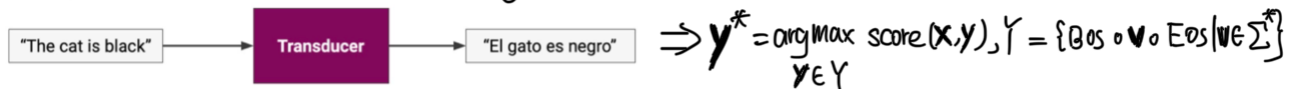


10. Machine Translation with Transformers

Translation

Translation is the process of mapping (translating) a sequence in a source language into a semantically preserving sequence in a target language. Machine translation is when a program performs such a mapping.



- y^* is the string in a target language
 - x is the input string
- } we want to find y^* that maximize some scoring function conditioned on input string x

For a translator, there are two frameworks to perform the task:

- Statistical Machine Translation (SMT)
 - Neural Machine Translation (NMT)
 - Modeling: creating a scoring function
 - Decoding: generating strings according to this scoring function.
- } Difficult. Need to use Models and algorithms.

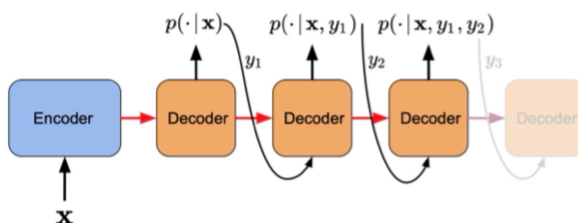
Sequence-to-sequence Models

Problem: model the probability distribution $p(y|x)$ over all strings $y \in Y$ for some sentence x , i.e., what is the most likely translation y of string x .

- $z = \operatorname{encoder}(x)$
- $y|x \sim \operatorname{decoder}(z)$

Then we estimate the parameters of our model by maximizing the log-likelihood of our data under the model p .

$$\operatorname{argmax}_{\theta} \sum_{i=1}^N \log p(y^{(i)} | x^{(i)}; \theta) = \operatorname{argmax}_{\theta} \sum_{i=1}^N \sum_{t=1}^{|y^{(i)}|} \log p(y_t^{(i)} | x^{(i)}, y_{<t}^{(i)}; \theta)$$



* Under the standard Recurrent Neural Network paradigm, the decoder only ever receives a single vector from the previous step. So it creates information bottleneck.

The Attention Mechanism

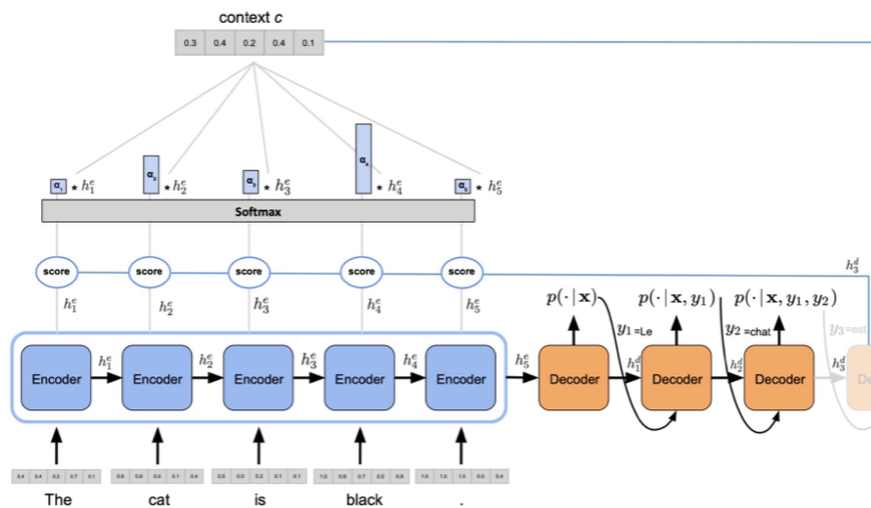
In a nutshell: Attention is a soft version of a hash table.

The attention mechanism enables a model to "attend" to information from different time steps by taking a convex combination of a model's states to build the context vector. It can be formalized as:

$$c = \operatorname{softmax}(\operatorname{score}(q, k))v,$$

where q is the query, k the key, v the values and c the resulting context

where q is the query, K the keys, V the values, and c the resulting context.



* We can denote it in another form:

$$a = \text{softmax}(\text{score}(q, K))$$

$$c = aV$$

- $k_i = v_i = h_i^{(e)}$ hidden state

- $q_t = h_t^{(d)}$ decoder at t

- $c_t = aV$ context at t

- $K = V = H^{(e)}$ vertically stacked encoder-vector representations

Variations of Attention Mechanisms

	Cross-attention	Self-attention (encoder)	Self-attention (decoder)
Query, keys and values without projection	$q_t = h_t^d$ $k_i = v_i = h_i^e, i \in 1 \dots n$ $K = V = H^e$	$q_t = h_t^e$ $k_i = v_i = h_i^e, i \in 1 \dots n$ $K = V = H^e$	$q_t = h_t^d$ $k_i = v_i = h_i^d, i \in 1 \dots m$ $K = V = H^d$
Query, keys and values with linear projection $W_q, W_k, W_v \in \mathbb{R}^{d \times d}$	$q_t = h_t^d * W_q^d$ $k_i = h_i^e * W_k^e, i \in 1 \dots n$ $v_i = h_i^e * W_v^e, i \in 1 \dots n$ $K = H^e * W_k^e$ $V = H^e * W_v^e$	$q_t = h_t^e * W_q^e$ $k_i = h_i^e * W_k^e, i \in 1 \dots n$ $v_i = h_i^e * W_v^e, i \in 1 \dots n$ $K = H^e * W_k^e$ $V = H^e * W_v^e$	$q_t = h_t^d * W_q^d$ $k_i = h_i^d * W_k^d, i \in 1 \dots m$ $v_i = h_i^d * W_v^d, i \in 1 \dots m$ $K = H^d * W_k^d$ $V = H^d * W_v^d$

- n is the length of the input sequence
- m the length of the output sequence
- the subscript $\in \{e, d\}$ specifies whether the hidden state is from the encoder or decoder
- the capital+bold notation represents horizontally stacked vectors
- $q_t, k_i, v_i, h_i^e, h_t^d \in \mathbb{R}^{1 \times d}$
- $H^e \in \mathbb{R}^{n \times d}, H^d \in \mathbb{R}^{m \times d}$

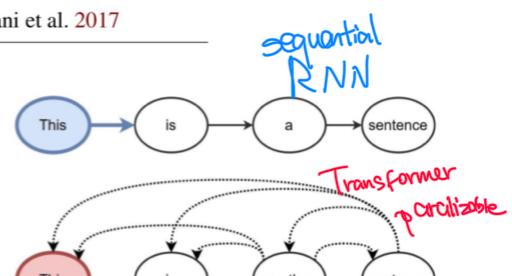
Score Functions for the Attention Mechanism

Name	Score function	Citation
Content-base attention	$\text{score}(q_t, k_i) = \cos(q_t, k_i)$	Graves, Wayne, and Danihelka 2014
Additive	$\text{score}(q_t, k_i) = v_a^T \tanh(W_a(q_t, k_i))$	Bahdanau, Cho, and Bengio 2016
Location-Base	$\text{score}(q_t, k_i) = \text{softmax}(W_a q_t)$	Luong, Pham, and Manning 2015
General	$\text{score}(q_t, k_i) = q_t^T W_a k_i$	Luong, Pham, and Manning 2015
Dot-Product	$\text{score}(q_t, k_i) = q_t^T k_i$	Luong, Pham, and Manning 2015
Scaled Dot-Product	$\text{score}(q_t, k_i) = \frac{q_t^T k_i}{\sqrt{n}}$	Vaswani et al. 2017

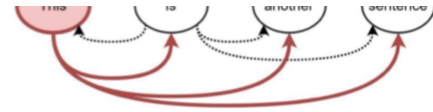
The Transformer

RNNs vs. Transformer

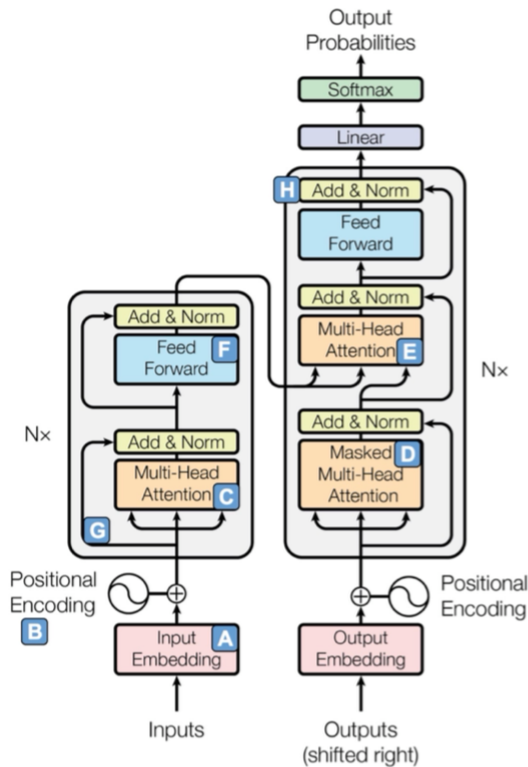
Transformers are neural networks whose architecture is highly dependent on the attention mechanism. The architecture



does away with the (slow) recurrence required by RNN encoders.



Transformer Architecture

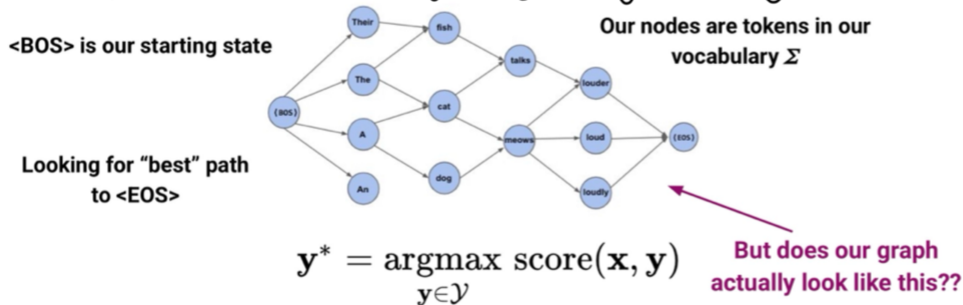


- **A: Word embeddings**, e.g., one-hot encoded words into dense representations, where semantically similar words are clustered.
- **B: positional encoding**, Vectors of the same dimension as input embeddings and encode a specific location within the sequence. Can be learned or fixed.
- **C: self-attention**, one of the versions of attentions. Queries Q , keys K , and values V are all based on the hidden representations produced by the encoder.
- **D: masked self-attention**, follows the same paradigm as standard self-attention. However, in the decoder, positions after the current time step cannot be attended when modeling $p(y_{t+1} | x, y_{<t})$. Used in decoder.
- **E: cross-attention**, allows the decoder to attend encoder.
- **F: feed-forward layers**, linear projections followed by non-linearities.

- **G: residual connection**, in both encoder and decoder, passes an input to a subsequent layer without applying a transformation. A solution to vanishing gradients.
- **H: layer normalization**, normalizes individual inputs to a layer such that collectively the elements in the input have mean 0 and variance 1. Helpful with numerical stability and convergence rate.

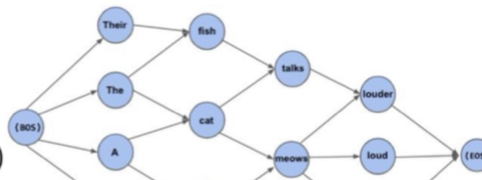
Decoding

Decoding is the process of generating strings according to predictions from a model.



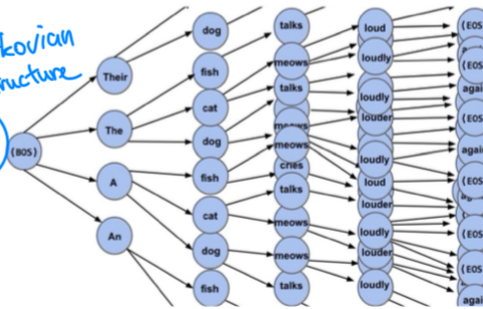
The resulting search graphs:

$$\text{score}(x, y) = \sum_{t=1}^{|y|} \log p(y_t | x, y_{t-1})$$



$$O(|\Sigma| \cdot n_{\max})$$

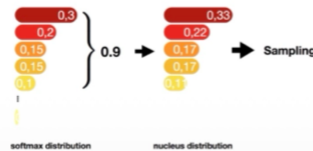
non-markovian structure


$$O(|\Sigma|^{n_{\max}}) !!!$$

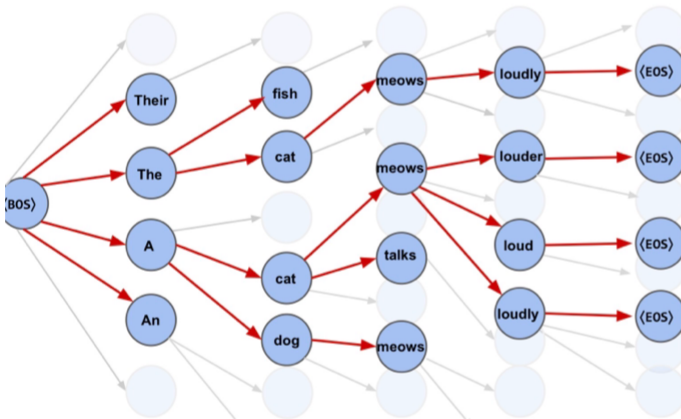
Deterministic Methods:

-

- Ancestral Sampling
- Nucleus/top-k Sampling
- Stochastic Beam Search
-

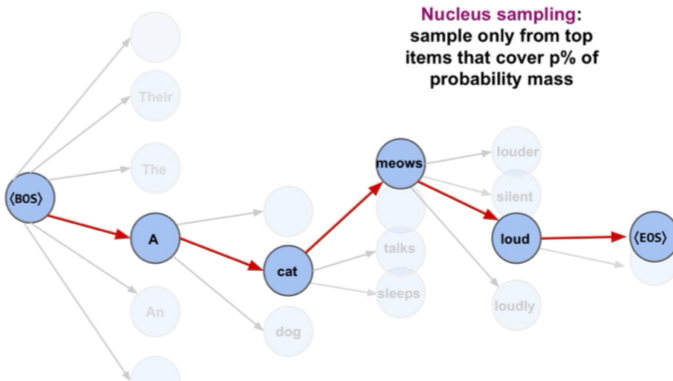


In short: pruned breadth-first search where the breadth is limited to size k



despite lack of
formal guarantees,
does well in practice

In short: sample according to the conditional distribution at each time step.



sampling from tail of distribution can lead to strange outputs

Evaluating Translations

Evaluation: Given the many different ways to say the same thing, how do we score our model's predictions?

- Using the log-likelihood of the test data is not reasonable in this scenario
- Need a holistic metric that reflects how much of the content in our source string is reflected in our predicted string

BLEU (Papineni et. al 2002): what fraction of the n -grams in the model prediction appear in the reference translation? **Formally:** "averaged" n -gram precision of the translation w.r.t. the reference for $n=1,2,3,4$ plus a brevity penalty

METEOR (Banerjee et. al 2005): combination of recall and precision of the translation w.r.t. the reference. **Formally:** weighted F-measure of where precision is weighted more highly

Note: while these metrics correlate with human judgments of quality, they are by no means perfect!