Machine Translation

Ankit Wadhawan - Machine Learning Nanodegree

Domain Background

Machine translation (MT) is a sub-field of computational linguistics that investigates the use of software to translate text or speech from one language to another. There have been multiple attempts to solve this problem via statistical/neural techniques in recent years, gaining success in varied degrees .

The approach I would be taking involves Deep neural machine translation. Deep NMT is an extension of neural machine translation(NMT), which is an approach to machine translation that uses a large artificial neural network to predict the likelihood of a sequence of words . Both use a large neural network with the difference that deep neural machine translation processes multiple neural network layers instead of just one.

Machine Translation using Neural Networks has multiple critical applications across wide domains, and it is currently a very resource intensive field where machines have only just started reaching human levels on accuracy. Industry leading translations provided by organisations such as Google have also started moving towards Neural Network based approaches (GNMT).

Problem Statement

To translate sentences from a given language to another. The source language I would be using is English, with the target language being French.

Dataset & Inputs

Typical productionised Language translation models are very resource intensive, with millions of sentences, words & large parallel corpuses required. This in turn requires expensive computing resources (the training needs to be done on large GPUs) & weeks of training.

To mitigate a few of these barriers, I will be using the dataset provided in the DLND language translation project by Udacity at Github, which includes a relatively small vocabulary & a limited amount of parallel sentences, hence not requiring a lot of computing power & time.

The dataset contains approximately **227** unique words in the source language (English) & **137K** sentences, with greater than **13** words on average in a sentence

Benchmark Model

The benchmark model I would be using is a character-based Sequence to Sequence Model built using Keras:
https://github.com/keras-team/keras/blob/master/examples/lstm_seq2seq.py
A Seq2Seq model is built on top of Recurrent Neural Networks(RNNs).
An RNN is an artificial neural network where connections between nodes form a directed graph along a sequence. In simple terms, this means that the output of the RNN is fed back to it. This allows it to exhibit dynamic behavior for a time sequence. Unlike feedforward neural networks where no cycles are formed, RNNs can use their internal state (memory) to process sequences of inputs.
The original design & the Benchmark model will be compared for accuracy against the trained model on the above dataset. The dataset would be divided into training & validation sets, and both models would be trained & tested with the same. The accuracy would be calculated as the mean of all valid words generated for the target sentence in French, given a source sentence in English.

Evaluation Metric

The model would be evaluated on the following metric:
Accuracy = Similarity between generated sentence & target sentence. To quantify the same mathematically using numpy, the following formula will be used:

```
Import numpy as np

accuracy = np.mean(np.equal(target_sentence,
generated_sentence) )
```

Where target_sentence & generated_sentence are both an array where the sentence is split by spaces.

Project Design

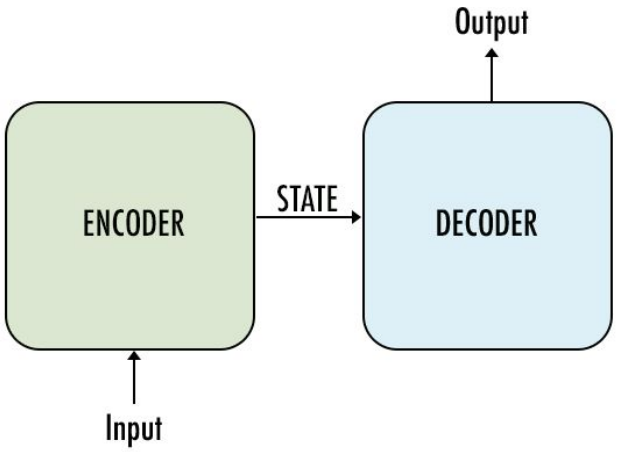
I will be using a **word - based Sequence to Sequence Model** to reach a solution to the above problem statement.

- Load the dataset mentioned above from text files as arrays of sentences.

- Preprocess data - create vocabulary lookup tables, add delimiters such as convert sentences to numerical representation that the Sequence to Sequence network can understand, as well as adding delimiters appropriately in the sentence for the network to identify the beginning, end and unknown words present in it.

- Build the Sequence to Sequence model based Neural Network:

  A typical sequence to sequence model has two parts – an **encoder** and a **decoder.** Both the parts are practically two different neural network models combined into one giant network.



Broadly, the task of an encoder network is to understand the input sequence, and create a smaller dimensional representation of it. This representation is then forwarded to a decoder network which generates a sequence of its own that represents the output.

- Train the network on a portion of the dataset.
- Validate the generated model against test data.

Resources:

https://en.wikipedia.org/wiki/Machine_translation

https://en.wikipedia.org/wiki/Recurrent_neural_network

https://github.com/udacity/deep-learning/tree/master/language-translation/data

https://github.com/keras-team/keras/blob/master/examples/lstm_seq2seq.py