyTorch sequence-to-sequence models in the Transformers library.

You can view the source code for Seq2SeqTrainer here:

<https://github.com/huggingface/transformers/blob/5f4ecf2d9f867a1255131d2461d75793c0cf1db2/src/transformers/trainer_seq2seq.py#L54>

The Seq2SeqTrainer class inherits functions from the parent class Trainer, which contains the implementation of the training function. You can view the source code for this function here:

<https://github.com/huggingface/transformers/blob/688f4707bfc5f6adc6f4f18c2081c5a66db590d1/src/transformers/trainer.py#L3698>

The loss function is defined here: <https://github.com/huggingface/transformers/blob/688f4707bfc5f6adc6f4f18c2081c5a66db590d1/src/transformers/trainer_pt_utils.py#L539>

* + How does Seq2SeqTrainer calculate the loss?
* Run the code and interpret the results. In particular:
  + What is the macro F1 score on the test set?

**Task 9 - corresponding to section 4**

* Briefly explain the section. In particular:
  + What does “trainer.predict()” do?
  + Explain each element of the output returned by “trainer.predict()”.
  + Explain the line “np.where(predictions.label\_ids != -100, predictions.predictions, tokenizer.pad\_token\_id)” and why we need this.
  + What does “skip\_special\_tokens=True” mean?

**Task 10 - corresponding to section 5**

* Briefly explain the section. In particular:
  + Explain the function “predict()”.
    - What is the input and what is the output?
    - How “predict()” produces output by input.

**Task 11 – Hyperparameter tuning**

* Change the hyperparameters to the values listed below:
  + Learning rate = 1e-3, 5e-3, 1e-4, 5e-4, 1e-5
  + num\_train\_epochs = 1, 5, 10
* Run the train and evaluate code again and interpret the results. In particular, do you see any improvements/degradation in the performance? If yes (or no), what do you think are the potential reasons?