Project 1: Checkpoint 1 (Group #69)

Dataset: What kind of data are you going to use to train your model, and how will you obtain this data?

We need broad-based data for our model to effectively predict words from multiple languages.

We are going to first use the **WikiText-103 dataset**. We are going to obtain the data from Kaggle. The data includes 100 million tokens from verified articles in Wikipedia. This dataset will give a broad-based training set for our model.

We are also going to use the **OpenSubtitles dataset**. We are going to obtain the data from the Opus Project. The data is **multilingual** (94 different languages) and includes tokens from movie subtitles, which will help train our model for different languages.

Method: What kind of method will you use, and how will you implement it (e.g. language, framework)?

Because we are new to Natural Language Processing and we have learned about N-gram models (and Kneser-Ney smoothing) in class, we are going to start with a character-level N-gram model and use Kneser-Ney smoothing to consider context diversity. Once we implement this method, we will then implement a Word2Vec model. A Stanford paper named “Word2Vec using character n-grams” explains how to enhance and improve Word2Vec using character n-grams. We use character n-grams because our vocabulary is essentially unicode characters.

The following describes how we will implement a Word2Vec model with character n-grams:

We will use the Python3 language. We will use the Pytorch framework to build tensors on the neural side.

We will normalize the data and then hash our character n-grams as sub-sequences within each window. We will vectorize each input window as an average of embeddings from characters and n-grams. We will use a Softmax to output probabilities across our chosen Unicode vocabulary, and then select the top 3 words.