# TP2: LSTM Language Model

Dorin Doncenco

November 2023

## 1 Introduction

A language model is designed to learn, understand and predict human language from its training data. In the context of this report, the task of the model is to learn the probabilities of words given an input sequence. The goal of this is to create a generative model, which is able to probabilistically generate language that simulates its training data (e.g. writing in a certain style).

## 2 Dataset

The dataset used for this project is the Penn Treebank, a dataset consisting of a collection of articles from the Wall Street Journal. It contains 9999 words, or tokens, among which <unk> serves for removed tokens, and $N$ serves as a number token. Additionally, the <bos> and <eos> tokens are added to all sentences, to signify the beginning and end of them. This information can be used to create a two-direction dictionary mapping words to IDs, enabling deep learning techniques to handle strings, a type of data which is not handled with ease by such algorithms.

## 3 Method

The architecture of the model briefly consists of a learnt embedder, a Long Short-Term Memory (LSTM) network, and a linear layer outputting its prediction on how likely each token in the dictionary is to follow the previous sequence fed into the model.

The embedder is able to convert word IDs to a latent representation, which will "learn" the meaning of these words and allow them to be processed by the LSTM.

The LSTM is an improvement to Recurrent Neural Networks (RNN). The RNN was introduced with the goal to process sequences of data, enabling inputs of varying lengths to be processed, opposed to typical linear neural networks, which are limited by a strict input format. This architecture also enables the model to take into account data where the order of the sequence is important,

through the use of a hidden state which keeps track of the evolution of the sequence.

A limitation of RNNs is the fact that over long sequences, they will tend to forget things due to the vanishing gradient problem. This is addressed with the introduction of LSTMs, which add an input, output and forget gate. These new gates regulate the flow of information through the network, with the forget gate handling the information which is to be kept or discarded.

The linear layer outputs a sequence of values, which is correlated with the dictionary of words. Given a sequence of words $y_s$ and the embedder $e$ for all the words $w$ in the dictionary, we can compute the hidden state $h_t^T$, and obtain the probability of the next word $y_n$ to be the next token by using the following softmax function:

$$p(y_n|y_s) = \frac{exp(h_t^T e_{y_n})}{\sum_{w \in V} exp(h_t^T e_w)} \tag{1}$$

We can then use this probability to predict the next token in the sequence by taking the most likely token (determinitic approach), or to sample from the probability distribution a word to continue the sequence (stochastic approach). This process can be repeated until an <eos> token is generated, or a certain sentence length is reached, generating a sequence following the training dataset style of Wall Street Journal.

To evaluate the performance of the model, we need to use a metric similar to accuracy; the challenge presented here is that we deal with an output of probabilities. To solve this problem, perplexity is used to evaluate the model. Perplexity can be described as how perplexed would the agent be upon seeing the sequence it is fed. In a more appropriate description, it is the metric which describes how unlikely is the model to generate the sequence that it sees. With the sequence $y_{1:end}$, perplexity is:

$$perp(y_{1:end}) = 2^{-\frac{1}{end} L(y_{1:end})} \tag{2}$$

,

where the loss function is the log likelihood:

$$L(y_{1:end}) = L(y_1, y_2, ..., y_{end}) = \sum_{t=1}^{end} log_2 p(y_t|y_{1:t-1}) \tag{3}$$

Another problem faced is the handling of batch computation. Tensors need to have the same sizes to allow matrix computations, however, tokenized sentences can end up with varying lengths. To handle this disparity, the shorter sentences are padded with end-of-sentence tokens to match the longest sentence in the batch. A mask is created, to keep track of which tokens appear in the original text, and which tokens are a result of the padding. The entire computation is handled as normal. At the computation of the negative log likelihood loss, the mask is used to remove the padded tokens from the loss, setting the gradient in this situation to 0.

# 4 Generated samples

**Example without any training (random weights):**

1. environmentalism economics rockwell modernize friday dumped u.s.a succeeded formed heat cumulative unfortunately paris writer plunge atoms guaranteed welfare noxell taiwan swap sandinista noble minutes dentsu distinct indicate psychiatric wastewater hearts declined point consolidation cutler cooperatives printing evasion imagine assessed unclear clues liquidated nor outperformed amoco admit prints conduct ballot ancient subsidies bail therefore barber aside maneuver petrochemicals slash shoulder drug-related upjohn catalog black outcome rand goupil argument infection w. insiders entrepreneurial ground california eddie remember texaco after wolf apparent calif. owns cautiously campaign production

Due to the model being initialized randomly, it has a very low chance of predicting the end of sentence token. Unless stopped, it will keep producing gibberish for a long time.

**Examples at the start of training (after 5 epochs):**

1. soviet as couples miss $< unk >$ was ease up by the fuji its deficit comments would which an governor higher i evenly $< unk >$ these $< unk >$ and N of N for the but there while ahead filing which produce it of one stock financial be $< unk >$ to bids s&p u.s. addition often columbia half gave european costs options efforts safe a paid

2. the steam meanwhile american at in look of managed mr. slowdown $< unk >$ in

3. the while the on $< unk >$ authorities out the of

4. however requirements was lined federal home said mr. changing $< unk >$ N he buses executive marketers $ the defendant

5. so rubicam platinum to officer money to $< unk >$ soo with meanwhile $ of ralph cupertino of uncertainty on on that capital excellent trump on fine ends ideas convertible sell new lined

6. kentucky reduction to a point in funds million of $< unk > < unk >$ the drop fine against reverse market parallels

7. mr. broke details $< unk >$ committee of it from to informed up program is promises for new specialists to began continue mexico by $ N datapoint interest

8. i lawyers previous so the productivity however economy from prospective interest

9. in that among the business among the thing said the stocks receive with legislators hopes georgia and as to reading was $< unk >$ healthy issues held would in N 's blood setback actually $< unk >$ disk financial such pay a 's almost case

We see that the model starts to predict the unknown token more often (as expected from the dataset), and it can cut itself off eventually. The sentences seem to be disconnected in ideas, but we get an idea of what words are popular in the text ("the while on out the of", "stock", "business", "economy").

**Examples at the end of training (after 50 epochs)**

1. that professor want to improve on november telephone is never enough to appear the wholesale funds
2. dollars according to the drug des by their issues in certain and unchanged futures via worth
3. devices ca n't refer $ N a share for tuesday 's $< unk >$
4. programs on the major river may be used to kids for the $< unk >$ and they are yesterday by it they ca n't alleges so the
5. eastern says that pleaded a leading phase for drexel bureau about
6. they $< unk >$ move $< unk >$ the concern and that should high-yield plan
7. early also rivals of $< unk >$ customers space wo n't line-item forward by eddie corp. and sustain doubled 's stock force had bill now before and pressure up N million a convertible real estate account
8. standing england will rjr say they to your owners that are believed to already 's many computer market says
9. cuba while switzerland metropolitan $< unk >$ which continues to avoid the studio in the rise students at texas u.s. and purchasing execution was

The model is able to be coherent; while sentences do not form a concrete idea, we get the feel that the model is replicating some phrases similar to expected english structures.

# 5 Conclusion

In this report, we have developed a language model capable of learning a distribution of language tokens, to create a language generator which can be used to write WSJ articles.