ASSIGNMENT 3: Few-Shot Cross-Lingual Transfer Learning for Natural Language Inference

Motivation: The motivation of this assignment is to get practice with multilingual models for transferring knowledge from high resource languages such as English to low resource languages such as Hindi, and Swahili.

Scenario: The main challenge with building multilingual models is the scarcity of labeled (and oftentimes even unlabeled) data in low resource languages. This gives rise to the *Zero-Shot learning* paradigm, in which models are trained on high resource languages such as English and are tested on low resource languages that were never used for fine-tuning (i.e. zero-shot setting). In practical settings, we do have very little data for such low resource languages, which can be used in combination with abundant English data for more efficient knowledge transfer to the same language. We call this paradigm *Few-Shot learning.* In this assignment, we will exploit this paradigm to build multilingual Natural Language Inference (NLI) models for the low resource languages.

Problem Statement: The goal of the assignment is to build NLI models for low resource languages. We are given a labeled dataset of English having 100K examples and 1K examples each from the following 14 languages - *French (fr), Spanish (es), German (de), Greek (el), Bulgarian (bg), Russian (ru), Turkish (), Arabic (ar), Vietnamese (vi), Thai (th), Chinese (zh), Hindi (hi), Swahili (sw) and Urdu (ur)*. The goal is to build one or more models, **that will be tested on 4 languages -- *Chinese, Hindi, Swahili and Spanish*** for the performance on NLI task, in which given a premise and a hypothesis, the goal is to predict one of the 3 classes - (a) *entailment*, (b) *contradiction* and (c) *neutral*. To make it simple, at test time you will be given an example from one of these four languages and the language itself as a string (E.g. *'Swahili'*) and you need to perform inference. If you have built one model for each language, you can use the language field to select the right model. The other extreme is building a single model for all languages, in which case the language information won't be required at test time. Note that you are allowed to use all given data (from 15 languages) at training time, but you are not allowed to use any other data from any other source.

Labeled Training Data: The training data can be accessed from **/scratch/cse/phd/csz198394/NLP_fall22/A3/train/train.tsv** on HPC.
It contains around 100K examples of English and 1000 from each of 14 languages.
The format of the data is tsv (i.e. tab separated) with 4 columns - *gold_label*, *premise*, *hypothesis* and *language*. You can use this data whichever way you want to build a multilingual model for each target. This includes combining data from multiple languages for training as well as validation for each target.

Sample Test Data and Evaluation: The sample i/o test files can be found in the folder **/scratch/cse/phd/csz198394/NLP_fall22/A3/test/ .** We will follow the exact same format while evaluating. **Any errors (even minor ones) in the formatting will lead to 20% penalty.** (A2 had lots of formatting errors. This time we won't accommodate any).
Note:- The sample input contains the header but the output doesn't.

The evaluation metric will be both micro-F1 and macro-F1 (over 3 labels). You will be separately evaluated on each of the target languages. We will share an evaluation script soon.

Multilingual Pre-trained Language Models:
mBERT (from BERT's original paper i.e. Devlin et al. , 2019) and XLM-R (Conneau et al., 2020) are 2 most commonly used language models for typically all the multilingual NLP applications. Here is the list of languages supported by these LMs -- https://github.com/google-research/bert/blob/master/multilingual.md
You are allowed to use either of these models as your base starting point. You may load both these models from HuggingFace. Check out this tutorial - https://github.com/huggingface/notebooks/blob/main/transformers_doc/en/multilingual.ipynb for beginners.

Zero-Shot Cross-Lingual Learning Methods:

- **Translate-train:** One can translate English training data to the target language, then train the model on translated training data for effectively transferring the knowledge to the target language. Major limitation of this approach is the poor quality of translations that leads to noisy training.
- **Translate-test:** On the other hand, one can train on original English examples, but while inference translate target examples to English before running the

inference. This approach is less expensive than *Translate-train* because now we only have to translate a few examples from Target -> English whereas in *Translate-train,* the whole training data had to be translated from English -> Target.

- **Adapter-based**: A recent development in this domain is the use of Adapters, which are small modules inserted in transformer layers and are the only trainable modules (all the parameters of transformer freezed), that have shown promising results for zero-shot transfer from English to low resource languages. Check out documentation and collab tutorial if you are interested in exploring the usage of Adapters for cross-lingual transfer.

The Task:

Your submitted model(s) will be evaluated on 4 target languages - ***Chinese, Hindi, Swahili and Spanish***. As discussed earlier, you may like to train 4 models (one for each language) on one extreme or a single multilingual model for all target languages on the other extreme. We will give you language as a string (E.g. '*Hindi*') along with the example itself that needs to be classified into one of 3 target classes while grading. A few points to note are following-

1. You will need multilingual pre-trained models for this assignment. You are allowed to use **ONLY mBERT and XLM-Roberta** (also called XLM-R) for the task.
2. You are allowed **ONLY the following 2 translation models** (in case you plan to use *translate-train* or *translate-test* methods) - 1. m2m100 (checkout https://huggingface.co/facebook/m2m100_418M) and 2. mBART (https://huggingface.co/docs/transformers/model_doc/mbart)
3. You are allowed to use pretrained Language Adapters as well. Reference - link
4. You may like to group related languages (E.g. French, Spanish and German) and fine tune the model on these (with English data) to make multilingual model for this family of languages (E.g. for *Spanish*). The idea is to leverage data from related languages to make a stronger model than the one trained separately on each of these languages. For this purpose, phylogenetic trees are generally used (Ref - link) to compute distances between languages.
   Similar kinds of grouping can be done for validation as well (for saving the best checkpoint for a given target language). This might help you in better model selection for each target language.
5. You can create the validation split in any way you want from the given datafile.

Submission Format:

<u>**The submission deadline for the assignment is 30/11/2022 at 11:59 PM. There will be no late submission allowed.**</u>

<u>**The Assignment can be done in groups of one or two.**</u>

The submission will be on HPC similar to A2. Detailed Submission instructions will be shared soon on Piazza.

Evaluation Criteria:

1. This assignment is worth **15** points (i.e., 1.5x compared to A1 and A2). Note that total assignment percentage is 50% so these 35 total marks will be normalized to 50 when computing the final grade.
2. **We will take the mean of micro-F1 and macro-F1 scores based on all 3 labels for each language separately.**
2. Bonus points awarded for outstanding performers

What is allowed? What is not?

1. The assignment is to be done individually or in groups of two.
2. You should use Python 3.7 and PyTorch for this assignment.
3. You must not discuss this assignment with anyone outside the class. **Make sure you mention the names in your write-up in case you discuss with anyone from within the class.** Please read academic integrity guidelines on the course home page and follow them carefully.
5. We will run plagiarism detection software. Any team found guilty will be awarded a suitable penalty as per IIT rules.
6. Your code will be automatically evaluated. You will get a significant penalty (at least 20%) if it does not conform to output guidelines.