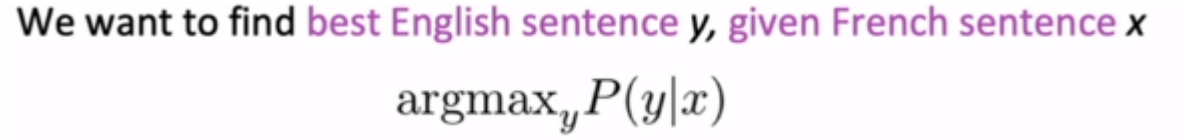
Statistical Machine Translation:

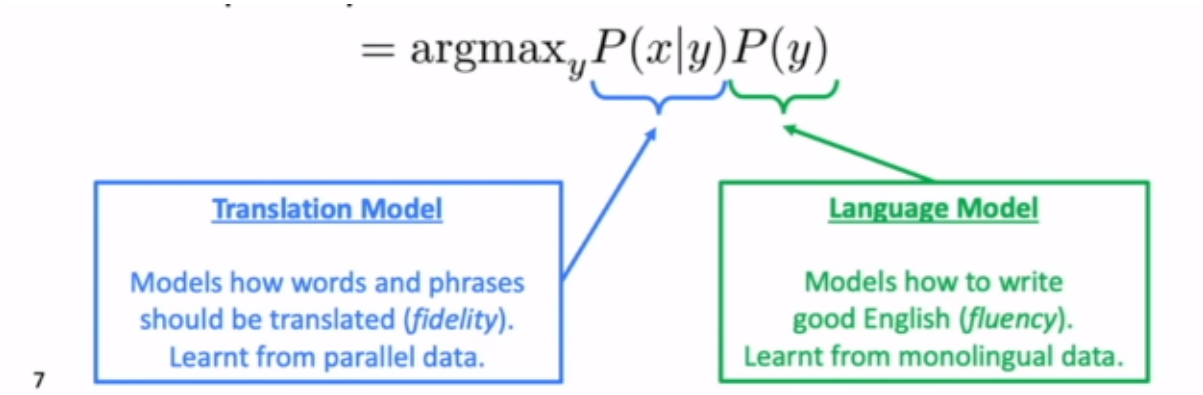
Find which sentence in language a has the most probability of existing given the sentence b in language b.



Using bayes theorem:

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

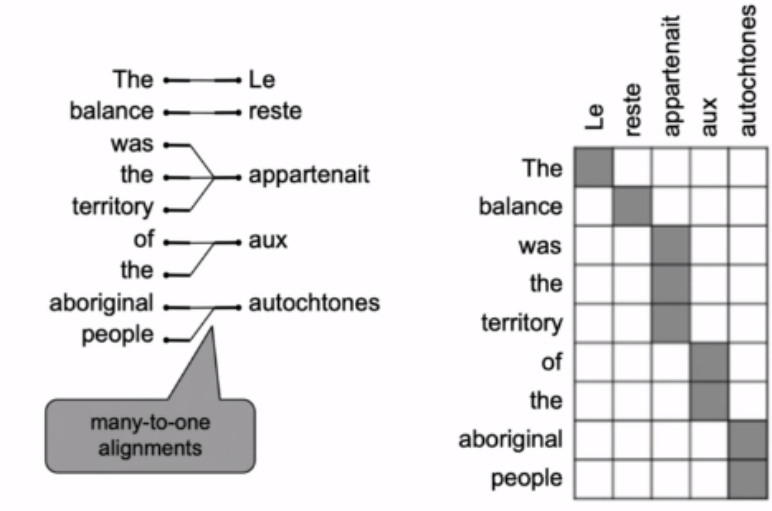
, here p(B) is the probability of sentence x occurring, which is a fixed quantity, therefore we just have to find:

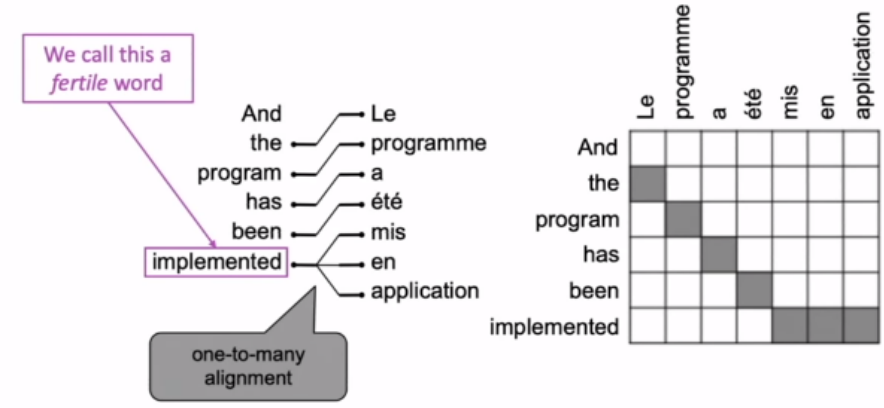


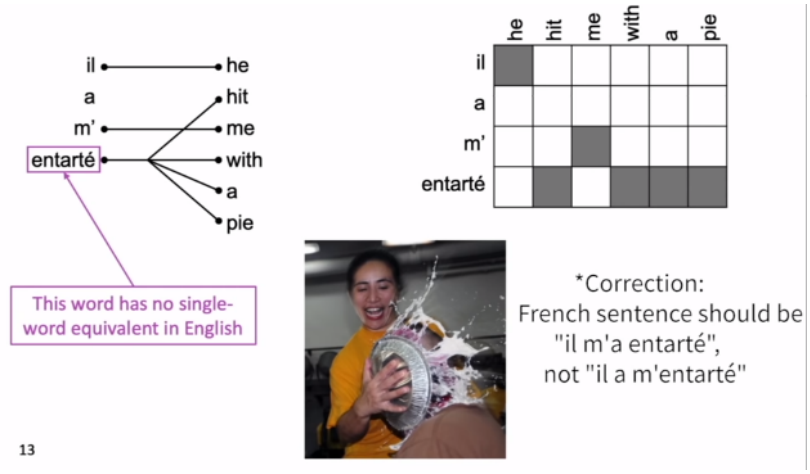
This helps us to achieve division of labour and allow two models do the work.

Alignment:

When we calculate P(x|y) what we actually want to consider is P(X,a|Y) where a is the alignment.
Examples of alignment:



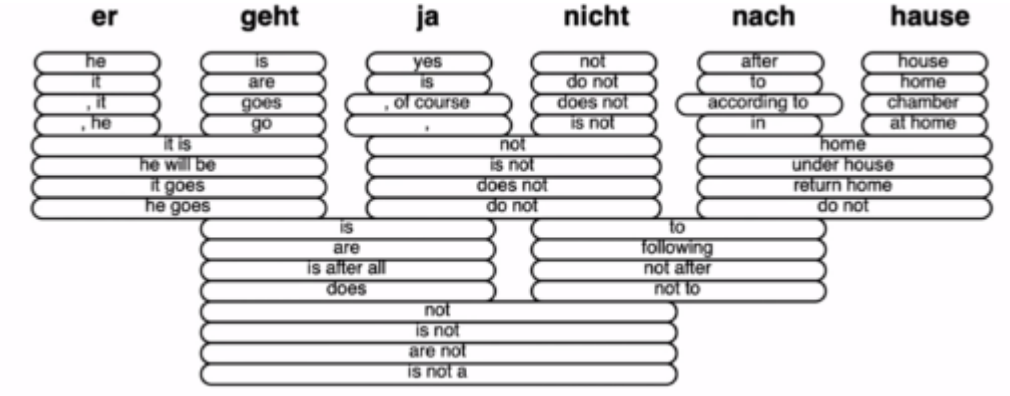




While learning P(X,a | y) we also learn what is the probability of two words coming together and what is the probability that a certain word has n as its fertility.

The calculation of P(X|y)P(y) over all the cases is obviously very expensive so what we do is use a heuristic search algorithm.

Heuristic search algorithm:

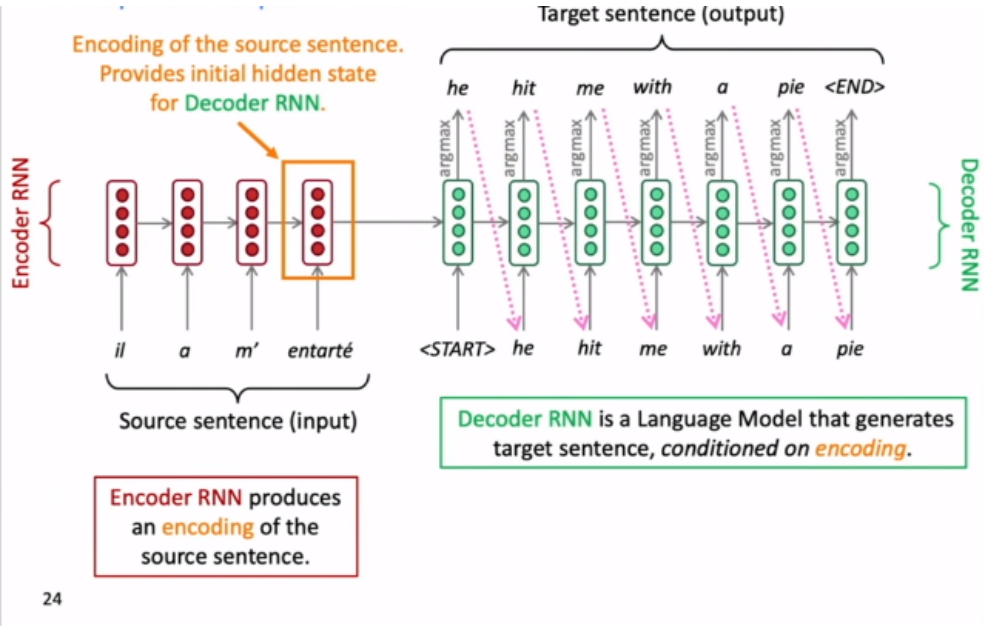


Disadvantages of statistical machine translation:

- The best systems were extremely complex
- Lot of feature engineering was needed to capture particular features of a language
- Requires a lot of extra resources.
- All of this had to be made and maintained for all the combinations of language pairs.

Neural Machine Translation:

NMT uses a single neural network that consists of two RNNs. One to encode the piece to be translated into a vector and another that decodes this in the second language. This two RNN structure is called **SeqtoSeq**.



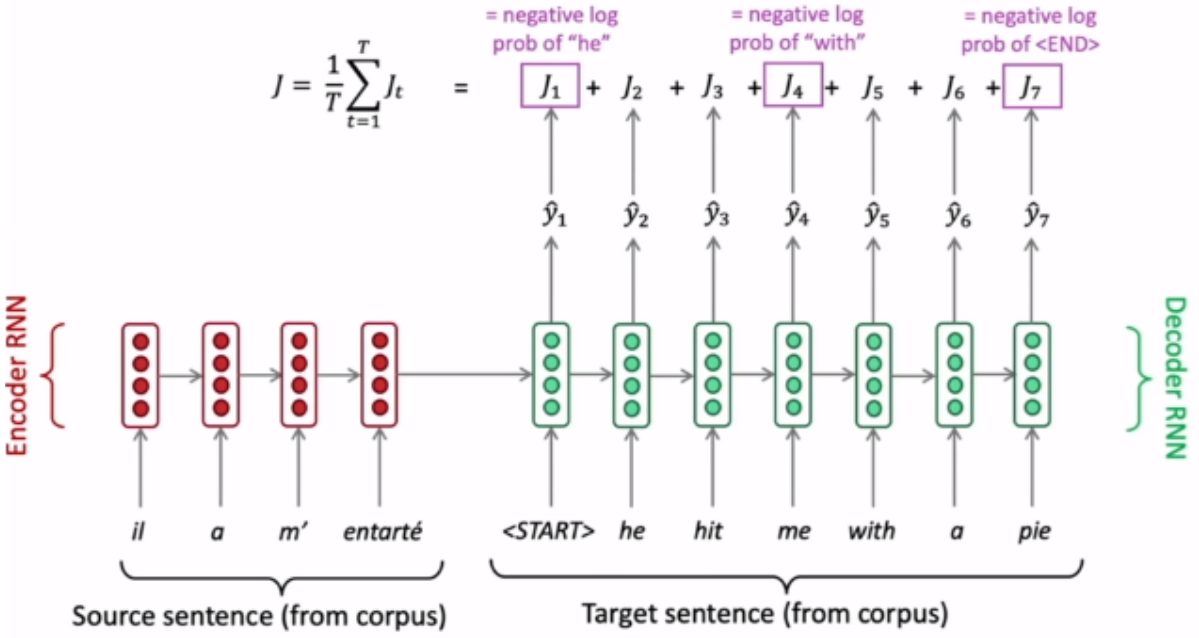
Other tasks that use SeqtoSeq include summarization, parsing, Dialogue writing (previous dialogue in, next out), Code generation (basically translating english to c code) etc.

NMT SeqtoSeq is a conditional Language Model as its condition is the input sentence X.

- **NMT directly calculates $P(y|x)$:**

$$P(y|x) = P(y_1|x)\,P(y_2|y_1,x)\,P(y_3|y_1,y_2,x)\ldots\underbrace{P(y_T|y_1,\ldots,y_{T-1},x)}_{\text{Probability of next target word, given target words so far and source sentence } x}$$

Training this NMT:

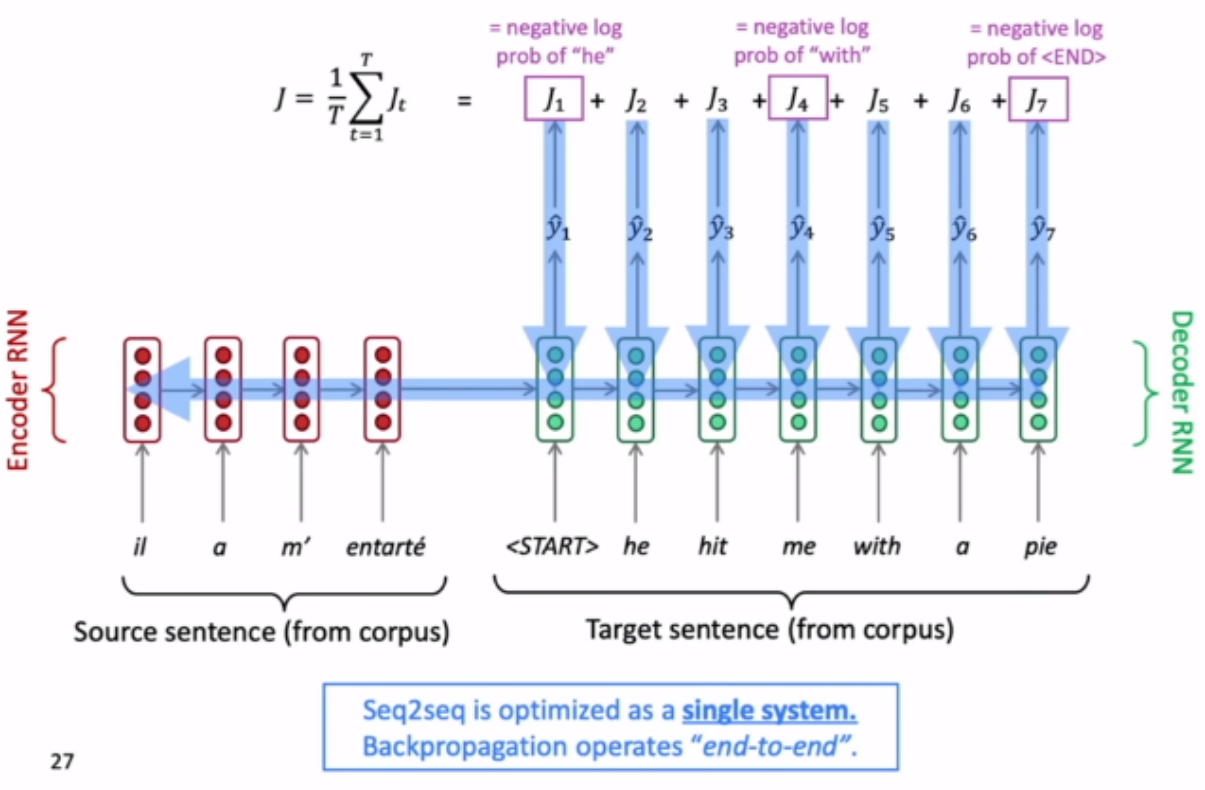


27

While training this SeqtoSeq we feed in both the sentence of language a to encoder and the corresponding sentence of language b along with a start token as tokens to decoder and calculate the cross-entropy / negative log likelihood cost in each timestamp of decode and average it.

Notice the difference in this between the training and test. While testing we test with the output of the previous timestamp as input token to the next step of the decoder but on the other hand while training, the inputs / tokens of the decoder are the words from corpus and then we calculate the loss.

The updation is done through end to end in backprop.



Methods of decoding:

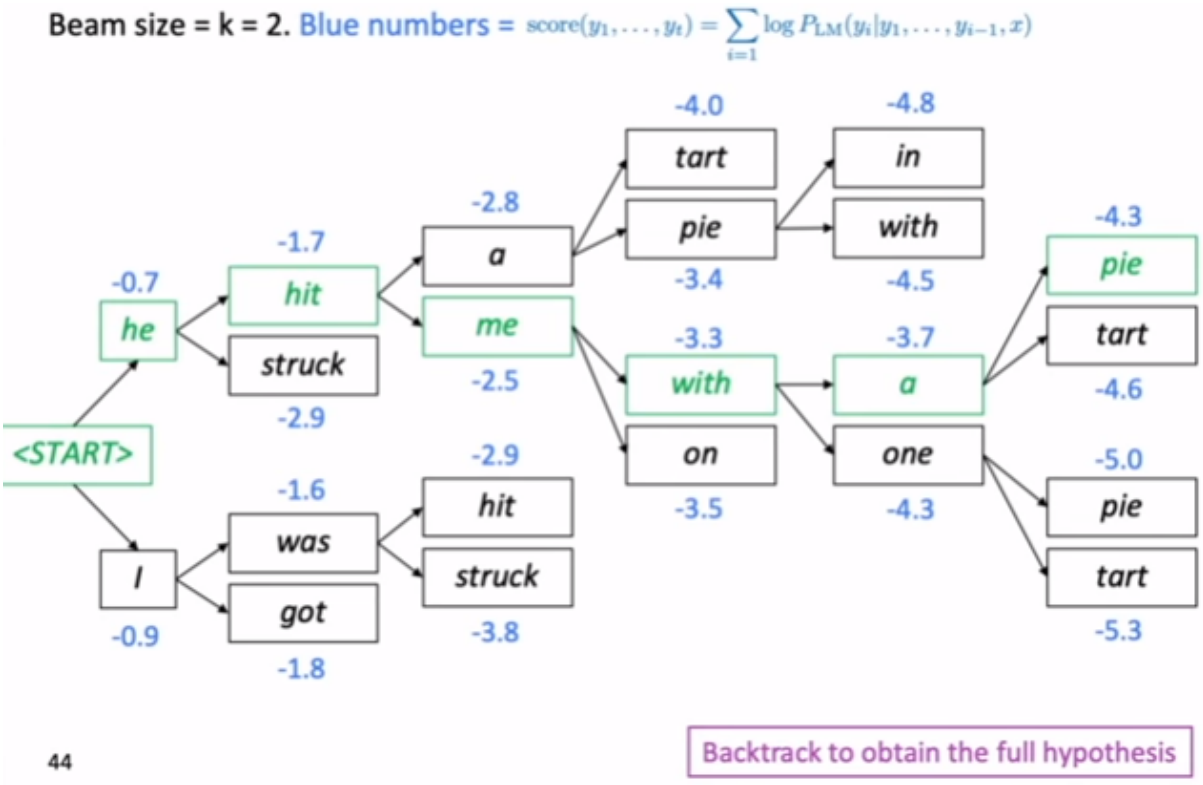
Greedy decoding:

This is the process of decoding where we decode taking the case (word in this case) which has the maximum probability.The problem with this that we can select a word that has most probability now but on considering the upcoming next few words we could easily have the case where choosing some other alternate word will give a result that makes more sense.

- Greedy decoding has no way to undo decisions!
  - Input: *il a m'entarté* *(he hit me with a pie)*
  - → *he* ____
  - → *he hit* ____
  - → *he hit a* ____ (whoops! no going back now...)

Beam Search Decoding:

In this method of decoding we choose k top cases and then we progress with all the k branches but before dividing into k branches again what we do is cut off all the branches which have a low “score”.



On top of this, what we do is divide the score by number of steps so that the longer sentences don’t automatically have lower scores.

We do this until we have end token in all the selected branches or we cut it off in t timestamps and choose the one with the lowest score among the ones that have ended.

Advantages of NMT:

- NMT is better with fluency and context.
- NMT is better with phrase similarities.
- Requires lesser human work.
- Same method b/w languages, only different corpuses.

The disadvantage of NMT is that the model is less interpretable when compared to SMT. The second disadvantage is to overlay rules like if we encounter a word always translate it to this word.

So is Machine Translation solved?

- **Nope!**
- Using common sense is still hard

English ▾ Spanish ▾

paper jam Edit

Mermelada de papel

Open in Google Translate Feedback

?

Malay - detected ▾ English ▾

Dia bekerja sebagai jururawat.
Dia bekerja sebagai pengaturcara. Edit

She works as a nurse.
He works as a programmer.

Didn't specify gender

Somali ▾ English ▾

Translate from Irish

ag ag ag ag ag ag ag ag ag ag ag ag ag
ag ag ag ag ag ag ag ag ag ag ag ag ag
ag Edit

As the name of the LORD was written in the Hebrew language, it was written in the language of the Hebrew Nation

Open in Google Translate Feedback

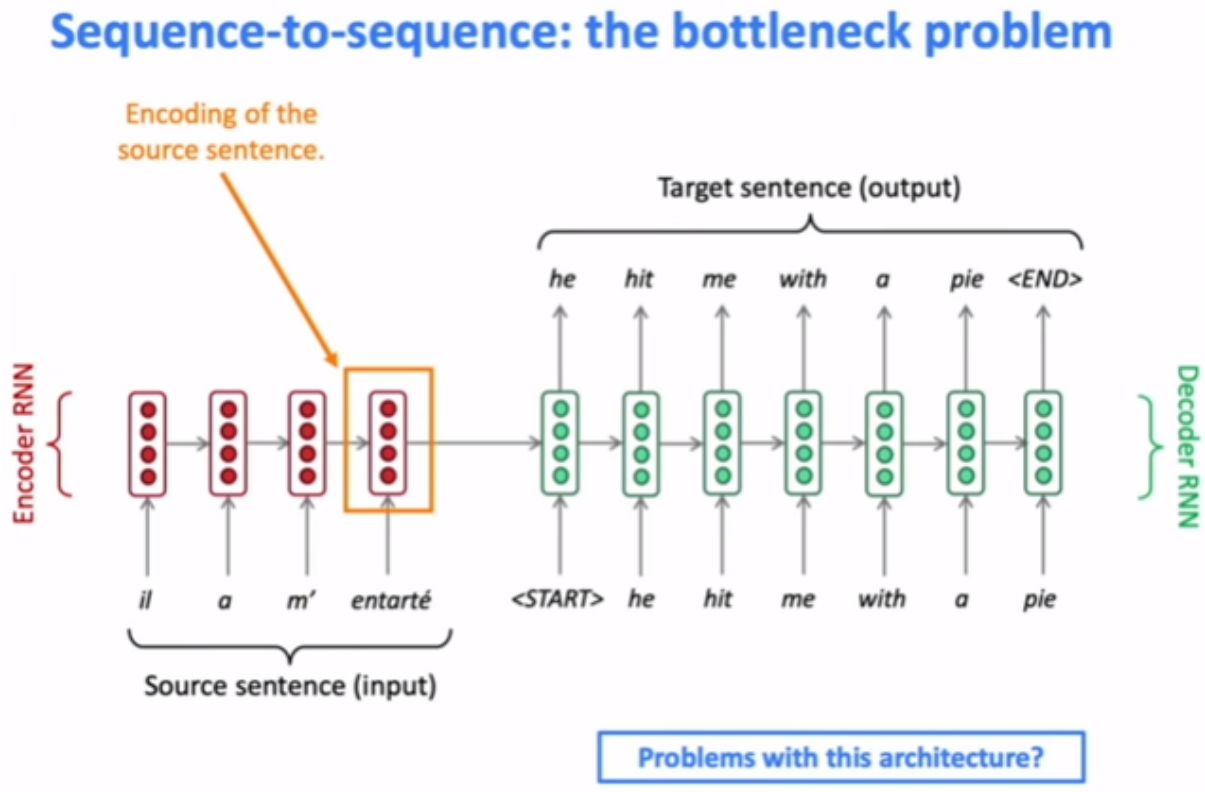
This shows how the copus affects the translation.

BLEU (BiLingual Evaluation Understudy):

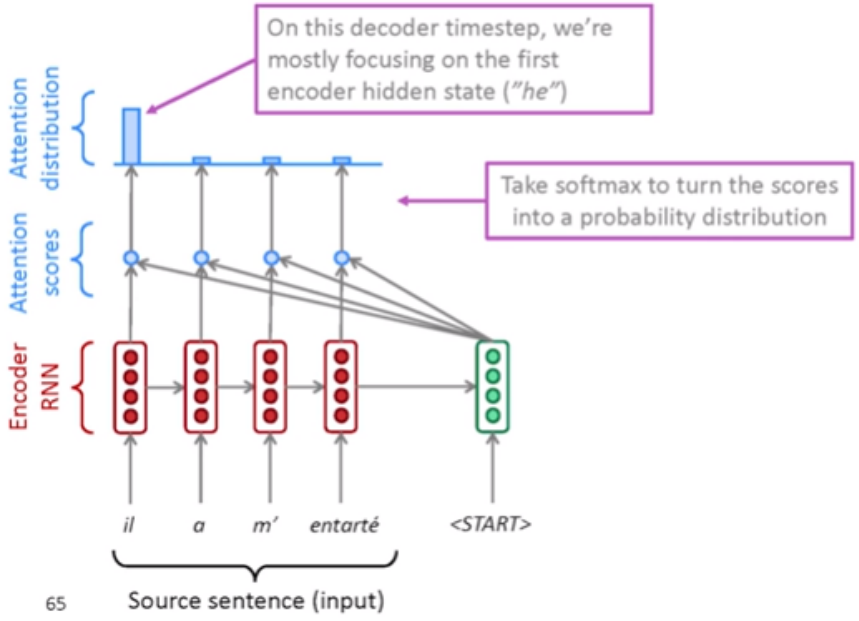
It is an algorithm used to calculate the precision of a machine translation using human translation.
Note: there is a penalty for very short sentences as it probably comes with loss in information.

BLEU is useful but imperfect as it does not accept synonyms.

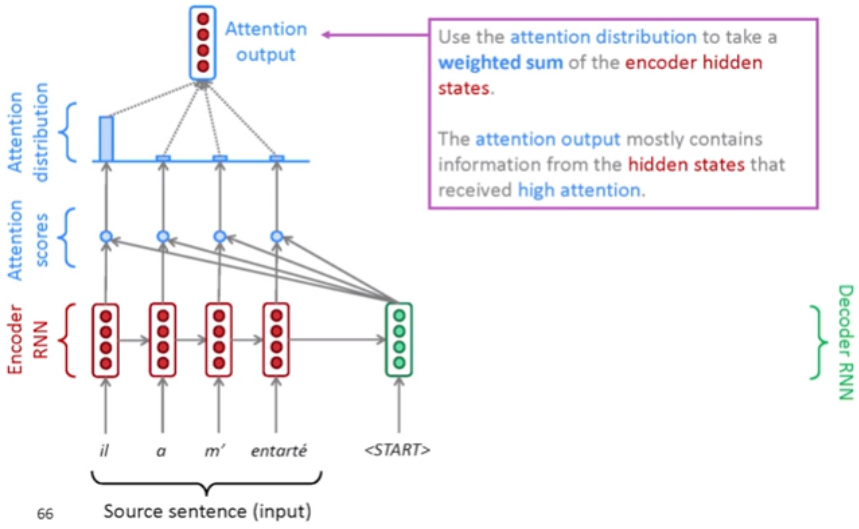
ATTENTION in NMT:



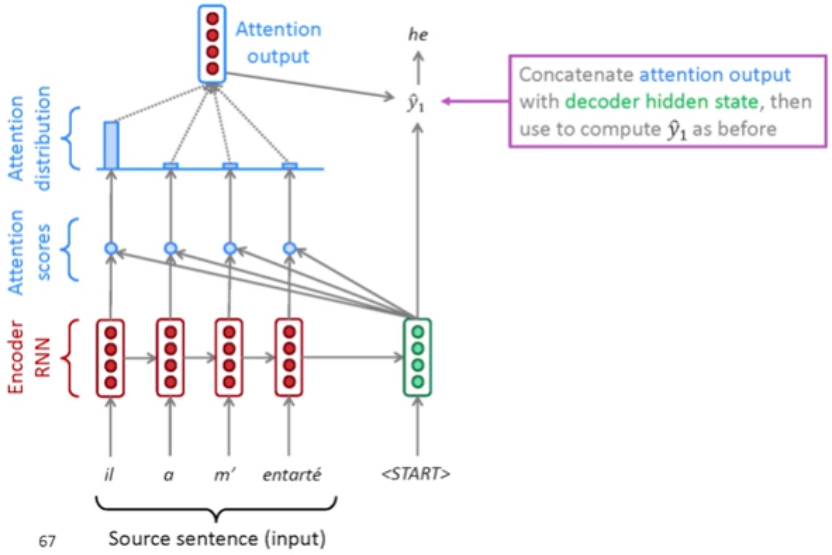
By looking at the above image we can say that there can be an info bottleneck, the whole input sentence should be contained by one vector.
To battle this we brought in the concept of attention.


On this decoder timestep, we're mostly focusing on the first encoder hidden state ("he")
Take softmax to turn the scores into a probability distribution
Attention distribution
Attention scores
Encoder RNN
il
a
m'
entarté
<START>
Source sentence (input)

65

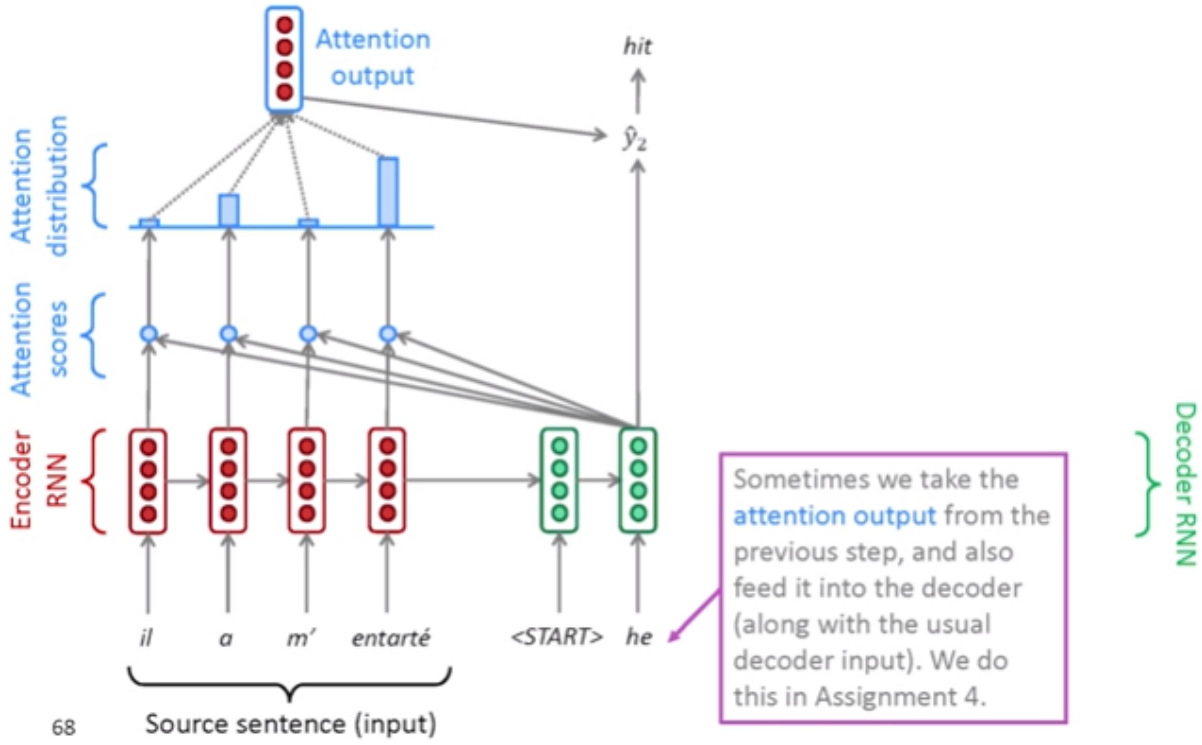

Attention output
Use the attention distribution to take a weighted sum of the encoder hidden states.
The attention output mostly contains information from the hidden states that received high attention.
Attention distribution
Attention scores
Encoder RNN
Decoder RNN
il
a
m'
entarté
<START>
Source sentence (input)

66


Attention output
he
$\hat{y}_1$
Concatenate attention output with decoder hidden state, then use to compute $\hat{y}_1$ as before
Attention distribution
Attention scores
Encoder RNN
il
a
m'
entarté
<START>
Source sentence (input)

67

Now, do the same for the next words !



68

Attention: in equations

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores e^t for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

- We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

73

Advantages of attention:

- It solves the bottleneck problem.
- It helps the vanishing gradient problem as backprop propagates from the decoder to the encoder.
- It also gives us some interpretability by representing alignment.

Note: attention is a more general concept that can be applied to many models.

Attention is a *general* Deep Learning technique

More general definition of attention:
Given a set of vector *values*, and a vector *query*, attention is a technique to compute a weighted sum of the values, dependent on the query.

Intuition:

- The weighted sum is a *selective summary* of the information contained in the values, where the query determines which values to focus on.
- Attention is a way to obtain a *fixed-size representation of an arbitrary set of representations* (the values), dependent on some other representation (the query).

There are *several* attention variants

- We have some *values* $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and a *query* $\boldsymbol{s} \in \mathbb{R}^{d_2}$
- Attention always involves:
  1. Computing the *attention scores* $\boldsymbol{e} \in \mathbb{R}^N$ ← There are multiple ways to do this
  2. Taking softmax to get *attention distribution* α:

$$\alpha = \text{softmax}(\boldsymbol{e}) \in \mathbb{R}^N$$

  3. Using attention distribution to take weighted sum of values:

$$\boldsymbol{a} = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^{d_1}$$

thus obtaining the *attention output* $\boldsymbol{a}$ (sometimes called the *context vector*)

77

Attention variants

There are several ways you can compute $\boldsymbol{e} \in \mathbb{R}^N$ from $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and $\boldsymbol{s} \in \mathbb{R}^{d_2}$:

- Basic dot-product attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$
  - Note: this assumes $d_1 = d_2$
  - This is the version we saw earlier
- Multiplicative attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$
  - Where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d_1}$ is a weight matrix
- Additive attention: $\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$
  - Where $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times d_1}, \boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times d_2}$ are weight matrices and $\boldsymbol{v} \in \mathbb{R}^{d_3}$ is a weight vector.
  - d_3 (the attention dimensionality) is a hyperparameter