# E1246 - Natural Language Understanding
# Assignment2 : Language Models

**Tapan Bhardwaj(14704)**
tapanb@iisc.ac.in

## Abstract

In Assignment-1 I had developed a four-gram based language model on **Brown** and **Gutenberg** corpus and then calculated perplexity on both the corpus using 90-10 split. This assignment Assignment is also related to the development of Language model only, but this time, it is implemented using Long Short Term Memory (LSTM) Neural Network, a variant of Recurrent Neural Network (RNN).

## 1 Introduction

Current assignment is divided into three tasks which are mentioned below:

### 1.1 Task 1

This task is to implement token level LSTM-based language model using Gutenberg Corpus (D2) in following settings:

- S2: Train: D2-Train, Test: D2-Test

### 1.2 Task 2

This task is to implement character level LSTM-based language model using the same settings as mentioned in Task-1.

### 1.3 Task 3

Using the best model implemented in **Task 1** and **Task 2**, we need to generate a sentence of 10 tokens by running a script with the name generate-sentence.sh.

## 2 Dataset

Gutenberg Corpus is chosen for the development of both Language Models. Due to limitations of the computation resources, only 40 percent of the corpus is used. The division of the corpus (from 40 percent) is given below:

- Training : 90
- Testing : 10

## 3 Baseline Language Model

N-Gram Language Model of Assignment-1 is used as a baseline for **Task 1** and **Task 2**. Baseline model is run against the new corpus splits and results are used to compare the outputs of the **Task 1** and **Task 2** models.

## 4 Model Description

### 4.1 Character level LSTM Model

I have used single layer of 128 LSTM units and then dense layer of softmax is used to predict the probability of next character. Used 8 characters as input and predicted the next character probability and thus calculated the loss based on this probability.

### 4.2 Word level LSTM Model

I have used single layer of 128 LSTM units and then dense layer of softmax is used to predict the probability of next word. Model is also learning word embeddings during training and dimension for each word embedding used is 50. Model takes sequence of 20 words as a input and predict the probability of next word and thus loss is calculated based on the predicted probabilities.

## 5 Language Model

LSTM Model is developed using Keras library which facilitates different variants. In absence of GPUs, model is configured with CPU only. In addition, model is using the keras state saver utility, to save model parameters on each epoch. Basic understanding of the LSTM RNN is understood from blogs of Chris Olah, Andrej Karpathy and Goldberg Book(Goldberg, S, 2017).

### 5.1 Preprocessing

- Raw files of gutenberg corpus are concatenated and the text is changed to lower case.
- Punctuations are removed from raw text.

### 5.2 Generating Sentences

- Picking a seeding sequence of word or characters from training data uniformly at random.
- Generating a probability distribution of next word or character from the trained model
- Using numpy multinomial method to pick next word or character which follows above mentioned probability distribution.

## 6 Result

### 6.1 Word-Level Language Model Results:

perplexity = 110.7

### 6.2 Character-Level Language Model Results:

perplexity = 146.8

### 6.3 Four Gram Language Model Results:

perplexity = 180.7

### 6.4 Some examples of sentences generated from Character Language Model:

- but you mean what she had been the blain to
- ended the think her own while they were so anxious
- any pleasure to a little infime resolved to be made
- it was a great dear miss bates that i had been
- so much belong to her friends when he was all the convenience of her

## 7 Acuuracy/Measures

### 7.1 Task 1

Perpexility is used as the measure for this task.

### 7.2 Task 2

Human Evaluation.

## 8 Github Link

https://github.com/TapanBhardwaj/NLP-projects