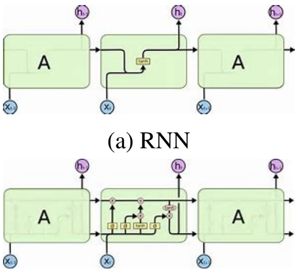
NLU Assignment 2

Rishi Hazra
Systems Engineering
14542
rishixtreme@gmail.com

Abstract

Recurrent nets are a type of artificial neural network designed to recognize patterns in sequences of data, such as text, genomes, handwriting, the spoken word, or numerical times series data emanating from sensors, stock markets and government agencies. They are arguably the most powerful and useful type of neural network, designed in form of loops which help them process the past data along with the present. The past data is known as the *state* and the present data is known as the input. This RNN can be trained to learn on a context.
However, it is practically impossible to model a huge context using an RNN. This is where LSTM or a Long-Short Type Memory cell comes in. LSTM cells have a cell-state which runs through all cells and are fed from the *forget* and *input* gates.

1 Introduction



(a) RNN

(b) LSTM

Figure 1: Basic rolled out cells

2 Problem Statement

Divide the Gutenberg corpus into train, dev, and test. Let the training split be D2-Train. Implement and build the best LM in the following setting.

Task 1: Build the best token level LSTM-based language model for the setting above.
Task 2: Build the best character level LSTM-based language model for the setting above.

3 Approach

For developing the models, *tensorflow* was used. Basic LSTM cells of size 128 was stacked into two layers, each of which contains equal number of cells. The number of cells in a layer were fixed according to the batch size of the input. For Task 1, batch size and sequence length were fixed at 20 during training. For Task 2, both were fixed at 50.

3.1 Pre-processing

For Task 1, the corpus was word tokenized and some special characters were removed from the processed set. The vocabulary is then sorted according to frequency and the words with frequency less than 2 were replaced randomly (with a probability of 0.2) into $'\langle unk \rangle'$ token for obtaining the perplexity during testing. Similarly, for character level model, the $'@'$ symbol was used to replace the less frequent characters after removing certain special characters. The data is then split into 80:20 for train and test respectively. The data is embedded into a 128 length vector before feeding them to the LSTM cells.

The train model is defined as follows:

Feature	Word LSTM	Char LSTM
unique words/characters	124888	46
number of epochs	20	20
learning rate	0.002	0.002
decay rate	0.97	0.97
sequence length	20	50
batch size	20	50
number of layers	2	2
size of LSTM	128	128

3.2 LSTM cell processing

Starting from sequential data, batchify arranges the dataset into columns. For instance, with the alphabet as the sequence and batch size 4, we'd get,

a	g	m	s
b	h	n	t
c	i	o	u
d	j	p	v
e	k	q	w
f	l	r	x

These columns are treated as independent by the model, which is the batch processing structure.

The sub-division of data into chunks is done in the following form:

a	g	m	s
b	h	n	t

The sub-division of data is not done along the batch dimension (i.e. dimension 1).The chunks are along dimension 0, corresponding to the sequence length dimension in the LSTM.

The output of the LSTM structure is in embedded form which is then passed through a dense softmax layer to get the probability distribution of the vocabulary of characters or tokens according to the model.

3.3 Test Model

The test data is again processed and embedded before prediction is done. The batch size and the sequence length in this case is set equal to 1. The number of tokens to be generated is fixed before hand.

The perplexity obtained from character model is 3.76 and from the word(token) model is

3.4 Sample

Character LSTM model: *The fortyyears old god of israel lived into a zidian11 now the princes of the rivers*

and the great continually down by the house of the lord unto the assessions of the border with the waters

Word LSTM model: *my praetors practice.sir simply practise themonstrous advantage.you adownright aery-light me.susan me.sir*

that spanned affected thoughtfulness obominable agony serapis