Department of Computing

Imperial College of Science, Technology and Medicine

Nueral Machine Translation

example

Author: Anton Zhitomirsky

Contents

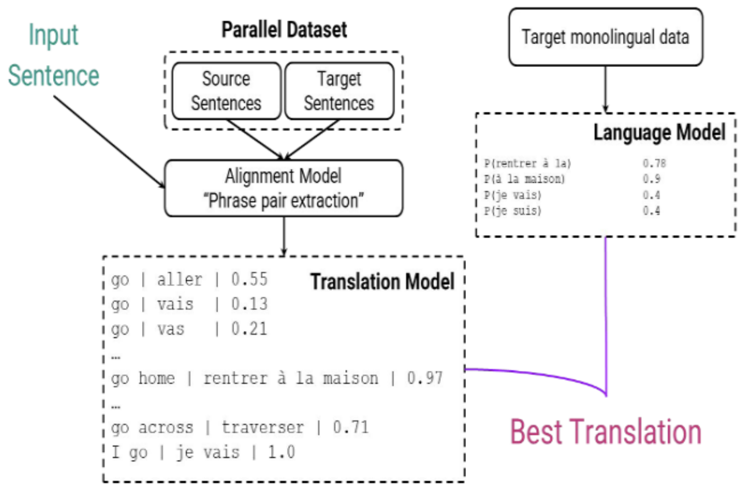
1 Machine Translation

At the highest level, given a document in source language, produce a translation in a target language.

1.1 Statistical Machine Translation

It is a pipeline of the models: Alognment model, Translation model, and Lanugage model.

Uses parallel corpa as its training set. A prallel corpa is that we have aligned sentences: i.e. an english sentence and its corresponding french sentence.



- Parallel dataset - paired sentences between two different languages (we wnat to construct correlations betwene the two sets)
- Alignment model - responsible for extracting the phrase pairs
- Translation model - lookup table, what is the porbability of the word 'go' appearing in the contex of this target. statistics over large pairs of parallel texts can help identify parallel phrases
- Corpus of target monolongual corpus - this is used to get n-grams from the data
- Language model - contains the probability of target language phrases

1.1.1 How to perform translations

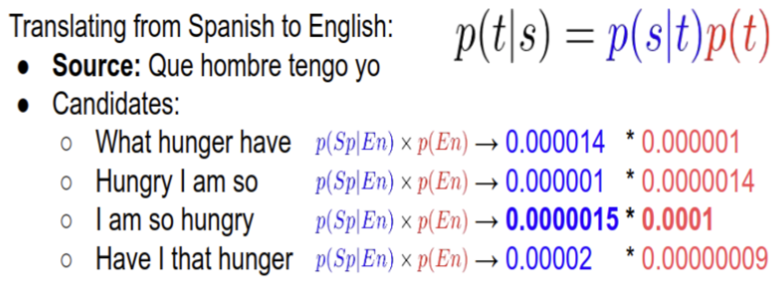
- The objective is to find $p(t|s)$; given a source sentence, s, we want target sentence, t. Language models can be trained on monolingual data[1] that gives us probability of a phrase occurring in that language.
- The translation model is the "objective" flipped. How likely is the source phrase going to occur given a cnadidate translation t. We will have multiple candidate translations.
- These probabilities/likelihoods are built from statistics of our bilingual parallel corpus.
- With this information we can apply Bayes rule to obtain $p(t|s)$.

$$p(t|s) = \frac{p(t)p(s|t)}{p(s)}$$

[1]todo

- p(s) is just a normalising factor. It doesn't contain t so is irrelevant to obtaining what we're interested in: the actual model
- The actual model is the argmaxs. Here - pick the phrase that has the highest probability of the product of the 2 terms $p(t)$ (language model) and $p(s|t)$ (translation model)

1.1.2 Example



Translating from Spanish to English:

- **Source:** Que hombre tengo yo
- Candidates:
  - What hunger have $p(Sp|En) \times p(En) \rightarrow$ 0.000014 * 0.000001
  - Hungry I am so $p(Sp|En) \times p(En) \rightarrow$ 0.000001 * 0.0000014
  - I am so hungry $p(Sp|En) \times p(En) \rightarrow$ **0.0000015 * 0.0001**
  - Have I that hunger $p(Sp|En) \times p(En) \rightarrow$ 0.00002 * 0.00000009

$$p(t|s) = p(s|t)p(t)$$

The first part is to use the trsnlation model, so given these target phrases which have been built by the alignment model and the probabilities which are in the translation modle, what is the most likely candidate, the most likely translation candidate for this given sole sentence.

The second part is the lnaguage model itself, which will take each one of these candidates and compute how likley they are to actually appear in target data.

1.1.3 Downsides

- Sentence Alignment

  In parallel corpora single sentences in one language can be translated into several sentences in the other and vice versa. Long sentences may be broken up, short sentences may be merged. There are even some languages that use writing systems without clear indication of a sentence end (for example, Thai).

- Word Alignment

  Is about finding out which words align in a source-target sentence pair

  One of the problems presented is function words that have no clear equivalent in the target language. For example, when translating from English to German the sentence ”John does not live here,” the word ”does” doesn't have a clear alignment in the translated sentence ”John wohnt hier nicht.”

- Statistical anomalies

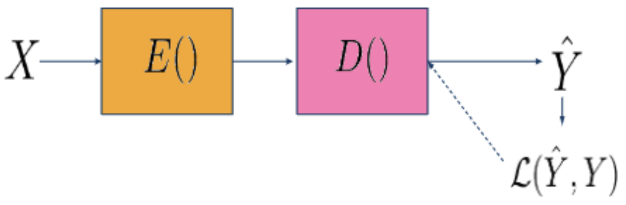
  Real-world training sets may override translations of, say, proper nouns. An example would be that ”I took the train to Berlin” gets mis-translated as ”I took the train to Paris” due to an abundance of ”train to Paris” in the training set.

- Idioms

  Only in specific contexts do we want idioms to be translated. For example, using in some bilingual corpus (in the domain of Parliment), ”hear” may almost invariably be translated to ”Bravo!” since in Parliament ”Hear, Hear!” becomes ”Bravo!”

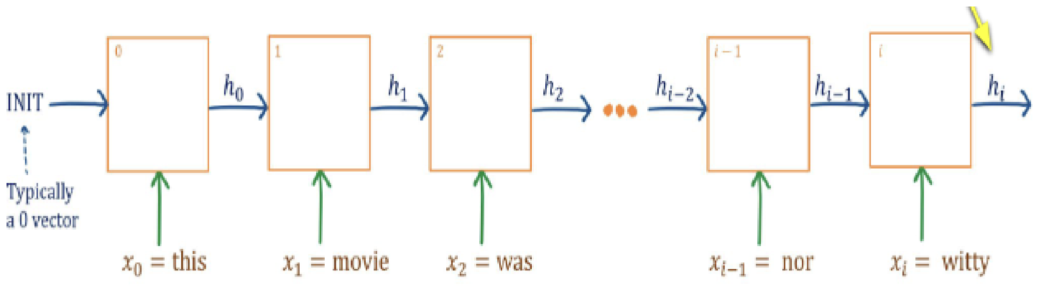
- Out of vocabulary words

1.2 Neural Machine Translation

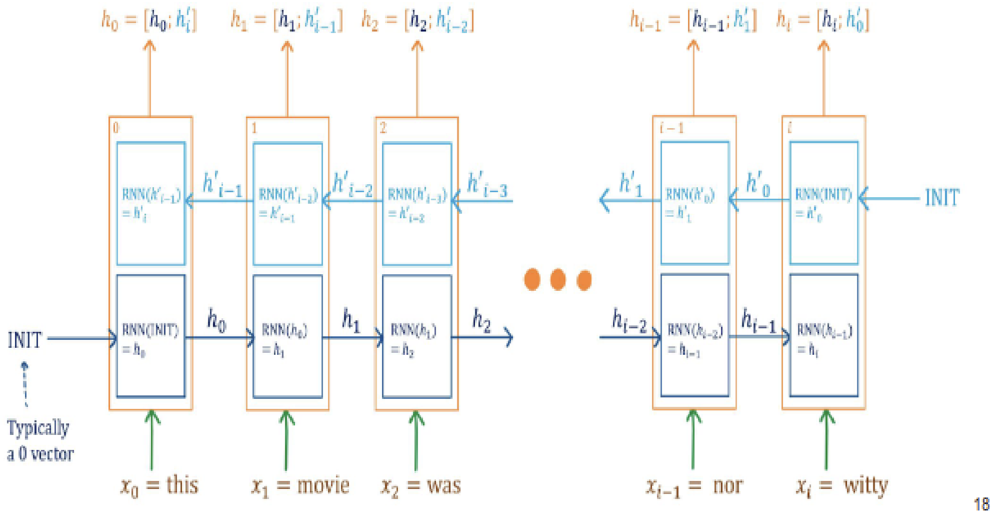


- Encoder function $E()$ - Represents the source as a latent encoding we cannot interpret
- Decoder function $D()$ - Generates a target from the latent encoding.
- Input sequence (source) $X = \{x_1, \ldots, x_i\}$
- Output sequence (target) $Y = \{y_1, \ldots, y_i\}$
- Loss over out outputs; our predicted output given our real output.

1.3 RNNs



(a) vanilla RNN

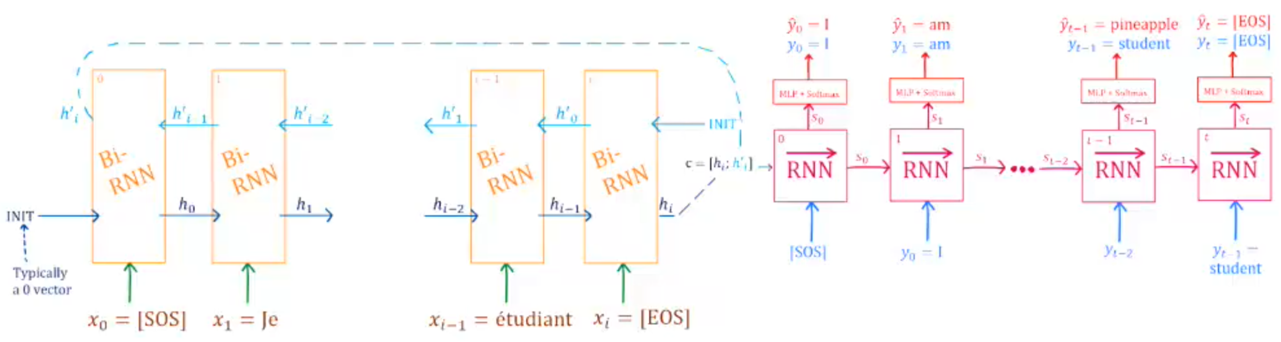


(b) bidirectional RNN

Using a BiRNN is appropriate for encoding the source as we're not trying to predict the source. We have access to the whole thing at inference time. Using a BiRNN, and reading information from both sides, allows us to gain a more information dense representation of our sequence

We can get a hint of representation for each word and hit a representation for each word. The inner representation of each word consists of the forward direction concatentated witht he backwards direction for that particular word or particular token concatenated together.

1.3.1 Naiive implementataion of machine translation



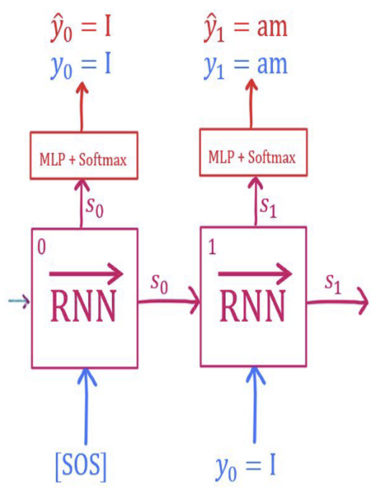
Given an input seqeunce we want to generate the output sequence. We get a c, the context vector, which we initialize our language model (the pink model, the decoder) which we initialize with the hidden state. The context comes from the encoder.

In the bidirectional RNN case, there are two ways that we can compute the context vector.

1. Simply take the lsat representations form the forward and backward direction and calculate them together
2. Average across each one of those representations adn theat can form our context vector (there is a leakage here)

The dimension of C if dimensionality of the model is d, then we either have to set s in R^{2d} or project c down to R^d.

1.3.2 Teacher Forcing



Decoder is auto-regressive only during inference

During training, we use teacher forcing. We are feeding the ground truth token ot the enxt deconder cell: S_t makes a prediciton, we calculate loss $\mathcal{L}(\hat{y}_t, y_t)$ then feed y_t to decoder cell at $t+1$.

We use teacher forcing with a **ratio**. I.e. we well teacher force ratio% of time. The ratio can be 100% (i.e. full teacher forcing), 50%, or you can even anneal it during training

If at timestamp t the model makes a wrong prediction $\hat{y}_t$ with standard autoregressive modelling, if we fed this back inot decoder at call $t + 1$ our model has (potentially) incorrect context. It will use this incorrect context to decode $\hat{y}_{t+1}$; we will likley produce another incorrect owrd. This leads to an accumulation of errors, which is difficult to optimze over.

Teacher forcing creates a phenomenon called Exposure Bias. Exposure Bias is when the data distributions the model is conditioned on vary between training and inference. In this case, if we set teacher forcing to 100%, then the model is never conditioned on its own predictions. So during inference, when the model uses the previously generated words as context, it is different than what it has seen in training.

1.4 Attention

1.5 Transformers