Pristine Sentence Translation: A New Approach to a Timeless Problem

Meenu Ahluwalia, Brian Coari, and Ben Brock

[1]Master of Science in Data Science
Southern Methodist University
Dallas, Texas USA
{mahluwalia, bcoari, bbrock}@smu.edu

Abstract. Translating text from one language to another is a continuous technological challenge. Although many technologies, such as Google Translate, have used machine learning and neural networks to close the translation gap, there are still many translation problems to be solved. Issues such as multiple word meanings, proper sentence structure, slang, colloquialisms, and determining the literal meaning of words vs contextual intent of those words are areas where we sometimes still see Google Translate struggle. In this paper we explore an original strategy that provides a solution to these translation issues, demonstrate a proof-of-concept of the solution, and examine the feasibility of a large-scale solution. For our translation solution we populated a database with translations of entire sentences from one language to another, instead of the words in a sentence. Since a sentence represents an entire thought instead of an assembly of words, the translation did not suffer from the issues that plague Google Translate. We also used Natural Language Processing (NLP) and predictive modeling in order to find sentences close to the sentence requested, which provides the user examples of common grammatically-correct sentences. With these approaches we were able to translate sentences that seemed impossible using traditional translation methods.

1 Introduction

The ability to easily communicate with people in another language is one of the most powerful and satisfying experiences in life. Technology has come a long way from the discovery of the Rosetta Stone in 1799, which allowed us to translate Egyptian hieroglyphics to ancient Greek in a mere 23 years. In the modern day, tools such as Google translate can be used in real time to convert between languages and allow people to connect from different cultures[1]. The latest iterations of Google Translate even use machine learning and neural nets to parse more than just single words, delivering a more satisfying user experience[2].

As far as we have come, however, the areas where we struggle are still painfully obvious. While Google Translate will usually allow you to find a bathroom and order off a menu, the intricacies and complexities of a normal, native

conversation still can cause a non-fluent speaker issues. For example, if an American coworker mentions to a Brazilian coworker about their performance on a project with "You hit one out of the park.", the Brazilian coworker could translate the words, but without familiarity with the context of a baseball game, the Brazilian would be confused and would have to ask for clarification if it was possible. It would be even harder if the Brazilian was reading a book in English with a colloquialism, since there would be no human to ask for help.

If you consider these kinds of issues from a high level they might seem unsolvable. How can you train a translation tool to look at the meaning behind sentences using on the words provided? We think we have a possible answer: an original concept we are calling Pristine Sentence Translations (PSTs). The concept of PST is that instead of translating words or phrases using neural nets and machine learning, we simply store an entire sentence in a database, and we have entire sentences in other languages that represent the meaning of that sentence.

For example, using the example above we would have an entry for the English sentence "You hit one out of the park", and we would have an entry for a Portuguese sentence mapped to that English sentence that says "Você foi ótima" which translates in English to the meaning behind the phrase: "You did great". For another example, in Portuguese there's a sentence "Eu adoro Cafuné" Google Translate does not have a translation for "Cafuné", because it's a complicated word which loosely means "the act of running fingers through hair" . Our program's goal is to return an English translation "I love the feeling of fingers running through my hair" when asked to translate "Eu adoro Cafuné" into English. Using this method there is no sentence or concept we will not be able to translate into another language given enough time and resources.

One main issue with the approach outlined above is that if we do not have an exact match for the sentence, our method return nothing. so if we tried to translate "You really hit one out of the park" from English into Portuguese we would not get any results. We decided to address this concern using Natural Language Processing (NLP) to filter out the noise in a sentence, and then use Predictive Modeling in order to find the sentence "most like" the input sentence. Using this method, "You really hit one out of the park" would ideally map most closely to "You hit one out of the park", and return the same translation: "Você foi ótima". The front-end will indicate that the translation is not for the original input sentence, instead it will indicate that it is "Showing Results for: You hit one out of the park."

Due to the strictly educational and academic nature of the project, we are not attempting to provide a full translation solution. We will limit our translations to English, Portuguese, and Hindi, and we will only provide translations for a few hundred phrases. This will be sufficient to demonstrate the appeal and power of this technique, and we will show how this solution could grow into a complete, living solution using crowdsourcing and time.

2 State of Translations

Meenu or Brian - Meenu to pick either "State of Translations" or "A New Approach to Translations"

2.1 Existing Tools and Methods

2.2 Outstanding Issues

3 A New Approach to Translations

Meenu or Brian - Meenu to pick either "State of Translations" or "A New Approach to Translations"

3.1 Pristine Sentence Translations Theory

3.2 Pristine Sentence Translations In Action

3.3 Database Design

4 Predictive Modeling

Meenu citing https://machinelearningmastery.com/develop-neural-machine-translation-system-keras/

Automatic or machine translation is one of the most challenging AI tasks given the fluidity of human language. Classically, rule-based systems were used for this task, which were replaced in the 1990s with statistical methods. More recently, deep neural network models achieve state-of-the-art results in a field that is aptly named neural machine translation.

Sequence to Sequence (often abbreviated to seq2seq) models are a special class of Recurrent Neural Network architectures typically used (but not restricted) to solve complex Language related problems like Machine Translation, Question Answering, creating Chat-bots, Text Summarization, etc. Our aim is to translate given sentence from one language to another. We will target sentence translations to and from English, Portuguese and Hindi languages only. Use of seq2seq (or Encoder-Decoder) architecture is appropriate in this case as the length of the input sequence does not has the same length as the output data

To summarize our model, the Encoder simply takes the input data, and train on it then it passes the last state of its recurrent layer as an initial state to the first recurrent layer of the decoder part. The Decoder takes the last state of encoder's last recurrent layer and uses it as an initial state to its first recurrent layer , the input of the decoder is the sequences that we want to get. We will use Keras API with Tensorflow backend to build our model.

4.1 Data Preparation

Before we start building the model, we need to clean up the text data (i.e. the sentences). We will remove all punctuation characters, normalize the case to lowercase, normalize all Unicode characters to ASCII and remove any tokens that are not alphabetic. To build the model, we need to map words to Integers. We will use Keras Tokenize class for this. The Tokenizer must be constructed and then fit on either raw text documents or integer encoded text documents. Once fit, the Tokenizer provides four attributes that you can use to understand about your text., viz.,

1. word-counts: A dictionary of words and their counts
2. word-docs: A dictionary of words and how many documents each appeared in.
3. word-index: A dictionary of words and their uniquely assigned integers.
4. document-count:An integer count of the total number of documents that were used to fit the Tokenizer.

We will also compute the vocabulary sizes and the length of maximum sequence for both the languages. We need to encode each input and output sentences to integers and pad them to the maximum phrase length to make all sentences of the same length. This is because we will use word embedding for the input sentence and one hot encoding for the output. In one hot encoding, a document is represented as a sequence of integer values, where each word in the document is represented as a unique integer. One hot encoding is needed because the model will predict the probability of each word in the vocabulary as output.

4.2 Encoder-Decoder Long Short-Term Memory (LSTM) Networks

A typical seq2seq model consists of 2 major components

1. Encoder
2. Decoder

Both these components are essentailly two different Recurrent Neural Network models combined into one giant network.

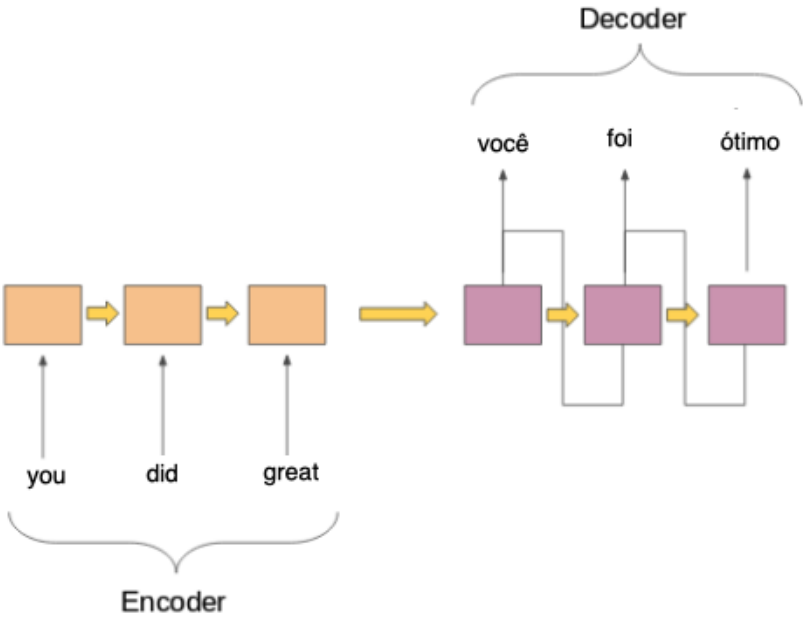


Fig. 1: Sequence to Sequence Modell

We will explain the Encoder and Decoder model in more detail.

Let's say we are trying to convert the following sentence from English to Portuguese.

Input sentence (English) - i have lost my passport Output sentence (Portuguese) - eu perdi meu passaporte

A sentence can be seen as a sequence of words or characters. We will split the sentence by words. So, for the above example in English, there are 5 words which are fed to the encoder as shown in the figure below. The input is referred to as X and Xi is the input sequence at time step i. So we have the following input. X1 = i, X2 = have, X3 = lost, X4=my, X5 = passport. Each Xi is mapped to a fixed-length vector using the built-in embedding layer of Keras API.

The LSTM will read this sentence word by word in 5 time steps as shown in the figure.

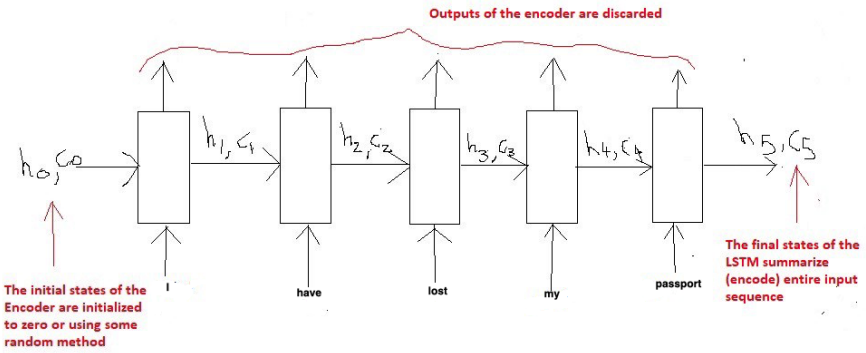


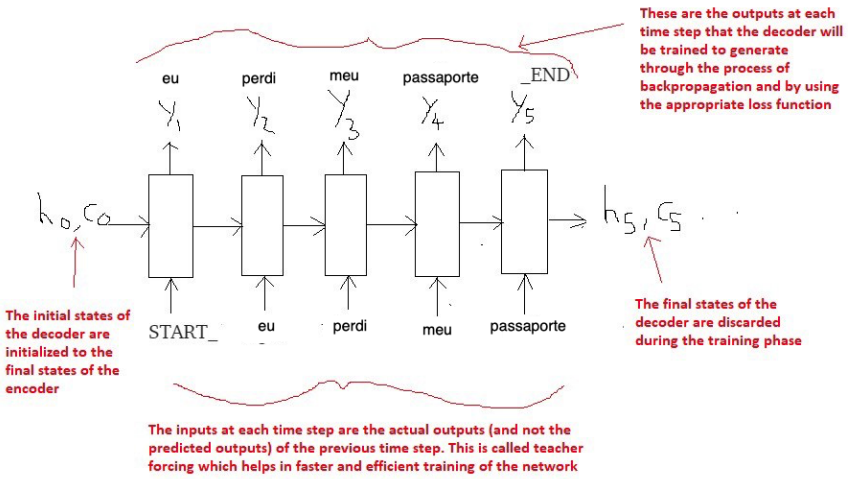
Fig. 2: Encoder LSTM

hi and ci in the figure above represent the internal state, viz., the hidden state and the cell state of the Encoder. In simple terms, they remember what LSTM has read till now. For example, h3, c3 vectors will remember that the network has read "I have lost" till now. Basically its the summary of information till time step 3 which is stored in the vectors h3 and c3 (thus called the states at time step 3). So, h5,c5 will contain the summary of the entire sentence. These states coming out of the last time step are also called as the "Thought vectors" as they summarize the entire sequence in a vector form. We initialize h0,c0 to zero as the model has not started to read the input.

Yi is the output of the LSTM at each step. We discard the outputs of the encoder and only preserve the internal states as the model has nothing to output unless it has read the entire English sentence.

Next, we define the Decoder. Unlike the Encoder LSTM which has the same role to play in both the training phase as well as in the inference phase, the Decoder LSTM has a slightly different role to play in both of these phases. Recall that given the input sentence "i have lost my passport", the goal of the decoder is to output "eu perdi meu passaporte".

The intial states (h0,c0) of the Decoder are set to the final states of the Encoder. This intuitively means that the decoder is trained to start generating the output sequence depending on the information encoded by the encoder.



[?]

Fig. 3: Decoder LSTM

4.3 Building the Neural Translation Model

We will split our dataset into train and test set for model training and evaluation, respectively. Our seq2seq model is defined as the following

1. For the encoder, we will use an embedding layer and an LSTM layer
2. For the decoder, we will use another LSTM layer followed by a dense layer

Fig. 4: Model Architechture

To be continued....

4.4 Evaluating the Neural Translation Model

5 Full Demo

Brian, but not by Friday. Maybe Sunday.

5.1 Conclusions

6 Ethical Considerations

Meenu or Brian - Meenu to pick either "Ethical Considerations" or "Conclusions and Other Work"

7 Conclusions and Other Work

Meenu or Brian - Meenu to pick either "Ethical Considerations" or "Conclusions and Other Work" Pristine Sentence Translations model is only built for about 200 sentences which could be translated from/to English, Portuguese and Hindi languages. This model could be expanded by adding more data and by incorporating more languages for translations.

One could try dropout and other forms of regularization techniques to mitigate over-fitting, or perform with hyperparamter tuning. Play with learning rate, batch-size, number of epochs etc.

It would be interesting to see how the model would perform when built using Attention.

8 References

1. Google's new translation software is powered by brainlike artificial intelligence (2016, September 27), Retrieved February 04, 2019, from https://www.sciencemag.org/news/2016/09/google-s-new-translation-software-powered-brainlike-artificial-intelligence?r3f_986=https://www.google.com/

2. Found in translation: More accurate, fluent sentences in Google Translate (2016, November 15), Retrieved February 04, 2019, from https://www.blog.google/products/translate/found-translation-more-accurate-fluent-sentences-google-translate

3. A ten-minute introduction to sequence-to-sequence learning in Keras, from https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-in-keras.html

4. figure 2 from https://cdn-images-1.medium.com/max/1600/1*37tROolA8uW7Nz2YpFsWqA.jpeg