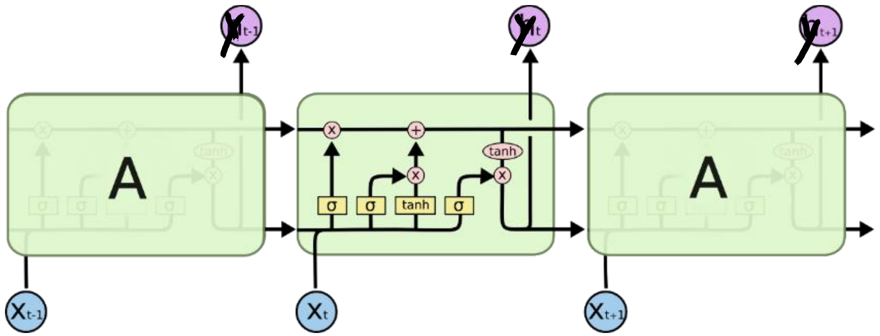
# CPSC 440: Machine Learning

Attention and Transformers

Winter 2022

# Last Time: LSTMs and Multi-Modal Learning

- We discussed long short term memory (LSTM) models:
  – RNNs with memory cells designed to remember information longer.



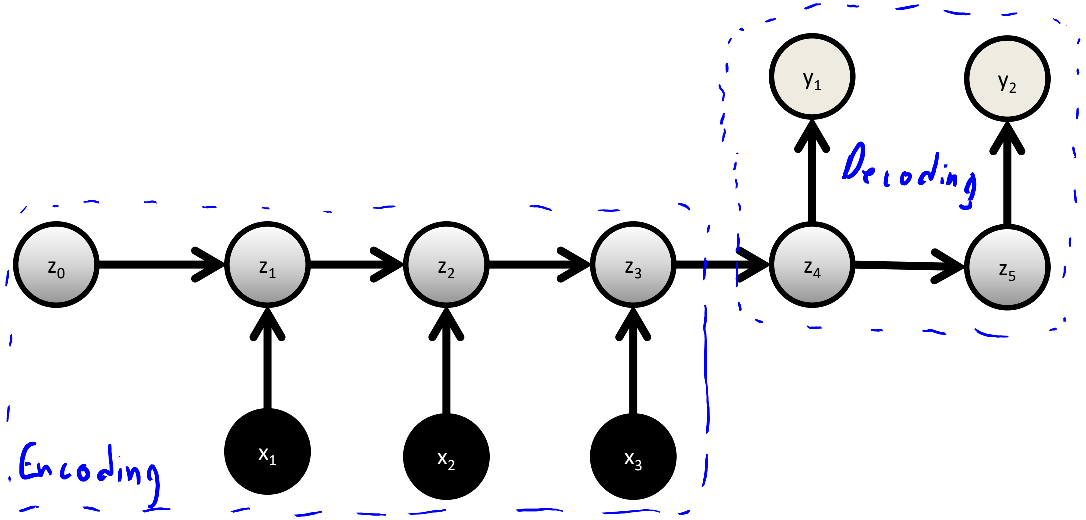

$$\underbrace{\begin{aligned} a_t &= o_t \circ h_o(c_t) \\ c_t &= f_t \circ c_{t-1} + i_t \circ g_t \\ g_t &= h_o(W_g x_t + U_g a_{t-1}) \end{aligned}}_{\text{activation/memory cell/new memory value}}$$

$$\underbrace{\begin{aligned} f_t &= h(W_f x_t + U_f a_{t-1}) \\ i_t &= h(W_i x_t + U_i a_{t-1}) \\ o_t &= h(W_o x_t + U_o a_{t-1}) \end{aligned}}_{\text{forget/input/output sigmoid "gates"}}$$

- We discussed using encoders and decoders of different data types:
  – Encoder takes an image and decoder outputs a sequence.
  – Image captioning, video annotation, lip reading, poetry about images.

# Previously: Sequence-to-Sequence RNNs

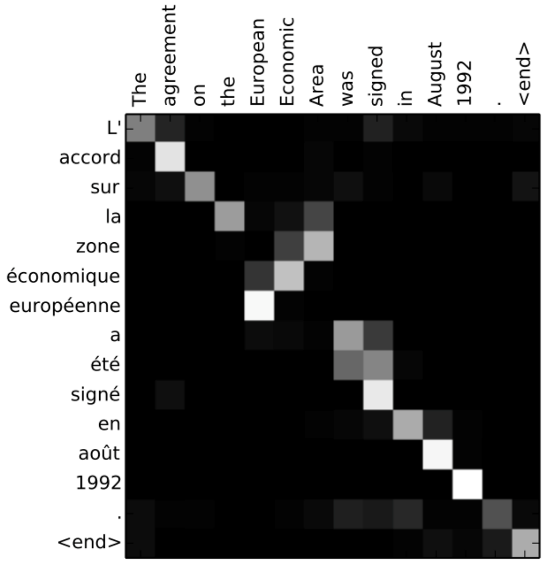
- Sequence-to-sequence:
  - Recurrent neural network for sequences of different lengths.




- Problem:
  - All "encoding" information must be summarized by last state ($z_3$ above).
  - Might "forget" earlier parts of sentence.
    - Or middle of sentence if using bi-directional RNN.
  - Might want to "re-focus" on parts of input, depending on decoder state.

# Attention

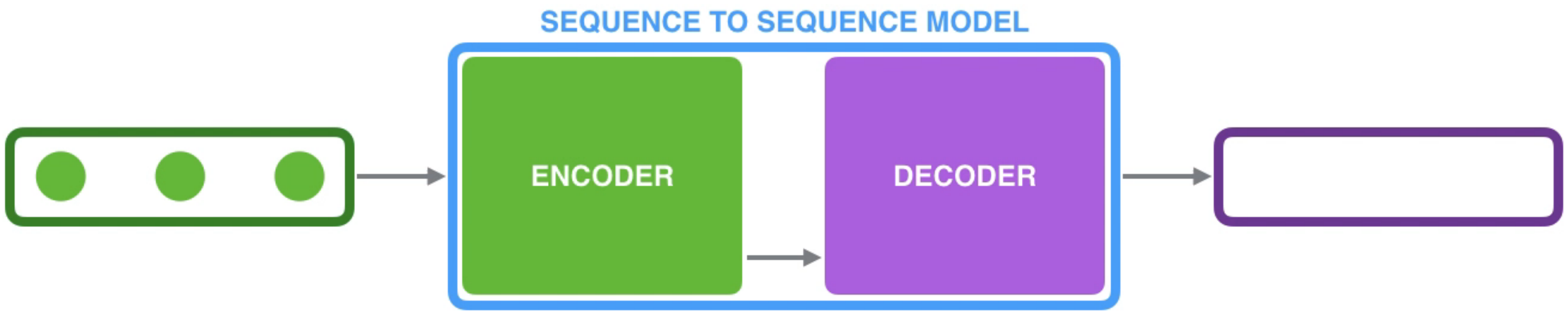
- Many recent systems use "attention" to focus on parts of input.
  - Including "neural machine translation" system of Google Translate.

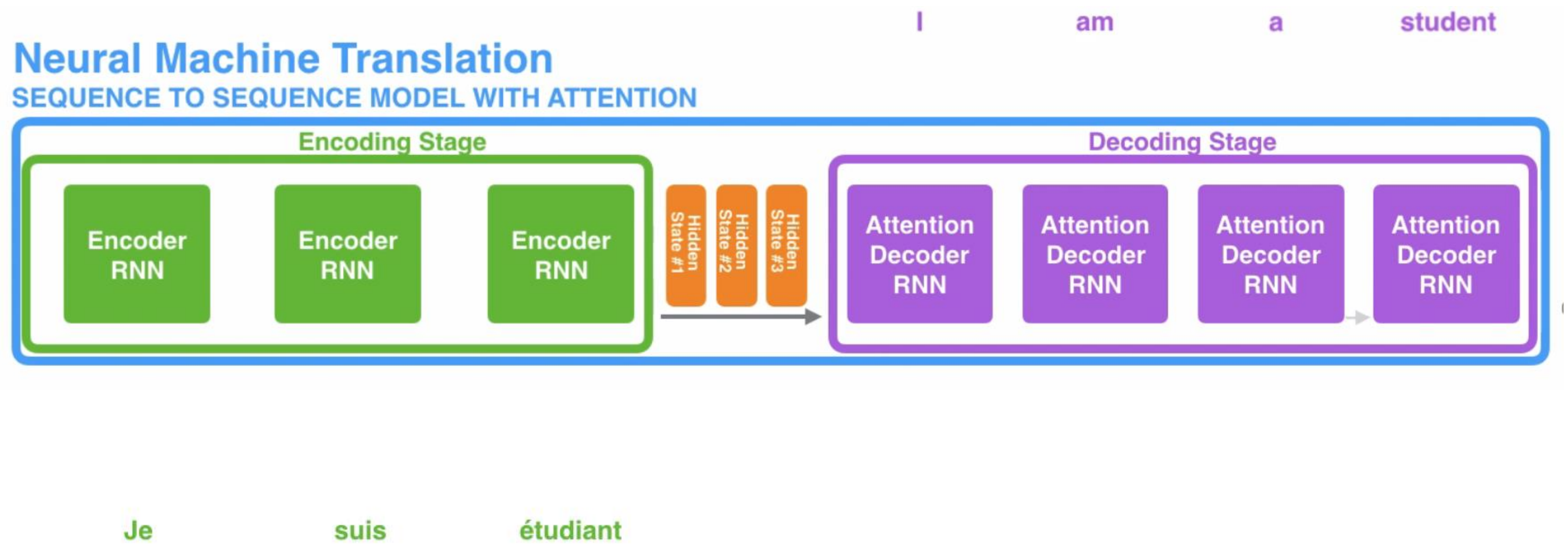



- Many variations on attention, but usually include the following:
  - Each decoding can use hidden state from each encoding step.
    - Used to re-weight during decoding to emphasize important parts.

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

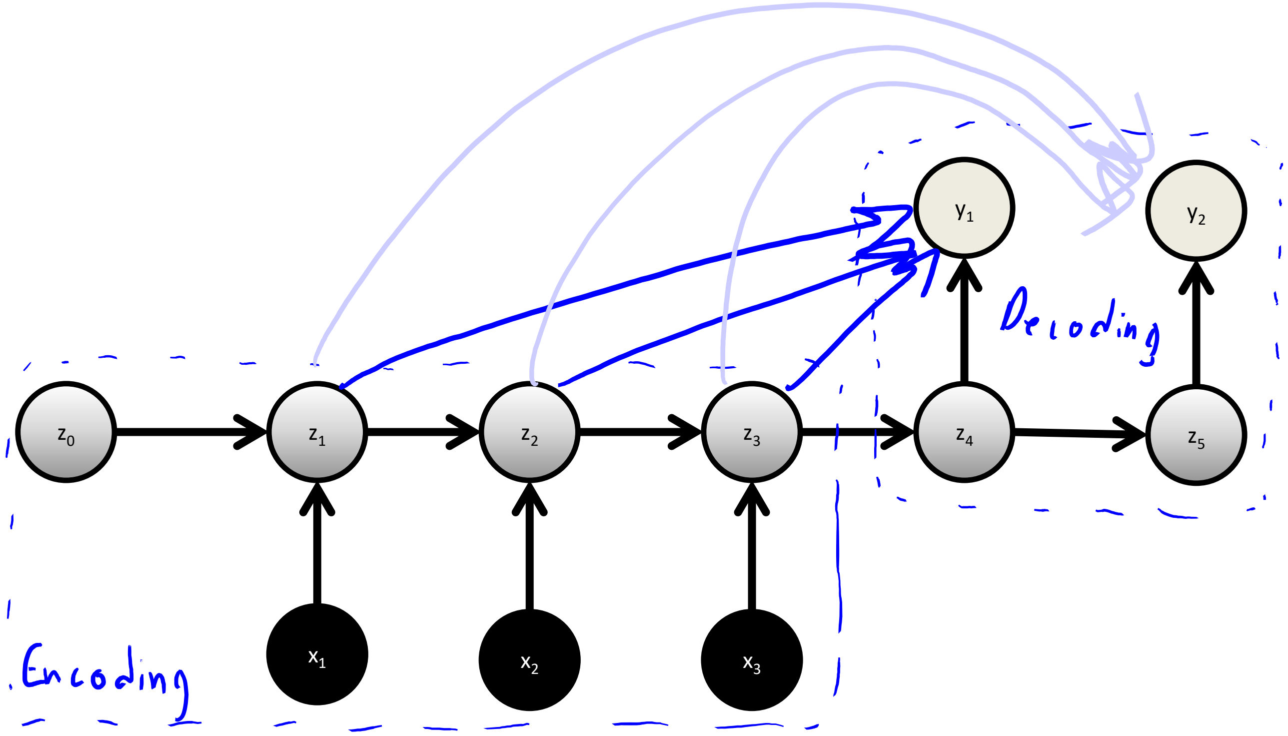
# RNN vs. RNN with Attention Videos



# Not-Very-Practical Attention

- A naïve "attention" method (no one uses this, but idea is similar):
  - At each decoding step, weight decoder state (as usual) and weight all encoder states.




Would require all inputs to have the same length.

  - Another variation on the "residual connection" or "denseNet" trick.
  - But this variant is not practical since number of decoding weights depends on input size.
    - Practical variations try to summarize encoder information through a "context vector".

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

# Context Vectors

- A common way to generate the context vector:
  - Take current decoder state.
  - Compute inner product with each encoder state.
    - Gives a scalar for each encoding "time".
  - Pass these scalars through the softmax function.
    - Gives a normalized weight for each time (what was previously shown in pairwise tables).
  - Multiply each encoder state by probability, add them up.
    - Gives fixed-length "context vector".

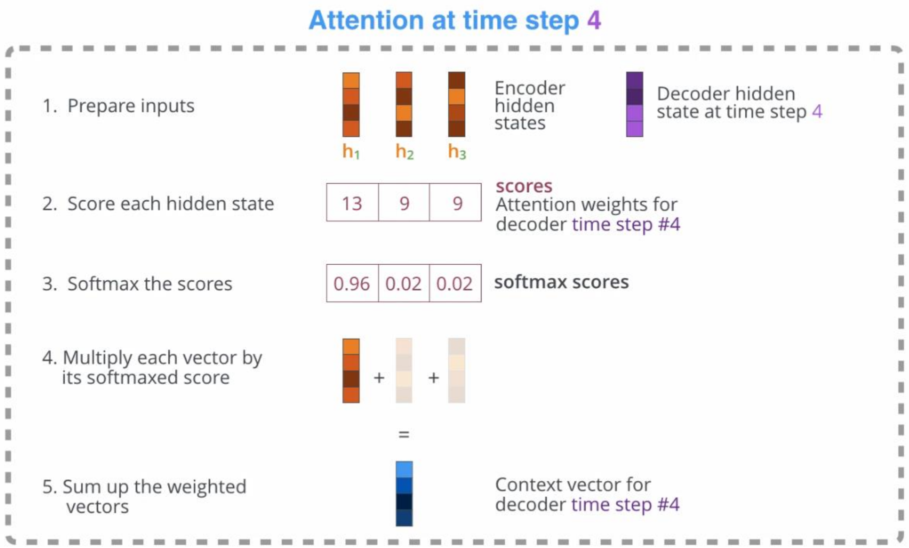
Attention at time step 4

- Alternate notation (like a hash function):
  - Input is "queries" and "keys".
  - Output is "values".

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

# Context Vectors

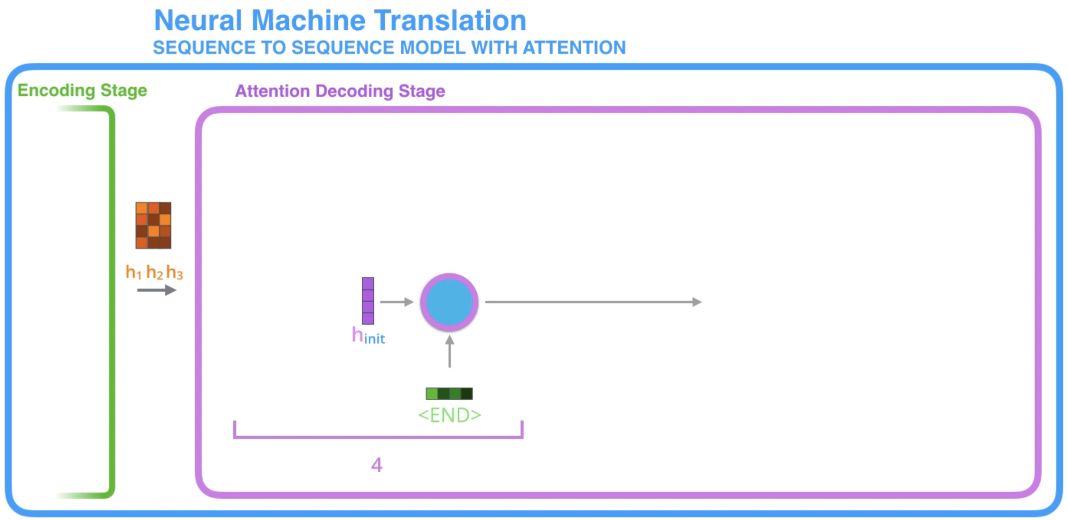
- A common way to generate the context vector:
  - Take current decoder state.
  - Compute inner product with each encoder state.
    - Gives a scalar for each encoding "time".
  - Pass these scalars through the softmax function.
    - Gives a normalized weight for each time (can be shown in pairwise tables).
  - Multiply each encoder state by probability, add them up.
    - Gives fixed-length "context vector".




- Alternate notation (like a hash function):
  - Input is "queries" and "keys".
  - Output is "values".

# Using Context Vectors for Attention

- Context vector is usually appended to decoder's state when going to next layer.
  - Output could be generated directly from this, or passed through a neural net.
  - Common variation is "multi-headed attention": can get scores from different aspects.
    - One context vector for semantics, one for grammar, one for tense, and so on.
    - Each is appended to decoder state when going to next layer.
    - Context vectors are usually not included when updating the decoder state temporally.

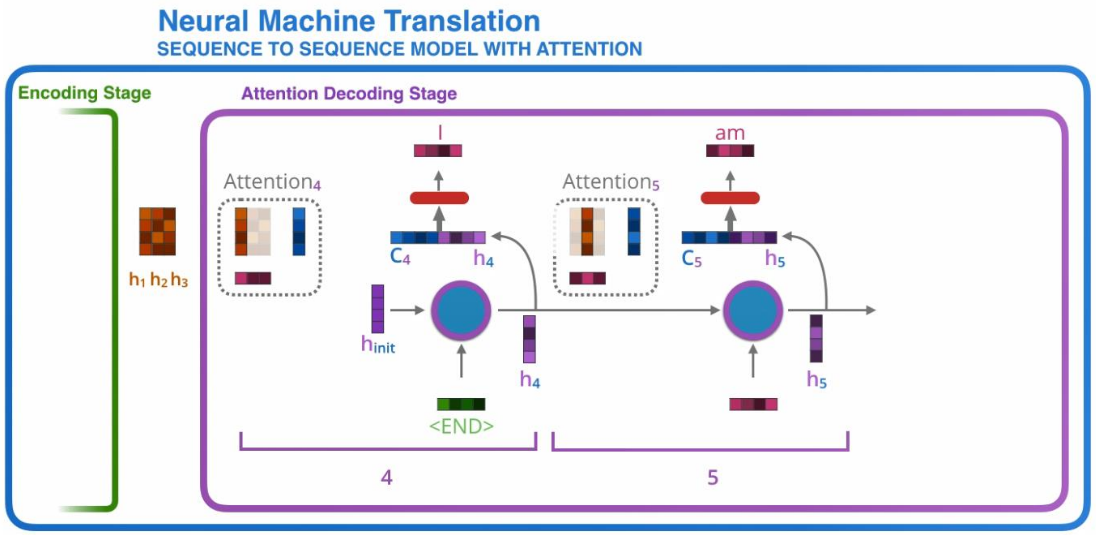



- Remember that we train the encoder and decoder at the same time.

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

# Using Context Vectors for Attention

- Context vector is usually appended to decoder's state when going to next layer.
  - Output could be generated directly from this, or passed through a neural net.
  - Common variation is "multi-headed attention": can get scores from different aspects.
    - One context vector for semantics, one for grammar, one for tense, and so on.
    - Each is appended to decoder state when going to next layer.
    - Context vectors are usually not included when updating the decoder state temporally.



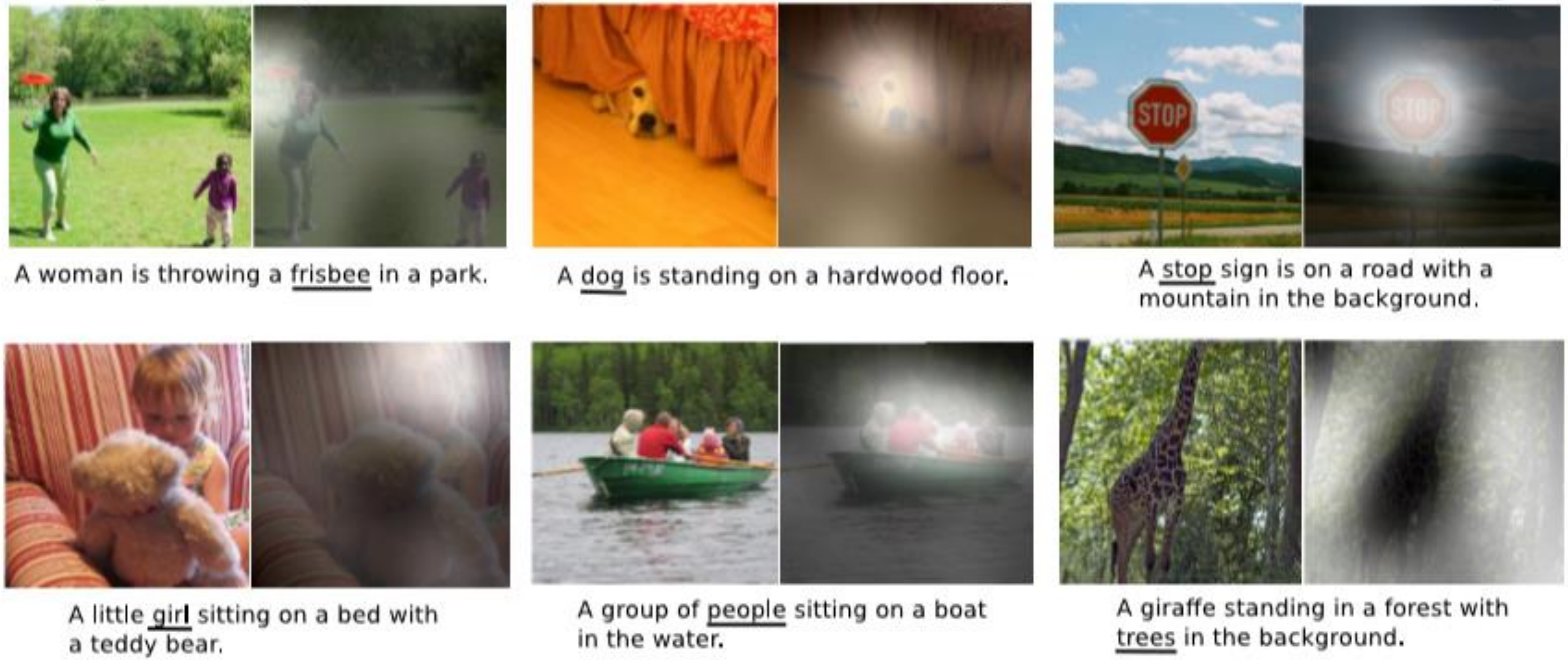

- Remember that we train the encoder and decoder at the same time.

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

# Multi-Modal Attention

- Attention for image captioning:

*Figure 3.* Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)




https://arxiv.org/pdf/1502.03044.pdf

# Biological Motivation for Attention

- Gaze tracking:
  - https://www.youtube.com/watch?v=QUbiHKucljw

- Selective attention test:
  - https://www.youtube.com/watch?v=vJG698U2Mvo

- Change blindness:
  - https://www.youtube.com/watch?v=EARtANyz98Q

- Door study:
  - https://www.youtube.com/watch?v=FWSxSQsspiQ

# Neural Turing/Programmers

- Many interesting variations on memory/attention.
  - A getting-out-of-date survey: https://distill.pub/2016/augmented-rnns

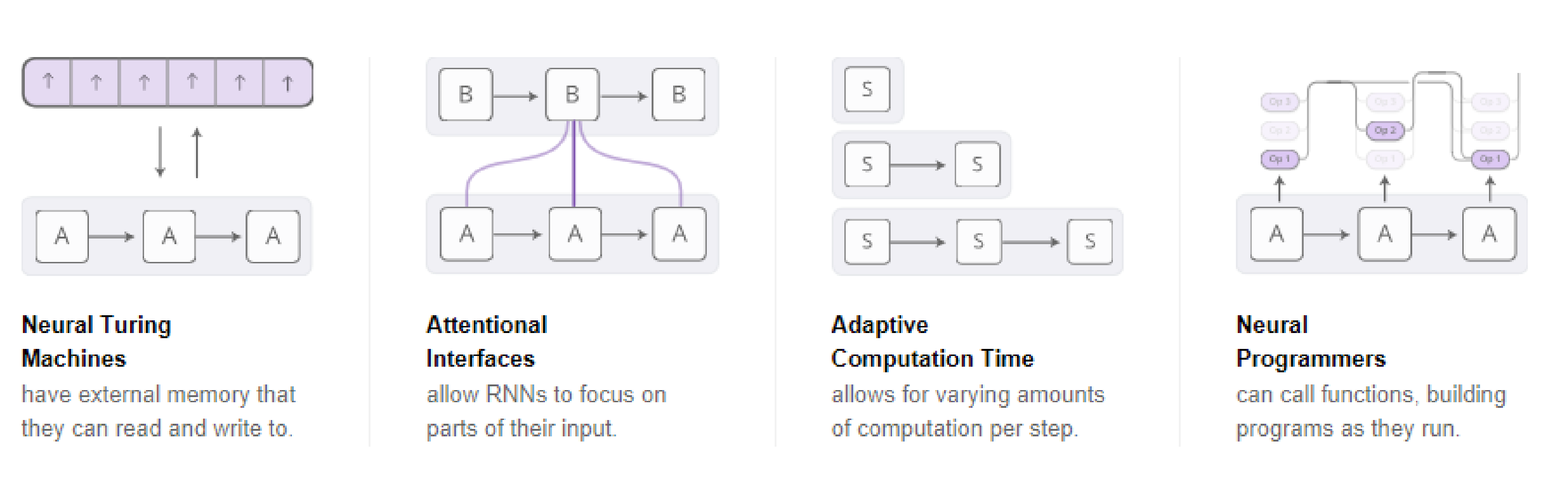
Here is an example of what the system can do. After having been trained, it was fed the following short story containing key events in JRR Tolkien's Lord of the Rings:

Bilbo travelled to the cave.
Gollum dropped the ring there.
Bilbo took the ring.
Bilbo went back to the Shire.
Bilbo left the ring there.
Frodo got the ring.
Frodo journeyed to Mount-Doom.
Frodo dropped the ring there.
Sauron died.
Frodo went back to the Shire.
Bilbo travelled to the Grey-havens.
The End.

After seeing this text, the system was asked a few questions, to which it provided the following answers:

Q: Where is the ring?
A: Mount-Doom
Q: Where is Bilbo now?
A: Grey-havens
Q: Where is Frodo now?
A: Shire

It's probably one of the few technical papers that cite "Lord of the Rings".




  - We will focus next on a wildly-popular variant called "transformers".

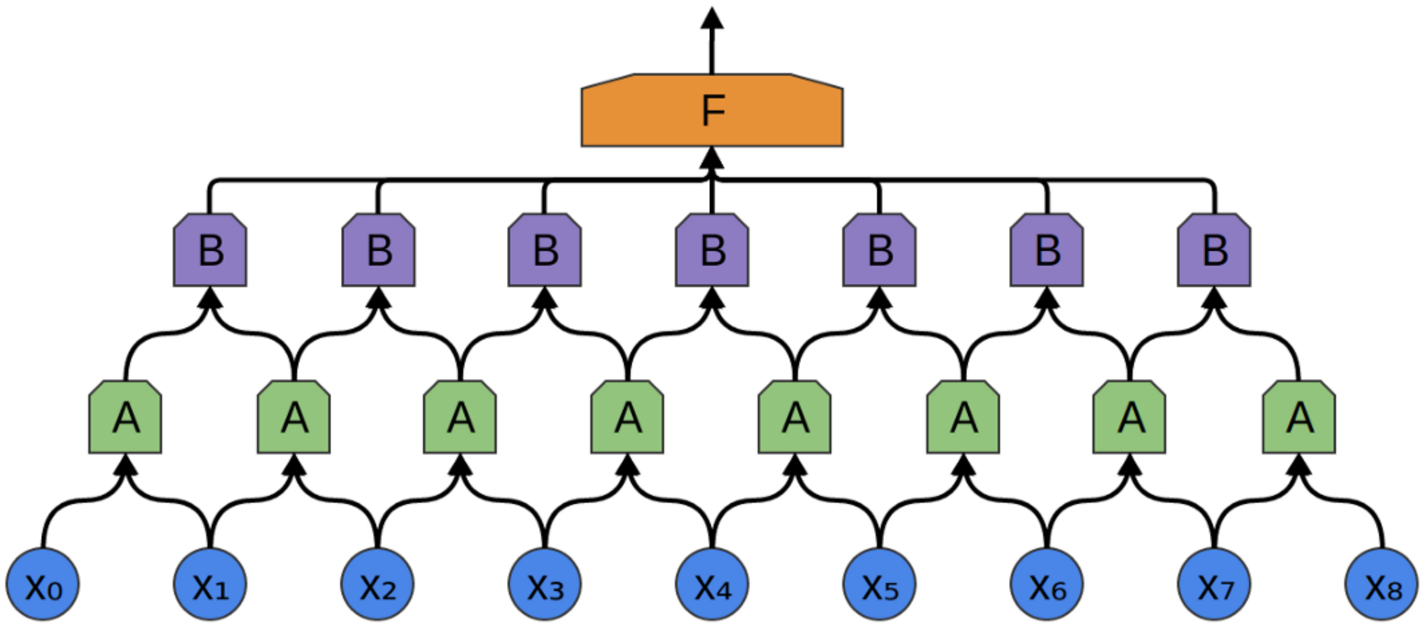
https://www.facebook.com/FBAIResearch/posts/362517620591864

# Next Topic: Transformers

# Convolutions for Sequences?

- Should we really be going through a sequence sequentially?
  - What if stuff in the middle is really important, and changes meaning?

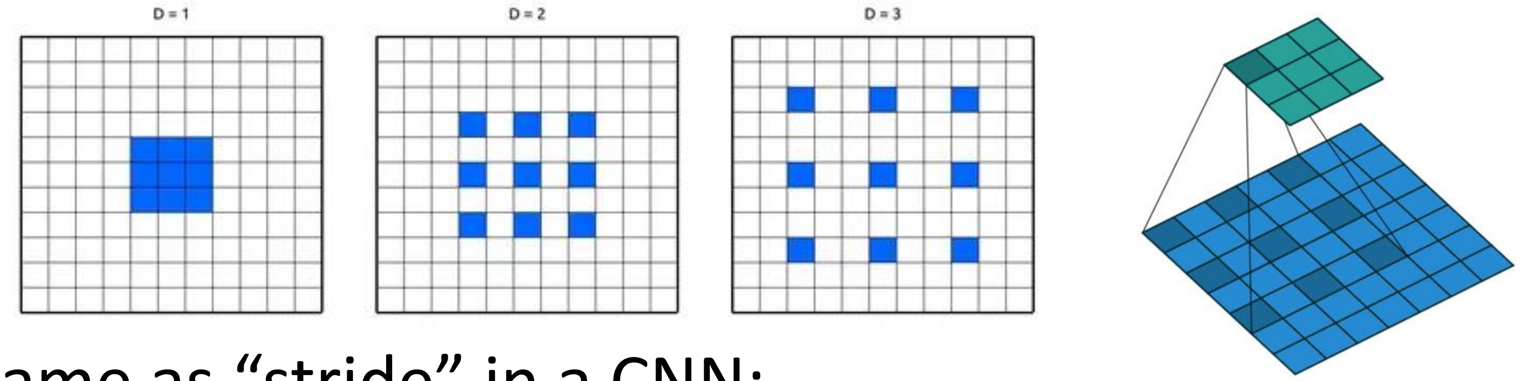
- Recent works have explored using convolutions for sequences.




https://medium.com/@TalPerry/convolutional-methods-for-text-d5260fd5675f

# Digression: Dilated Convolutions ("a trous")

- Best CNN systems have gradually reduced convolutions sizes.
  - Many modern architectures use 3x3 convolutions, far fewer parameters.
- Sequences of convolutions take into account larger neighbourhood.
  - 3x3 convolution followed by another gives a 5x5 neighbourhood.
  - But need many layers to cover a large area.
- Alternative recent strategy is dilated convolutions ("a trous").

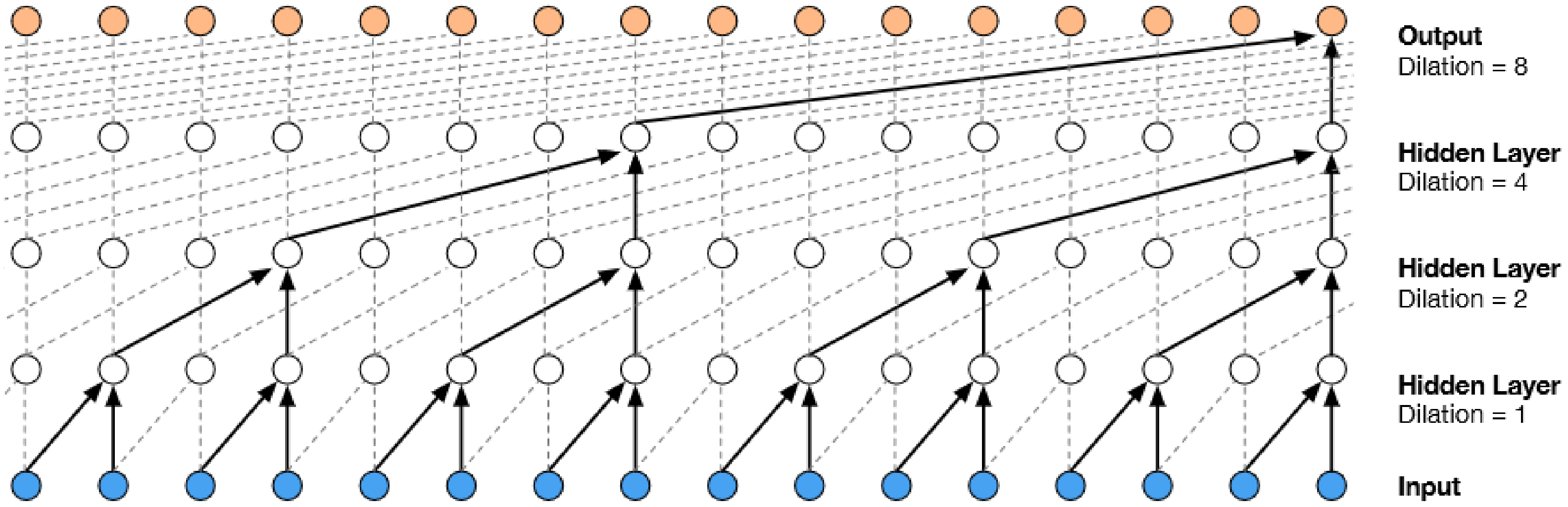



- Not the same as "stride" in a CNN:
  - Doing a 3x3 convolution at all locations, but using pixels that are not adjacent.

https://medium.com/@TalPerry/convolutional-methods-for-text-d5260fd5675f
https://github.com/vdumoulin/conv_arithmetic

# Dilated Convolutions (“a trous”)

- Modeling music and language and with dilated convolutions:



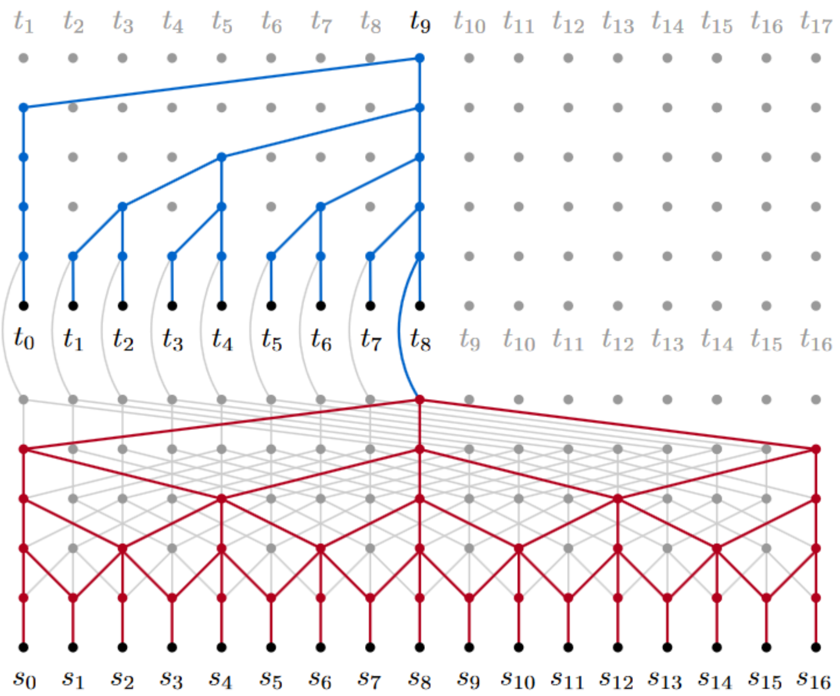




*Figure 1.* The architecture of the ByteNet. The target decoder (blue) is stacked on top of the source encoder (red). The decoder generates the variable-length target sequence using dynamic unfolding.

https://arxiv.org/pdf/1610.10099.pdf
https://arxiv.org/pdf/1609.03499.pdf

# RNNs/CNNs/Attention for Music and Dance

- Music generation:
  - https://www.youtube.com/watch?v=RaO4HpM07hE

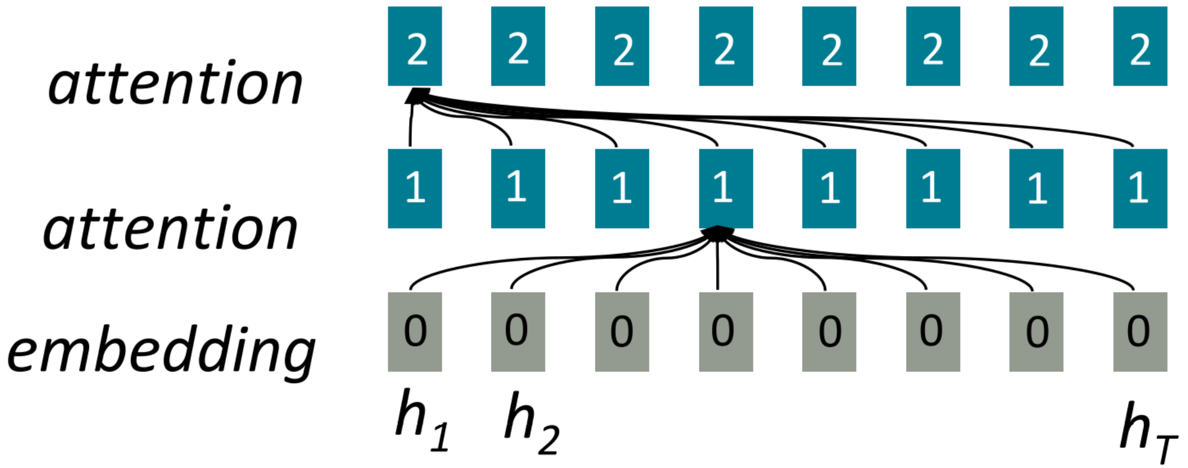
- Text to speech and music waveform generation:
  - https://deepmind.com/blog/wavenet-generative-model-raw-audio

- Dance choreography:
  - http://theluluartgroup.com/work/generative-choreography-using-deep-learning

- Music composition:
  - https://www.facebook.com/yann.lecun/videos/10154941390687143

# Transformer Networks

- CNNs are less sequential, but take multiple steps to combine distant information.
- "Attention is all you need": keep the attention, ditch the RNN/CNN.
  - Constant time to transfer across positions.
  - Uses "self-attention" layers to model relationship between all words in input.
    - Queries/keys/values all come from input in these steps.

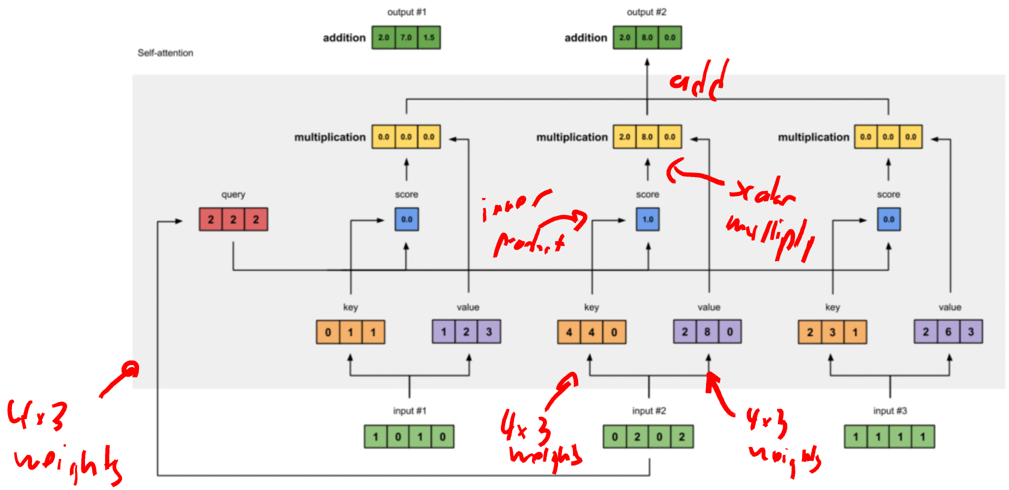



All words attend to all words in previous layer; most arrows here are omitted

- Sequence of representations of words, each depending on all other words.

# Transformer Networks

- CNNs are less sequential, but take multiple steps to combine distant information.
- "Attention is all you need": keep the attention, ditch the RNN/CNN.
  - Constant time to transfer across positions.
  - Uses "self-attention" layers to model relationship between all words in input.
    - Take weighted combinations of each input to generate a "key", a "value", and a "query".
    - Compute inner product between "query" from word with "key" for each word to give scalar "score".
    - Compute softmax of "scores", multiplied by word's "value", add these across words to get context vector.

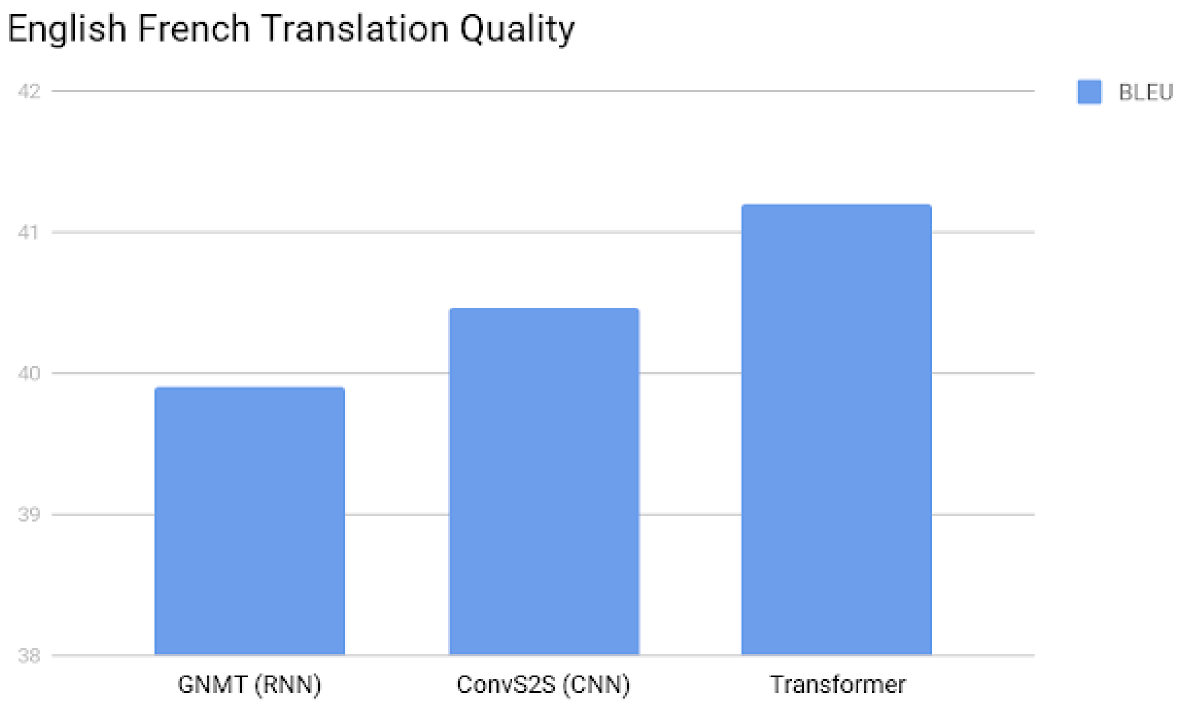



- Many variations exist.

https://towardsdatascience.com/illustrated-self-attention-2d627e33b20a

# Transformer Networks

- Multiple "self-attention" layers in transformers replacing RNN/CNN.
  – Has improved on state of the art results in many tasks.




(these models have around 100M parameters)

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

# Transformer Networks: Practical Issues

- "Self-attention" layers are basis for transformer networks.
  - Simple idea, but practical systems have a lot of moving pieces.

- Problem: position information is lost (self-attention is unordered).
  - "Position representations" are additional variables added to each layer.
- Problem: information about the future can be visible in the past.
  - During training, prevent decoder from looking ahead.

- Further "standard" tricks to make it work better:
  - Multi-headed attention, skip/residual connections, and layer normalization.
  - Between layers, pass each embedding through a feedforward neural network.

http://web.stanford.edu/class/cs224n/slides/cs224n-2021-lecture09-transformers.pdf

# Transformer Architecture (from paper)

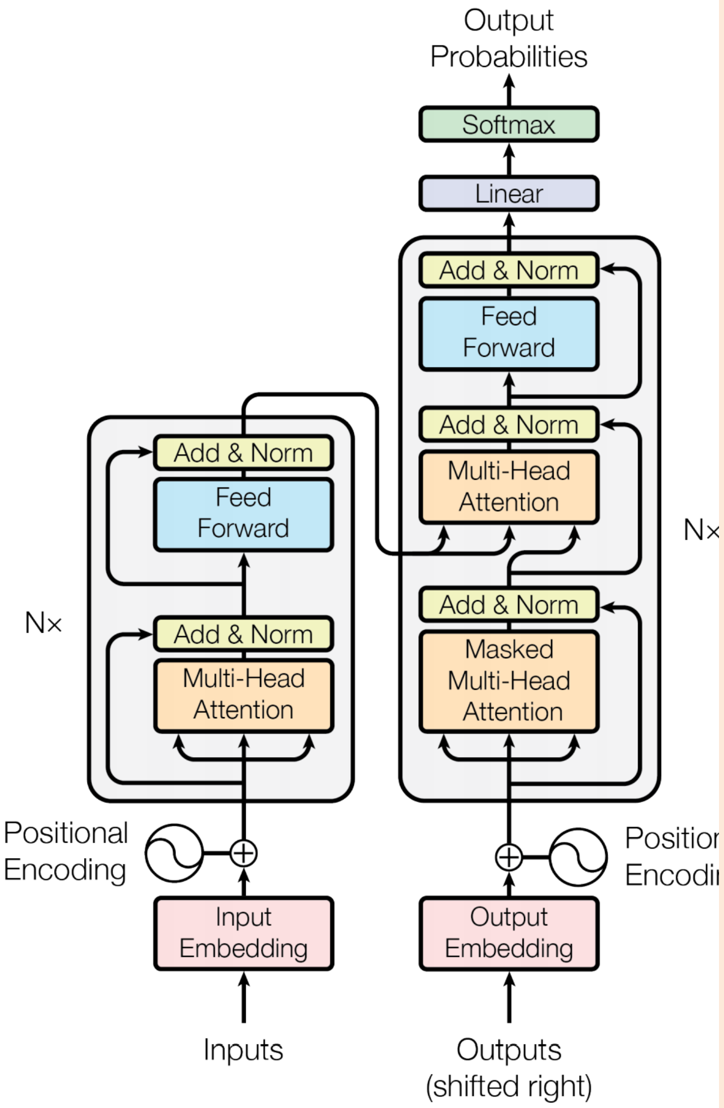



Figure 1: The Transformer - model architecture.

https://arxiv.org/pdf/1706.03762.pdf

# Subsequent Work

- BERT: incredibly-popular model in natural language processing.
  - Transformer model trained on masked sentences to predict masked words.
  - Then fine-tune the architecture on specific applications.

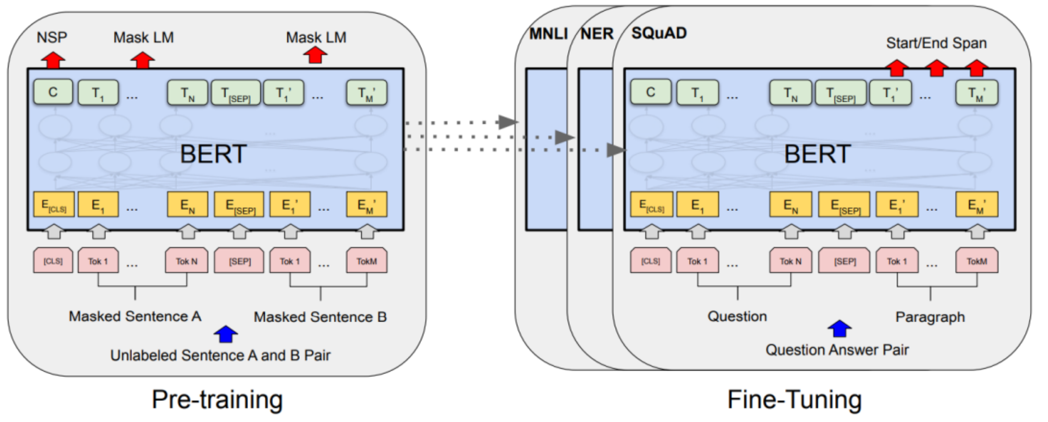



Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

- Transformers also form basis for other advanced language models (GPT).
- Transformers have been adapted to images, music, and so on.
  - Also see the reformer for decreasing the quadratic cost of transformers.

https://arxiv.org/pdf/1810.04805.pdf

# What are we learning?

| Alteration | Movie Review | Label |
|---|---|---|
| Original | A triumph, relentless and beautiful in its downbeat darkness | + |
| Swap | A triumph, relentless and beuatiful in its downbeat darkness | – |
| Drop | A triumph, relentless and beautiful in its dwnbeat darkness | – |
| + Defense | A triumph, relentless and beautiful in its downbeat darkness | + |
| + Defense | A triumph, relentless and beautiful in its downbeat darkness | + |

Table 1: Adversarial spelling mistakes inducing sentiment misclassification and word-recognition defenses.

- Single-character attacks on Bert can lower accuracy from 90 to 45%.
- Large datasets used to train often contain some toxic content.

https://arxiv.org/pdf/1810.04805.pdf

# Summary

- Attention:
  - Allow decoder to look at previous states.
- Context vectors:
  - Combine previous states into a fixed-length vector.
- [Dilated] convolutions for sequences.
  - Alternative to sequential architectures like RNNs.
- Transformer networks:
  - Layers of “self-attention” to build context.
    - “Everything depends on everything”, and you learn how.
    - Lots of implementation details, but excellent performance on many tasks.
    - Basis for modern enormous/impressive language models and applications.

- Next time: everyone’s favourite distribution.