Language Technology Assignment: Neural Machine Translation

Taskoudis Dimitris and Fragkouli Styliani Christina

Aristotle University of Thessaloniki

Language Technology Assignment

Juny 12th, 2020

Introduction

- One of the earliest goals for computers was the **automatic translation** of text from one language to another.
- Statistical machine translation is the use of **statistical models** that learn to make transaltions
- ”Given a sentence T in the target language, we seek the sentence S from which the translator produced T. We know that our chance of error is minimized by choosing that sentence S that is **most probable** given T. Thus, we wish to choose S so as to maximize Pr(S|T).”
- Suffered from a **narrow focus** on the phrases being translated, losing the broader nature of the target text.

Neural Machine Translation

- Use of **neural network** models to learn a statistical model for machine translation
- Single system can be trained **directly on source and target text**, no longer requiring the pipeline of specialized systems
- **end-to-end** = single model (NN)

Word Embeddings

- Word embeddings are a type of **word representation** that allows words with **similar meaning** to have a similar representation
- Each word is mapped to one **vector**
- "**distributional hypothesis**" by Zellig Harris that could be summarized as: "**words that have similar context will have similar meanings.**"

Sequence-to-sequence with LSTM

- Seq2seq learning is about training models to convert sequences from one domain to sequences in another domain
- **Canonical** seq2seq case is when input and output sequences have **different lengths**
- Addressed with an **RNN** architecture
- **Encoder**: processes the input sequence and returns its own internal state
- **Decoder**: trained to predict the next characters of the target sequence
- Key to this architecture is the ability of the model to encode the source text into an internal **fixed-length** representation called the **context vector**.

BLEU Score

- Bilingual Evaluation Understudy, is a score for **comparing** a candidate translation of text to one or more reference translations.
- compelling **benefits:**
- is **quick and inexpensive** to calculate
- is **easy** to understand
- it is **language independent**
- it correlates highly with **human evaluation** and it has been widely adopted.
- Depending on which $n - grams$ were used the corresponding score is also $BLUE - n$

Pre-Processing

- The data refer to the translation of sentences - words from **German** to **English**
- The first step: clean the data from its **original form**
- The text divided by **line** and into a **sentence**
- **Remove** all non-printable characters, all punctuation characters, any they are not alphabetical, **normalize** all Unicode characters in ASCII, the case in lower case letters
- **Reduced** the data set to the first 10,000 copies and **mixed** the samples for higher performance of the model

Create the neural machine translation I

- Matching words to integers and a separate **tokenizer** was applied to English and German sequences
- Encode each input and output sequence into **integers**
- Create a **table** for each sequence and **add zero values** to the sequences that were smaller in maximum length
- **One-hot** encode sequences
- **Decoding** the output sequence because the model must **predict** the probability of each word in the vocabulary as output

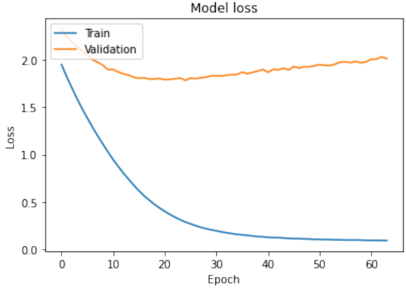
Create the neural machine translation II

- The next phase: **creation** of the model
- **Model structure**: two LSTM layers, one Embedding, one RepeatVector,one TimeDistributed and one Dense layer
- **Embedding** layer turns positive integers (indexes) into dense vectors of fixed size
- **Repeat vector** layer repeats the input n times
- **Dense** layer applied to every sample it receives as an input (the size of the English vocabulary size)
- Apply the last **LSTM** output a value for each time step in the input data
- The input sequence encoded by a **front-end** model called the encoder then decoded word by word by a **backend** model called the decoder

Model evaluation

- The **evaluation** includes two steps: create a translated output sequence and repeat the process for many input examples
- The model can **predict** the entire output sequence in a **one-shot manner**. A sequence of integers that can be enumerate and lookup in the tokenizer to map back to words
- **Repetition** for each source phrase in a set of data and **compare** the predicted result with the expected phrase target in English
- Use **BLEU score** and **loss function**
- **Results**: Train score >>Test score

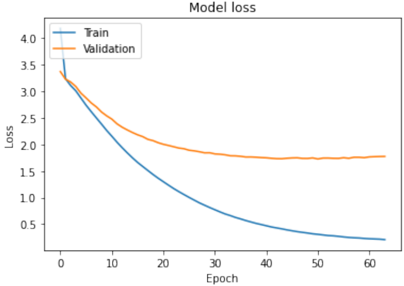
Initial model results



Training Evaluation	
BLEU-1	0.92
BLEU-2	0.88
BLEU-3	0.79
BLEU-4	0.46
Testing Evaluation	
BLEU-1	0.56
BLEU-2	0.44
BLEU-3	0.36
BLEU-4	0.16

Dropout 30percent

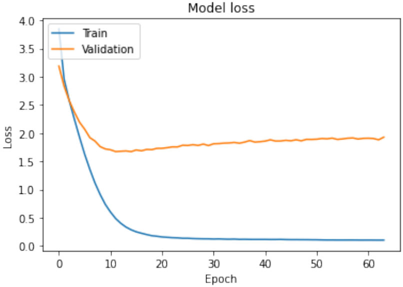
- Use of the **dropout parameter** set to 30 percent in the LSTM layers. Purpose: reduction over-fitting



Training Evaluation	
BLEU-1	0.93
BLEU-2	0.90
BLEU-3	0.81
BLEU-4	0.48
Testing Evaluation	
BLEU-1	0.58
BLEU-2	0.46
BLEU-3	0.38
BLEU-4	0.19

Bidirectional LSTM layers

- **Duplicate** the first recurrent layer in the network



Training Evaluation	
BLEU-1	0.92
BLEU-2	0.88
BLEU-3	0.80
BLEU-4	0.47
Testing Evaluation	
BLEU-1	0.60
BLEU-2	0.47
BLEU-3	0.38
BLEU-4	0.18

Thank you!