

# Notes

28 March 2024 08:10

## Outline:

**Lecture Plan**

Today we will:

1. Introduce a **new task**: **Machine Translation** [15 mins], which is a major use-case of
2. A **new neural architecture**: **sequence-to-sequence** [45 mins], which is improved by
3. A **new neural technique**: **attention** [20 mins]

- **Announcements**
  - Assignment 3 is due today – I hope your dependency parsers are parsing text!
  - Assignment 4 out today – covered in this lecture, you get 9 days for it (!), due Thu
    - Get started early! It's bigger and harder than the previous assignments 🤖
  - Thursday's lecture about choosing final projects

Stanford

## Machine Translation:

- Task of translating a sentence from one language (the source language) to another language (the target language)

## Pre-Neural formulation:

- Prerequisites: human translated parallel data (same data in source and target languages)

**1990s-2010s: Statistical Machine Translation**

- **Core idea**: Learn a **probabilistic model** from **data**
- Suppose we're translating French → English.
- We want to find **best English sentence**  $y$ , given **French sentence**  $x$

$$\operatorname{argmax}_y P(y|x)$$

- Use Bayes Rule to break this down into **two components** to be learned separately:

$$= \operatorname{argmax}_y \underbrace{P(x|y)}_{\text{Translation Model}} \underbrace{P(y)}_{\text{Language Model}}$$

**Translation Model**  
Models how words and phrases should be translated (*fidelity*).  
Learnt from parallel data.

**Language Model**  
Models how to write good English (*fluency*).  
Learnt from monolingual data.

- Using the alignment variable – to establish word-level correspondence between source and target language sentences

**Learning alignment for SMT**

- **Question**: How to learn translation model  $P(x|y)$  from the parallel corpus?
- Break it down further: Introduce latent  $a$  variable into the model:  $P(x, a|y)$

where  $a$  is the **alignment**, i.e. word-level correspondence between source sentence  $x$  and target sentence  $y$

Morgen fliege ich nach Kanada zur Konferenz

Tomorrow I will fly to the conference in Canada

- Issues with alignment: Alignment can be one-to-many, many-to-one. Such alignments are defined by a parameter fertility.
- Also some words from source language might not exist in target language and vice versa. e.g. Le Japon from French is Japan in English

**1990s-2010s: Statistical Machine Translation**

- SMT was a **huge research field**
- The best systems were **extremely complex**

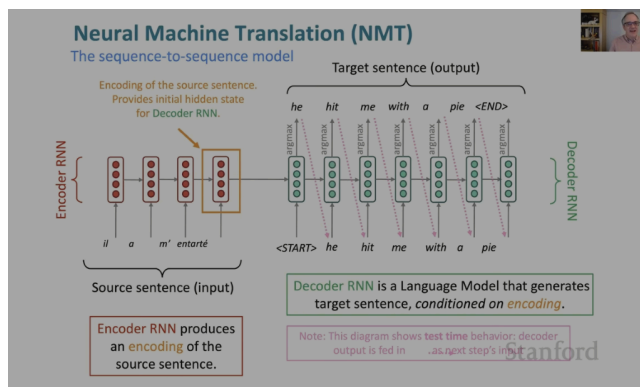
- Hundreds of important details we haven't mentioned here
- Systems had many **separately-designed subcomponents**
- Lots of **feature engineering**
  - Need to design features to capture particular language phenomena
- Require compiling and maintaining **extra resources**
  - Like tables of equivalent phrases
- Lots of **human effort** to maintain
  - Repeated effort for each language pair!

Stanford

16

## Neural Machine Translation:

- A way to do translation using a single end-to-end neural network.
- The most widely used neural network(NN) architecture is called as Sequence to Sequence model i.e. using two RNNs – one NN to encode the source sentence and another to decode the sentence in target language
- Below is an overview of seq2seq model during inference



- During training: The architecture stays same. Except the decoder is not being used in autoregressive manner but the decoder is also trained in teacher forcing manner.

## Sequence-to-sequence is versatile!

- Sequence-to-sequence is useful for **more than just MT**
- Many NLP tasks can be phrased as sequence-to-sequence:
  - **Summarization** (long text → short text)
  - **Dialogue** (previous utterances → next utterance)
  - **Parsing** (input text → output parse as sequence)
  - **Code generation** (natural language → Python code)

Stanford

## Seq2seq model is a Conditional Language Model

**Neural Machine Translation (NMT)**

- The **sequence-to-sequence** model is an example of a **Conditional Language Model**
  - **Language Model** because the decoder is predicting the next word of the target sentence  $y$
  - **Conditional** because its predictions are *also* conditioned on the source sentence  $x$
- NMT directly calculates  $P(y|x)$ :
 
$$P(y|x) = P(y_1|x) P(y_2|y_1, x) P(y_3|y_1, y_2, x) \dots P(y_T|y_1, \dots, y_{T-1}, x)$$

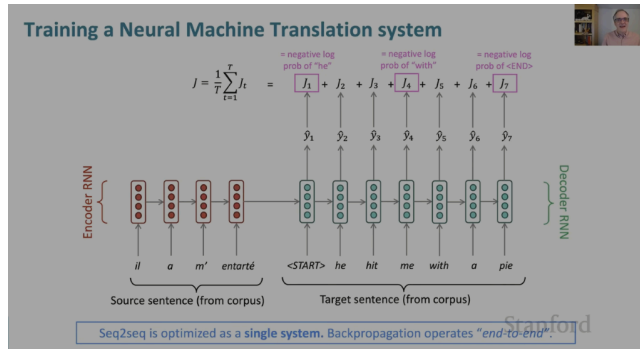
Probability of next target word, given target words so far and source sentence  $x$

Stanford

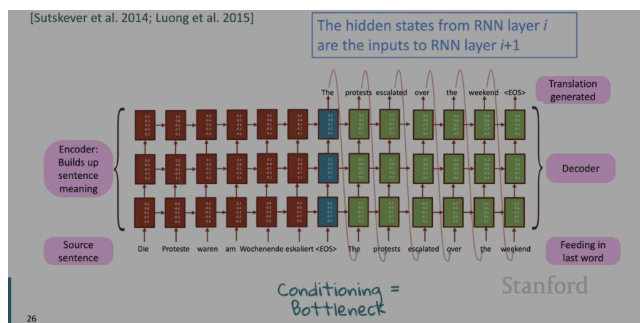
## Training of NMT systems

- (Once again) Prerequisites: human translated parallel data
- Build batches: N Source sentences + N corresponding target sentences

- Feed source sentence to encoder. Use the last hidden stage of encoder as input to decoder
- Train decoder in a teacher forcing manner by comparing decoder predictions at each time steps vs the GT target language sentence

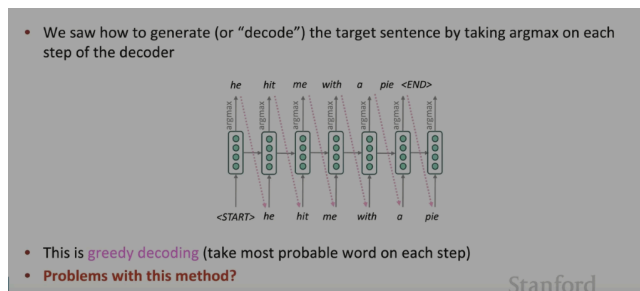


- Single encoder-Decoder RNNs did not work well in practice. Hence multiple (stacked) RNNs were used as encoder + decoder. Below is a representation of such a stacked RNN-based NMT system



## Decoding:

- Generating the words in target language using the context given by the encoder (more generally – predicting next word using encoder context)
- Until now – we take argmax at each timestep in decoder. This is called as Greedy Decoding.

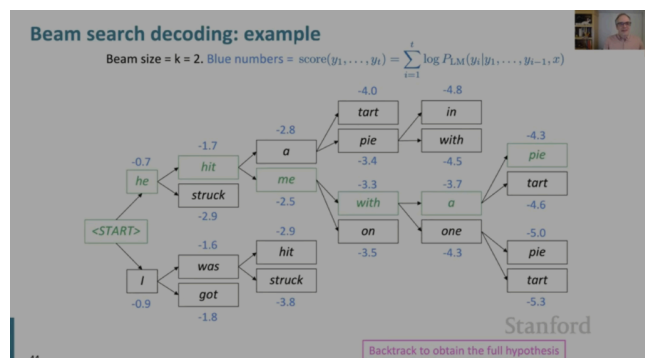


- Problems with Greedy Decoding: Settle for locally optimal prediction (word). No way to correct errors / consider global

## Beam Search:

- Retain the top-k most probable outputs at each step of decoder (outputs are called as hypothesis). Where k is the beam size
- Each hypothesis is scored by assigning a log probability
  - Hypothesis scores are negative
  - Scores are added from along the time steps as we progress
- At each time steps, num\_hypothesis=k are retained and their output is fed to the model in autoregressive manner
- The decoding processes is carried out until all hypothesis return an END token or they reach a pre-determined length
- At the end, we trace back through the tree to obtain the hypothesis with highest scores.

- Not guaranteed to produce optimal results



- Stopping Criteria:
  - Different hypothesis might produce END token at different timestamps.
  - Keep the "finished hypothesis" aside and continue with others until they produce END token or a pre-determined length is reached
- Scoring of completed hypothesis:
  - As said above: as hypothesis scores are negative – longer hypothesis will have lower scores. -- We can't just add up the local scores and then select a hypothesis
  - Normalize the scores wrt. The length of the hypothesis

## Advantages of NMT

**Advantages of NMT**

Compared to SMT, NMT has many advantages:

- Better performance
  - More fluent
  - Better use of context
  - Better use of phrase similarities
- A single neural network to be optimized end-to-end
  - No subcomponents to be individually optimized
- Requires much less human engineering effort
  - No feature engineering
  - Same method for all language pairs

Stanford

## Disadvantages of NMT

**Disadvantages of NMT?**

Compared to SMT:

- NMT is less interpretable
  - Hard to debug
- NMT is difficult to control
  - For example, can't easily specify rules or guidelines for translation
  - Safety concerns!

Stanford

## Evaluating NMT Systems

- Bilingual Evaluation Understudy
  - N-gram precision: overlap between machine translation and human translation when using one word window(1-gram), two words window (2-gram) and so on

**How do we evaluate Machine Translation?**

BLEU (Bilingual Evaluation Understudy)

- BLEU compares the machine-written translation to one or several human-written translation(s), and computes a similarity score based on:
  - n-gram precision (usually for 1, 2, 3 and 4-grams)
  - Plus a penalty for too-short system translations
- BLEU is useful but imperfect

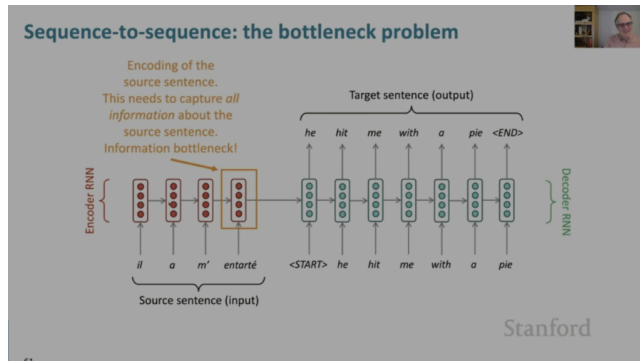
Stanford

- There are many valid ways to translate a sentence
- So a **good** translation can get a **poor** BLEU score because it has low  $n$ -gram overlap with the human translation ☹

Stanford

## Attention

- Information bottleneck in RNN Encoder: loss of information happens as we progress from one timestamp to another. I.e. not all information from prev. hidden layer (I.e. encoding of prev word) is getting propagated to the next hidden layer



- Core Idea of attention: At each stage in decoder, use a encoder signal from corresponding timestamp to focus more on a particular part of the sentence. See the below image --
- Explanation of the below image: This image shows attention score calculation for first timestamp of the decoder.
  - Assume that "START" is first timestamp in decoder. It has received the encoder feature representation (output of last block of encoder)
  - To generate the attention scores – we take dot product between each of the encoder's hidden stages and decoder's first hidden stage
  - Using softmax the attention scores are converted into probability distribution -- I.e. Attention distribution
  - Attention Output: is then a weighted average of all encoder hidden stages wrt. The attention probability distribution.
  - This attention output is then combined with the decoder's hidden stage

