Abstract
In this assignment, you will implement a tiny neural machine translation system in Tensor-Flow [Abadi et al., 2015], in Eager Execution mode. The starter code is provided for you, with all the data loading mechanisms and training driver implemented. Your job is to understand the starter code and implement a sequence-to-sequence (Seq2Seq) model [Sutskever et al., 2014], and potentially with attention [Bahdanau et al., 2015], to translate French into English. It is required that your implementation works on batches of sentences, as it is an important practice that significantly speeds up deep learning algorithms.

Code and Dataset
Datasets. There are 3 datasets provided to you along with the code file: Toy Reverse, English-Spanish and English-French.
1. Toy Reverse1: In practice, it's always a good idea to start with some toy dataset. This dataset contains 10K training pairs and 2K validation ones. Each single line contains a pair: source sentence and target sentence (which is the source being reversed). The Vocabulary size is only 24 and should be a good validation of your implementation. We recommend you play with this dataset first to make sure your model works well before moving on.
2. English-Spanish: This dataset has the similar format to Toy Reverse but for pairs of English and Spanish sentences.
3. English-French: This one is also similar but with French and is also the biggest one of the three. As with the last assignment, you will see the TODOs and hints which suggest what you should or should not do. You will be mainly working one 2 files: train.py and models.py and no need to change the other ones.

Your Work and Gradings
The entry point of the program is in the file train.py. From this file, relevant modules will be called.
It is your responsibility to read the code and figure out all the relevant points. We also provided some TODOs and hints, however, to make your tasks less challenging.
You are required to implement and train a Seq2Seq model on top of the given code base. The breakdown of the your work is as follows:

Implement a Seq2Seq model that runs in batches. We already did the padding and batching for you so you just focus more on the model itself and how to train it properly w.r.t what you have learned in class. Those are the breakdown details:
• **models.py**: You are to implement the Encoder and Decoder part of the Seq2Seq, extended from tf.keras.Layers. We expect you do use the tf.while_loop and the mechanism at the low level. You are also given the lstm inference function for its use in the Encoder and Decoder. If you plan to use high-level APIs from keras for submission, you will get Zero bonus point, although it's a good idea to use it to test. For example, if you find it's easy, you can use it for Decoder and test your lower-level implementation for the Encoder, or vice versa.

• **train.py**: You have to implement the Teacher-Forcing training mechanism in the teacher forcing method. At test time, however, you have to use evaluate method for you have no ground truths and have to live with your predictions.
• **Evaluate your work by yourself**: Periodically, the train driver saves your model and randomly presents 2 translations for you to observe the overview quality of the training. Furthermore, it also carries out the full validation translation and generate a file named translated n.txt where n is the final epoch number. It is very slow for this operation so we just set it up at the very end of the training, esp. for the English-French dataset. Nonetheless, it is not too long for the Toy Reverse dataset and you can test it at the end of a single epoch or however you want. When you have that translated file(s), you can calculate the BLEU score(s) as follow: cat translated 10.txt | sacrebleu source.txt in your dataset folder, using sacreBLEU2. The very first result of that script is what we use to grade you.

You are expected to implement a single RNN for Encoder and another one for the Decoder. For Decoder, it is recommended that you can implement Attention model(s) Bahdanau et al. [2015], Luong et al. [2015] which will greatly affect the final results.
Although not required, you can implement any kind of dropout you want. We recommend you leave it to the end if you still have time. Similarly for such extentions as stacked layers of LSTM, bidirectional layers or others.

These are the baselines:
• Toy Reverse: With no attention, you should achieve at least BLEU 50.0 and with attention, it should be at least BLEU 90.0. A good implementation should show that the test BLEU can be 81.7 (no attention) and 99.9 (attention), respectively after 30 epochs.
• English-French: The baseline is BLEU 10.0 (no attention). A good implementation should have at least BLEU 24.0 (no attention) with 30,000 training pairs.