U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

LIACC

M.EIC

# Natural Language Processing

Henrique Lopes Cardoso

FEUP / LIACC

hlc@fe.up.pt

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

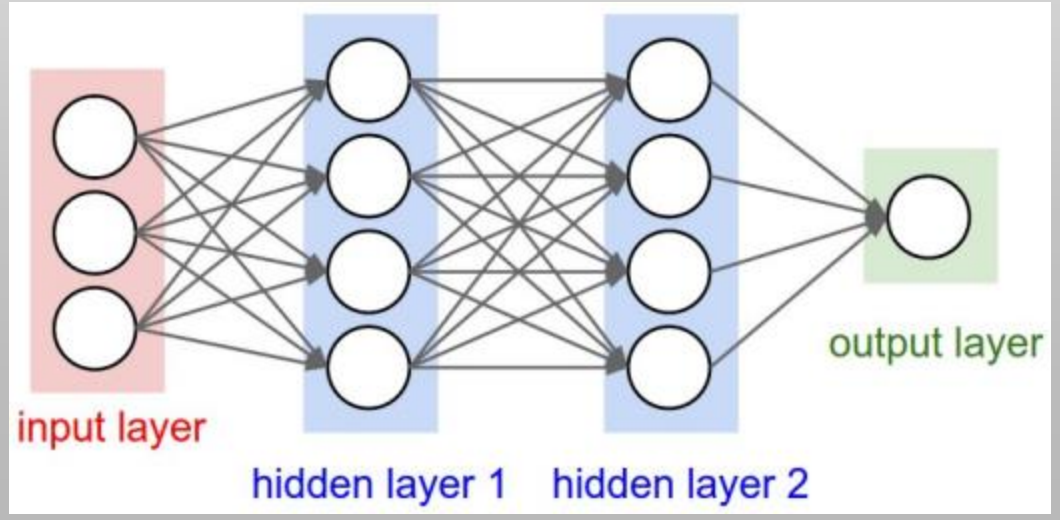
LIACC

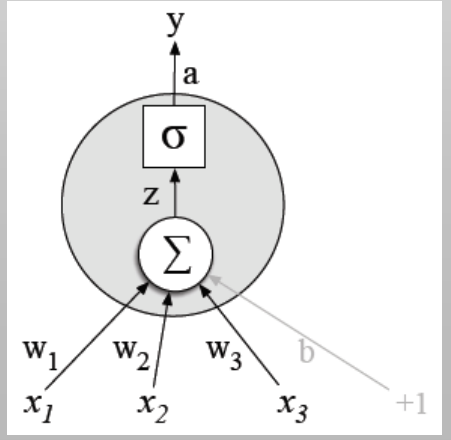
# Neural Networks in NLP

representation learning, deep learning, RNN, sequence-to-sequence, attention, Transformer, large language models

# Neural Networks

- Neurons organized in multiple layers
- Feed-forward neural network
  - Fully connected: often, neurons in a layer have connections to every neuron in the next layer
- Loss function (cross-entropy loss) and gradient descent using error back propagation



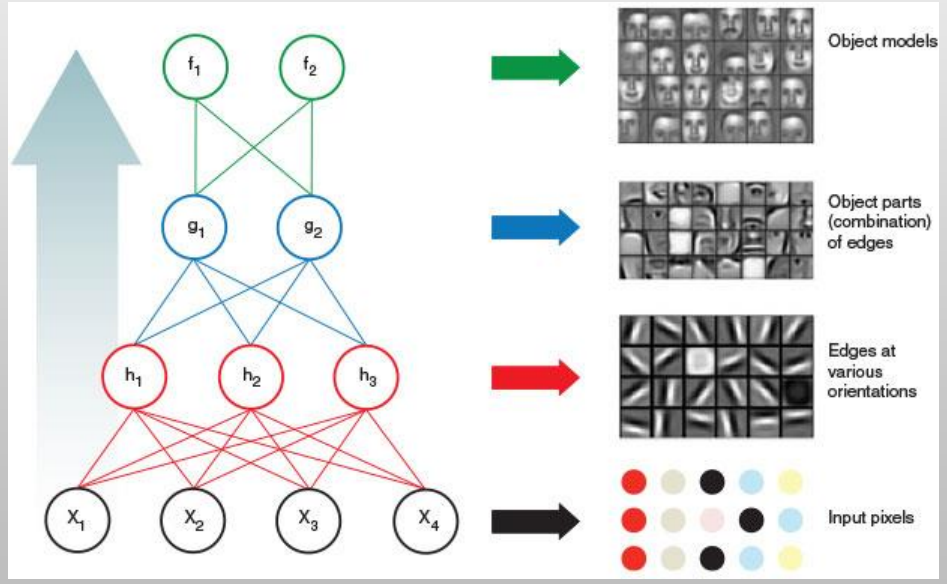



# Learning in Neural Networks

- Non-convex optimization problem
- Useful to initialize weights with small random numbers and to normalize input values
- Regularization: dropout, early stopping (dev set)
- Hyperparameter tuning on a dev set: learning rate, mini-batch size, model architecture and activation functions, regularization techniques, optimizer, …

# Representation Learning

- Most early work on NLP focused on human-designed representations and input features
- Representation learning attempts to automatically learn good features and representations
  - Word embeddings to represent continuous semantics
  - Each hidden layer builds a vector which is a hidden representation of its input
- Deep learning attempts to learn multiple levels of representation of increasing complexity/abstraction



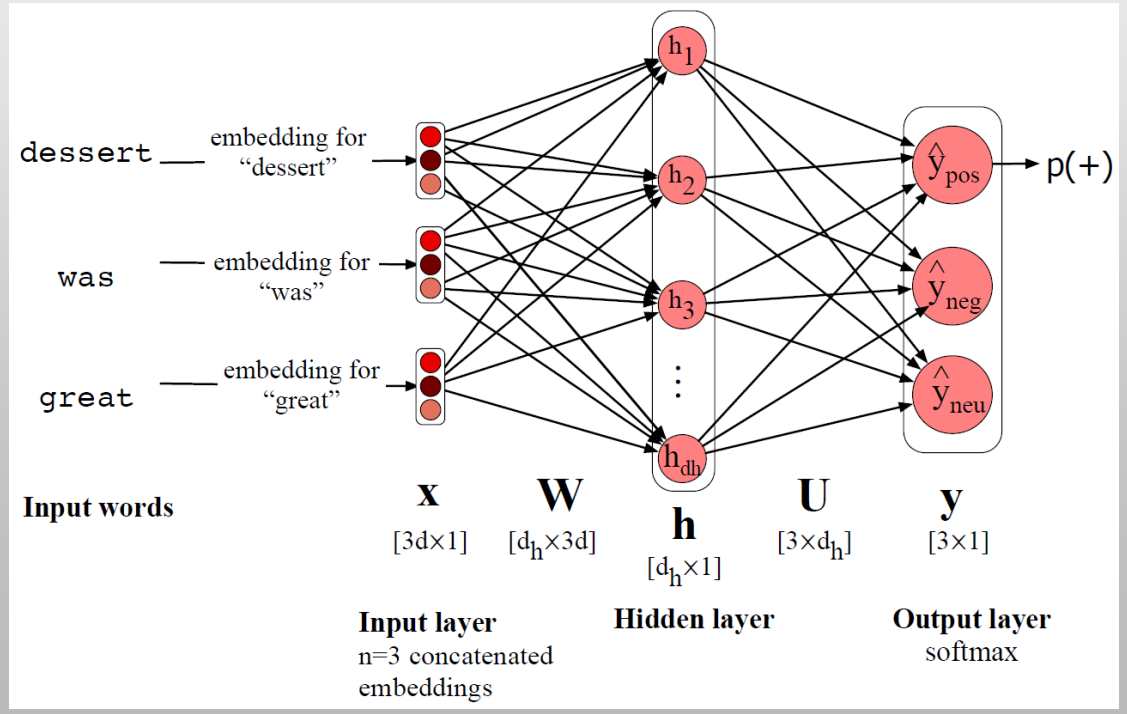
# Deep Learning

"The general approach to building Deep Learning systems is compelling and powerful: The researcher defines a model architecture and a top-level loss function and then both the parameters and the representations of the model self-organize so as to minimize this loss, in an end-to-end learning framework."

[Chris Manning, 2015]

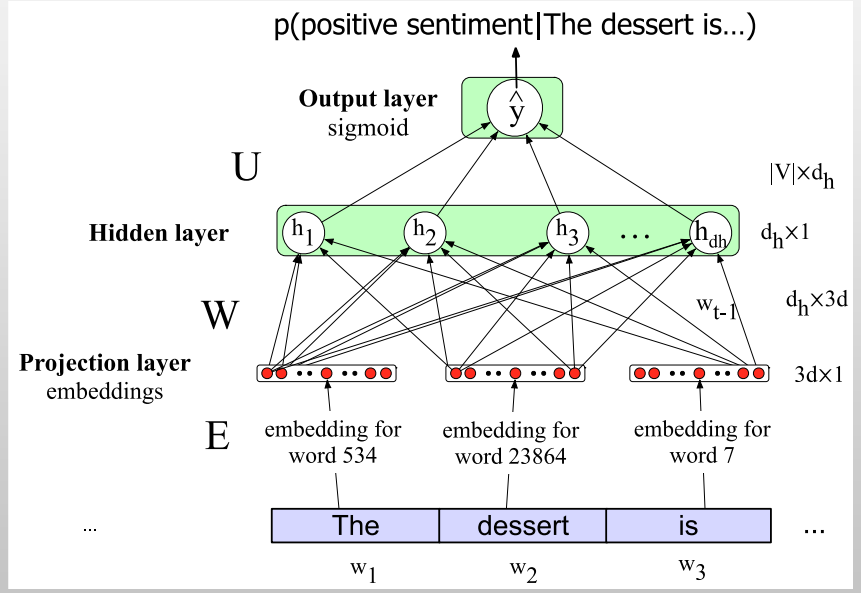
# Using Embeddings as Input

- Feed-forward neural network



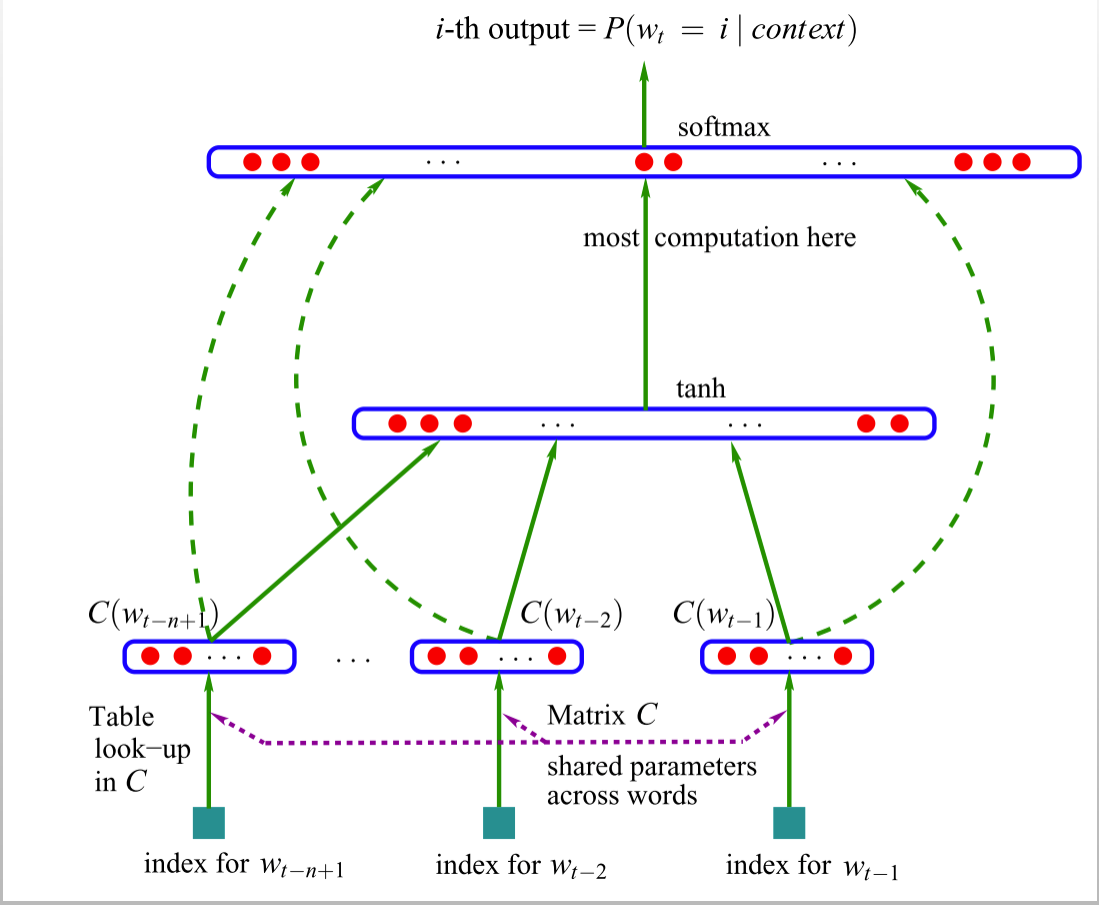
# Using Embeddings as Input

- How do we deal with text of varying length?




- Simple approaches:
  - Set input length to the length of the longest sentence
    - If shorter, pad with zero embeddings
    - If longer (at test time), truncate
  - Create a single sentence embedding
    - Take the mean of all word embeddings in the sentence
    - Take the element-wise max of all word embeddings in the sentence

# Feed-forward Neural Language Models

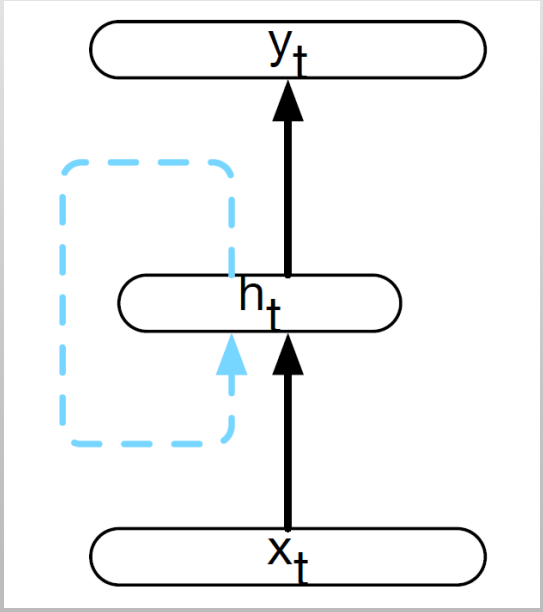


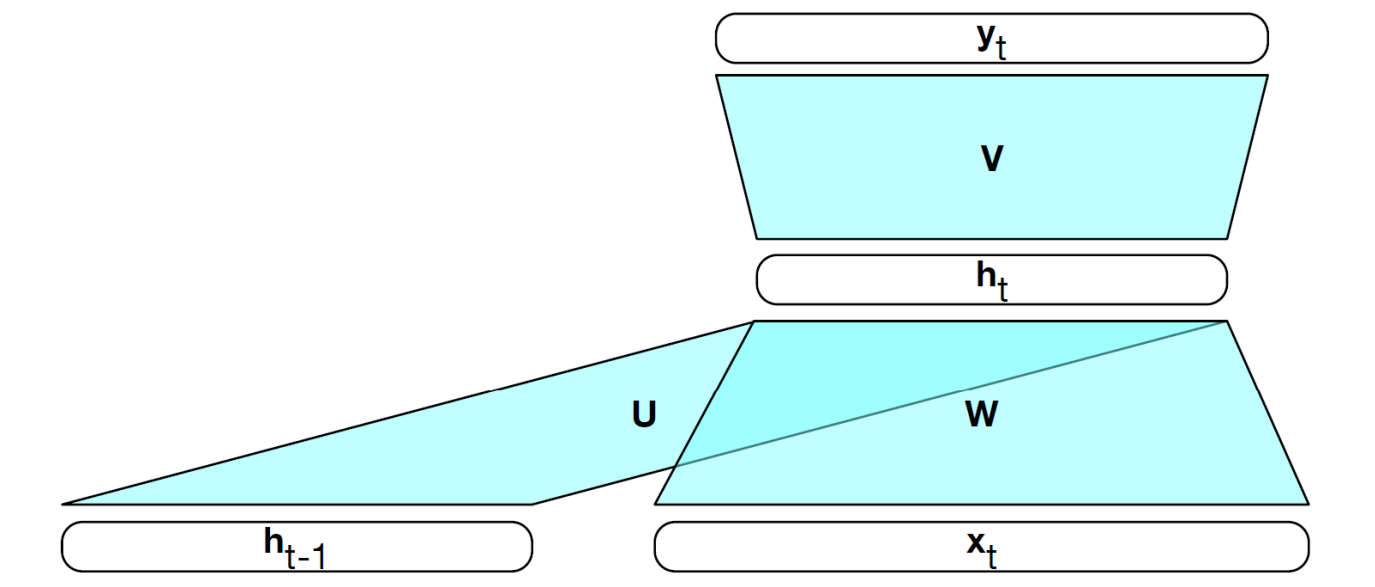

[Bengio et al., 2003]

- Advantages over n-gram language models:
  - Handle much longer histories
  - Generalize better to similar words
  - More accurate at word-prediction
- Drawbacks:
  - More complex and slower to train
  - Less interpretable

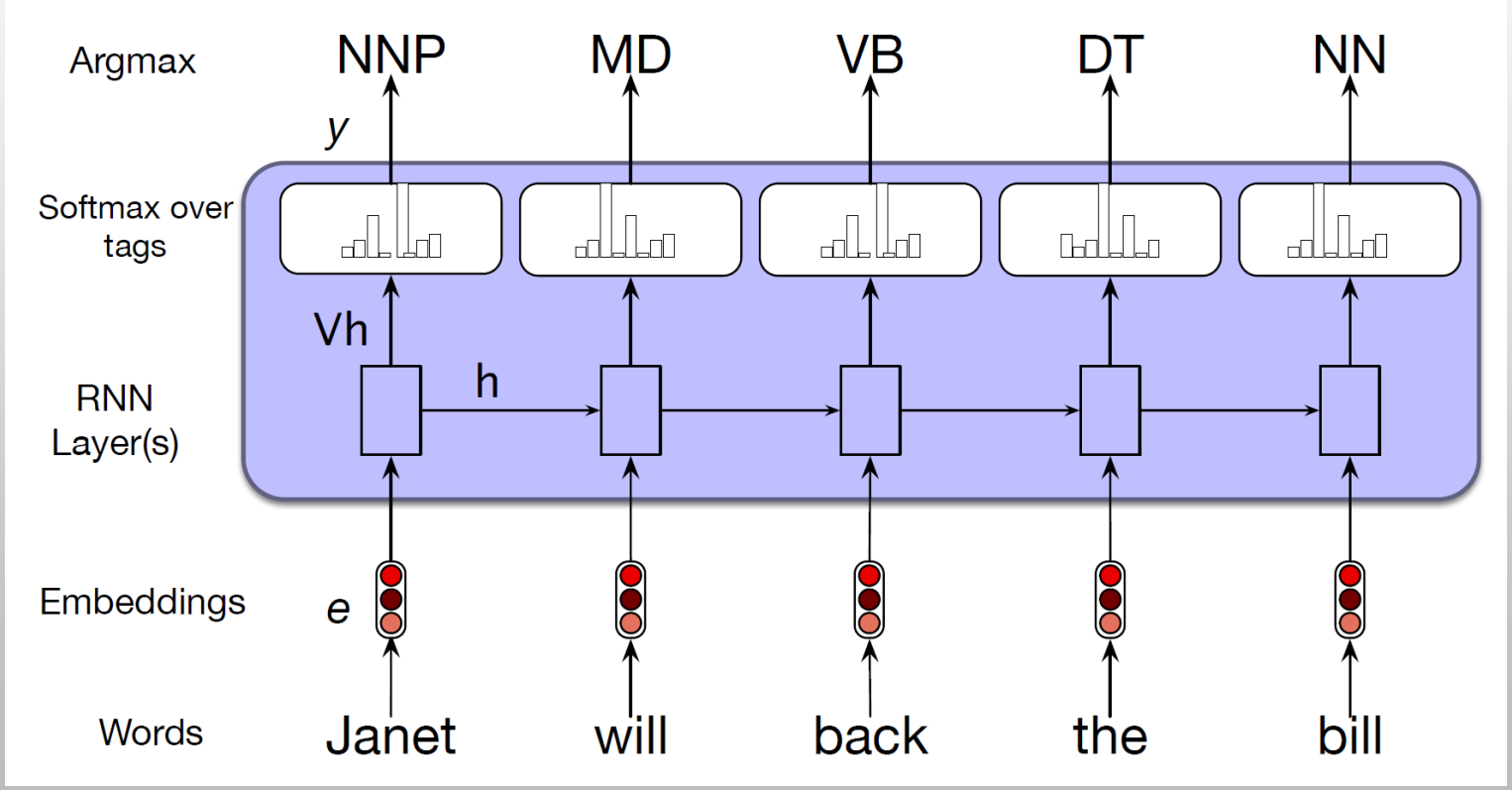
# Recurrent Neural Networks (RNN)

- Dealing with dynamic input sequences

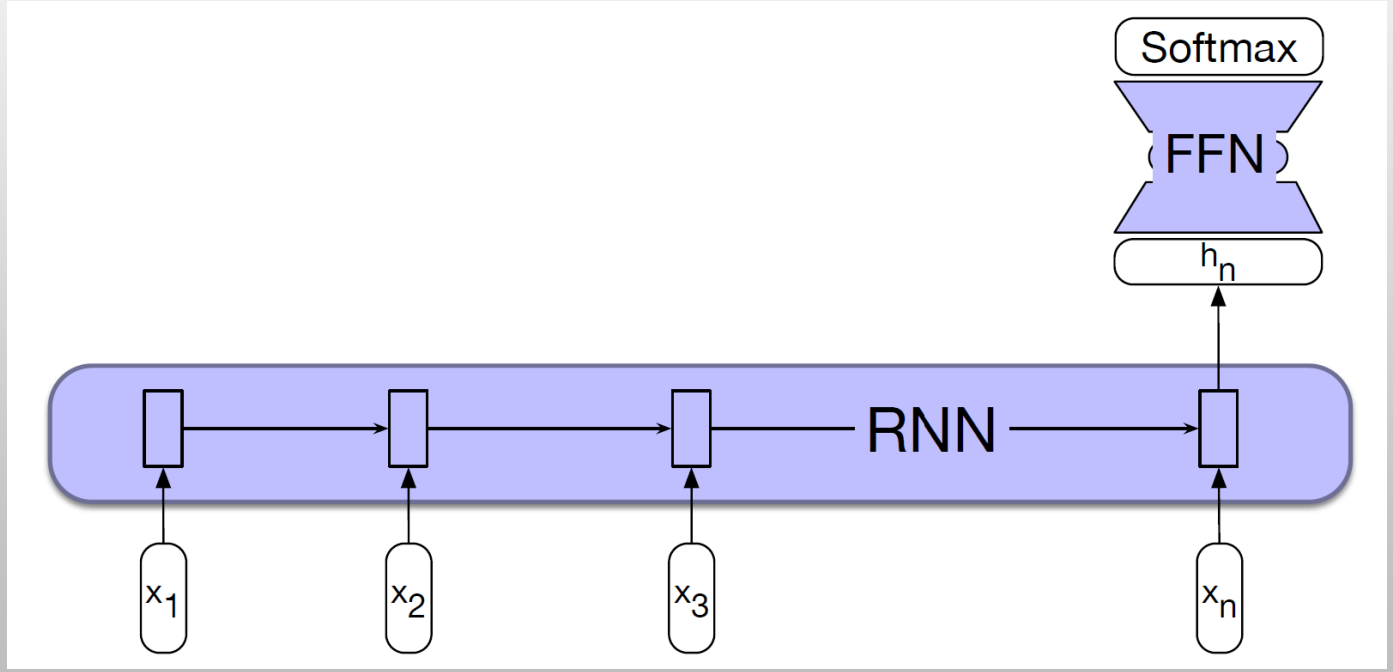


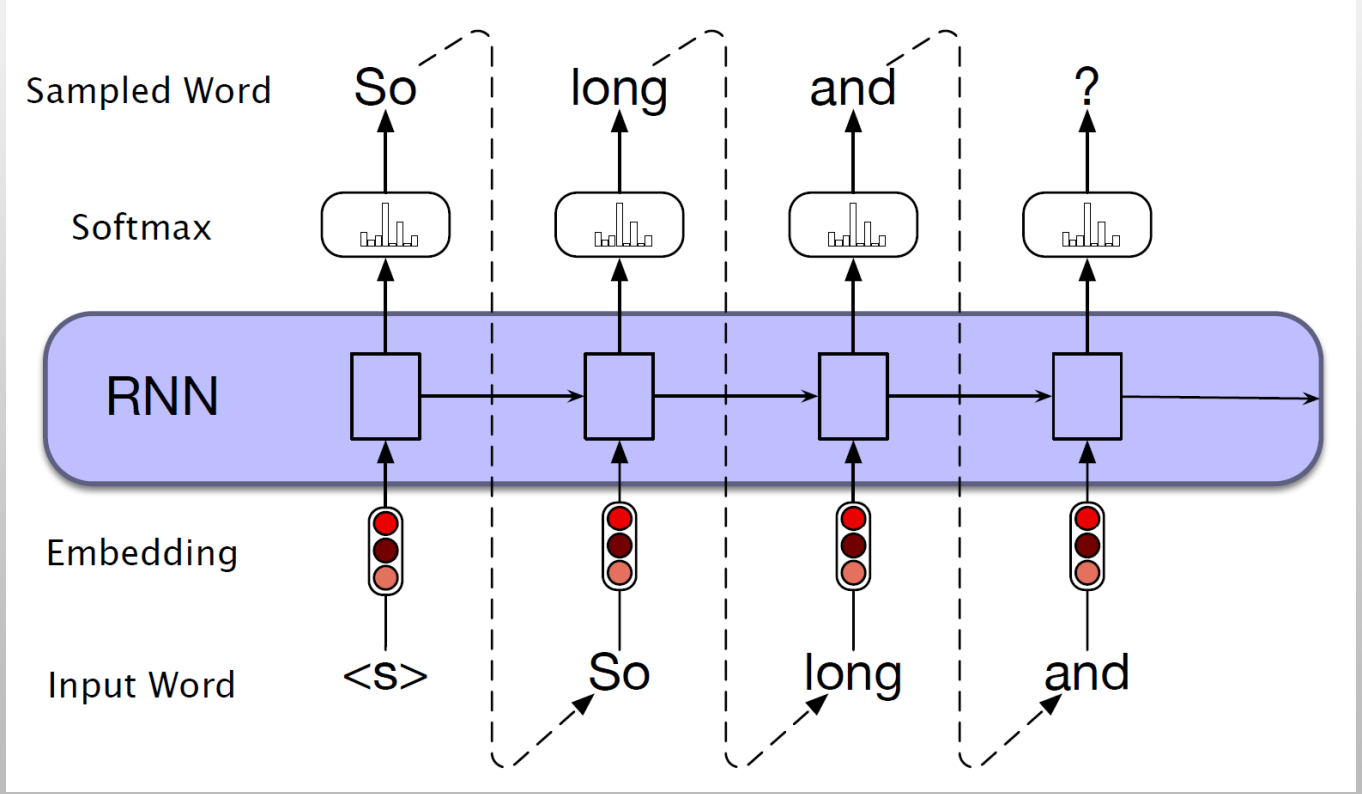
# RNN for Sequence Labeling



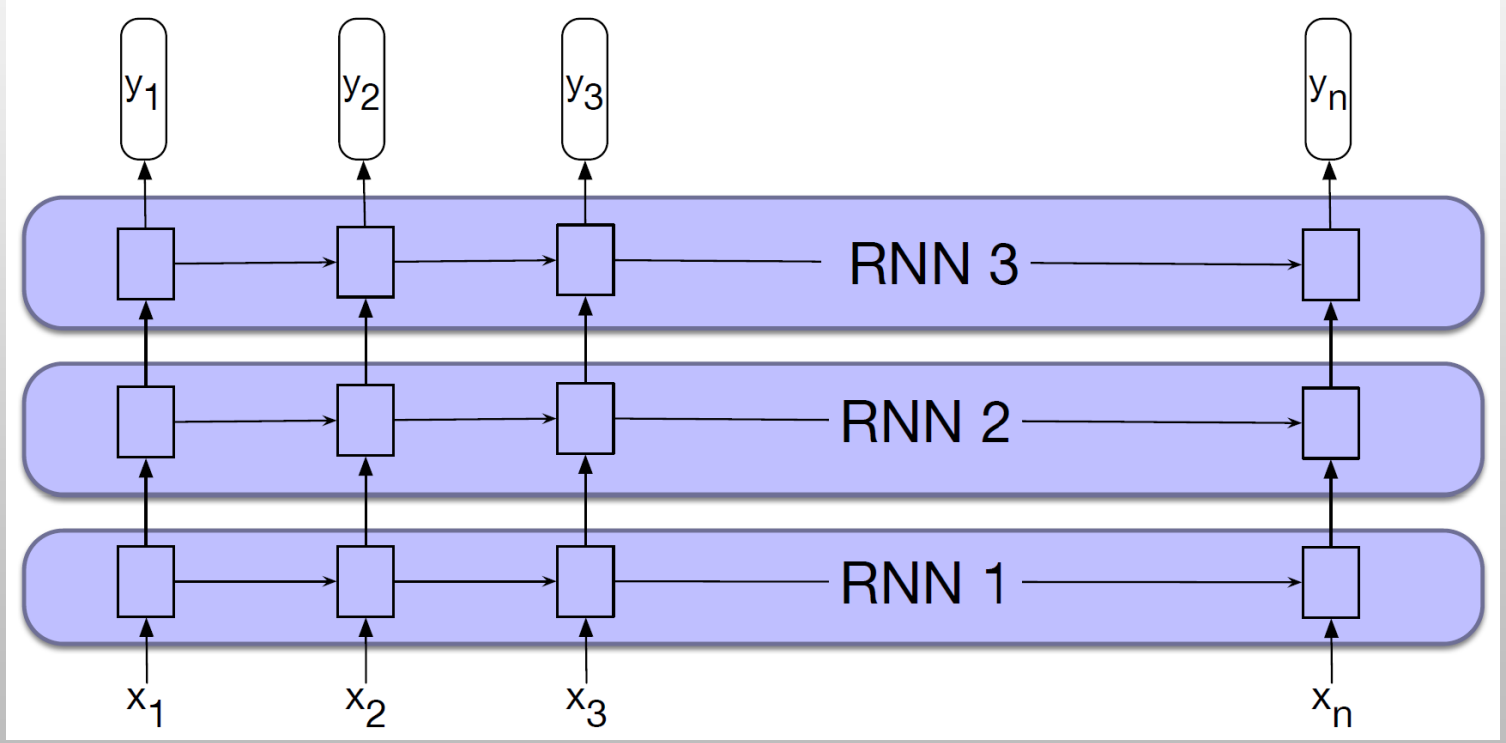
# RNN for Sequence Classification



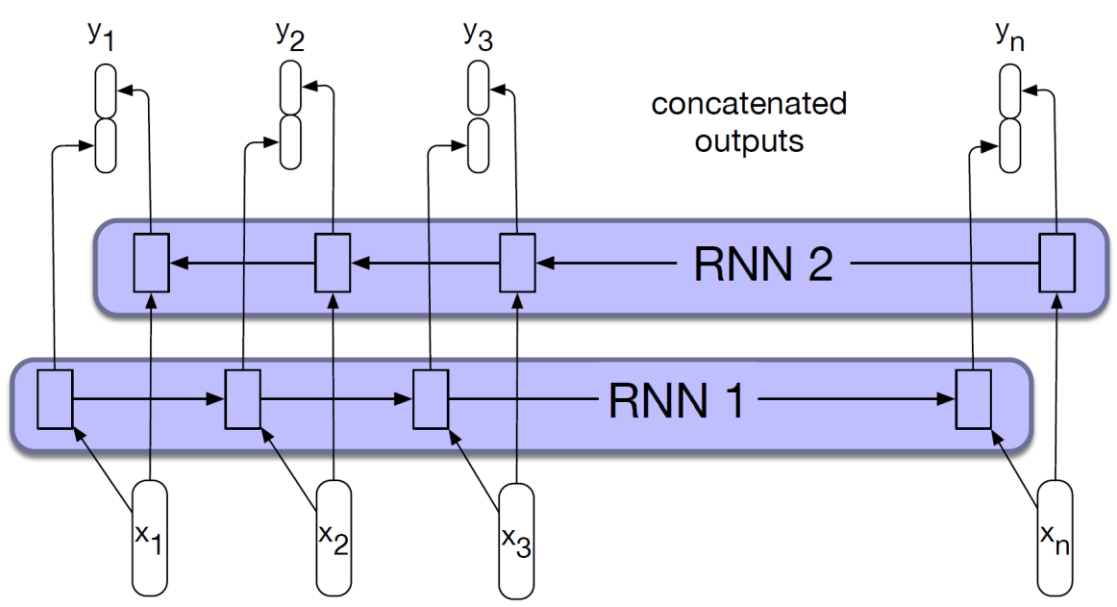
# RNN for Language Modeling and Text Generation

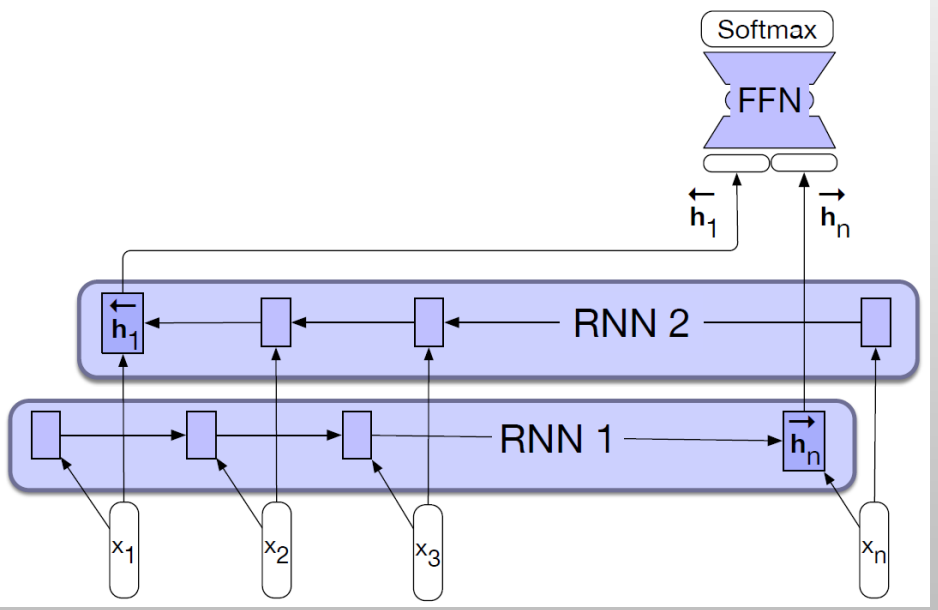


# Stacked RNN

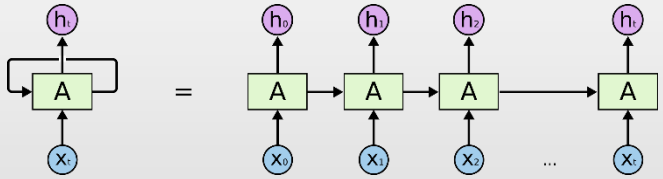


# Bidirectional RNN



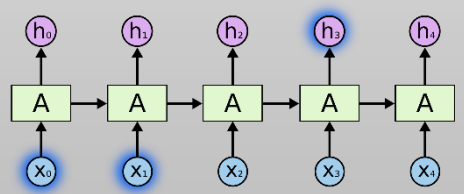



# Long-term Dependency Problem

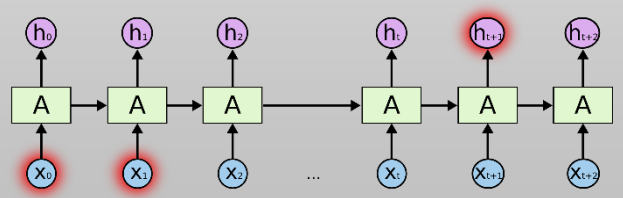



[source]

- Vanilla RNNs have problems with long-term dependencies



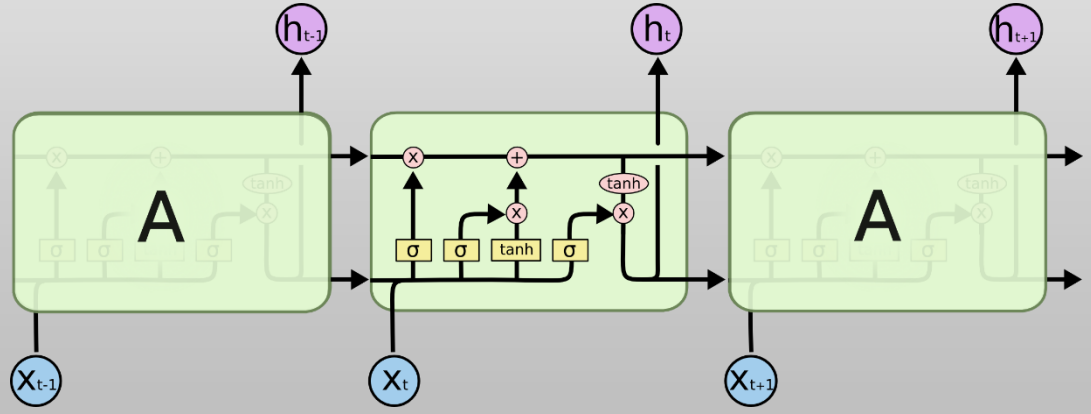

the clouds are in the *sky*

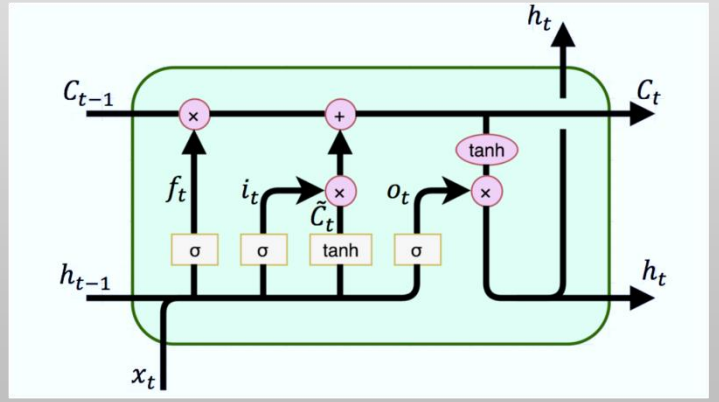



I grew up in France (…) I speak fluent *French*

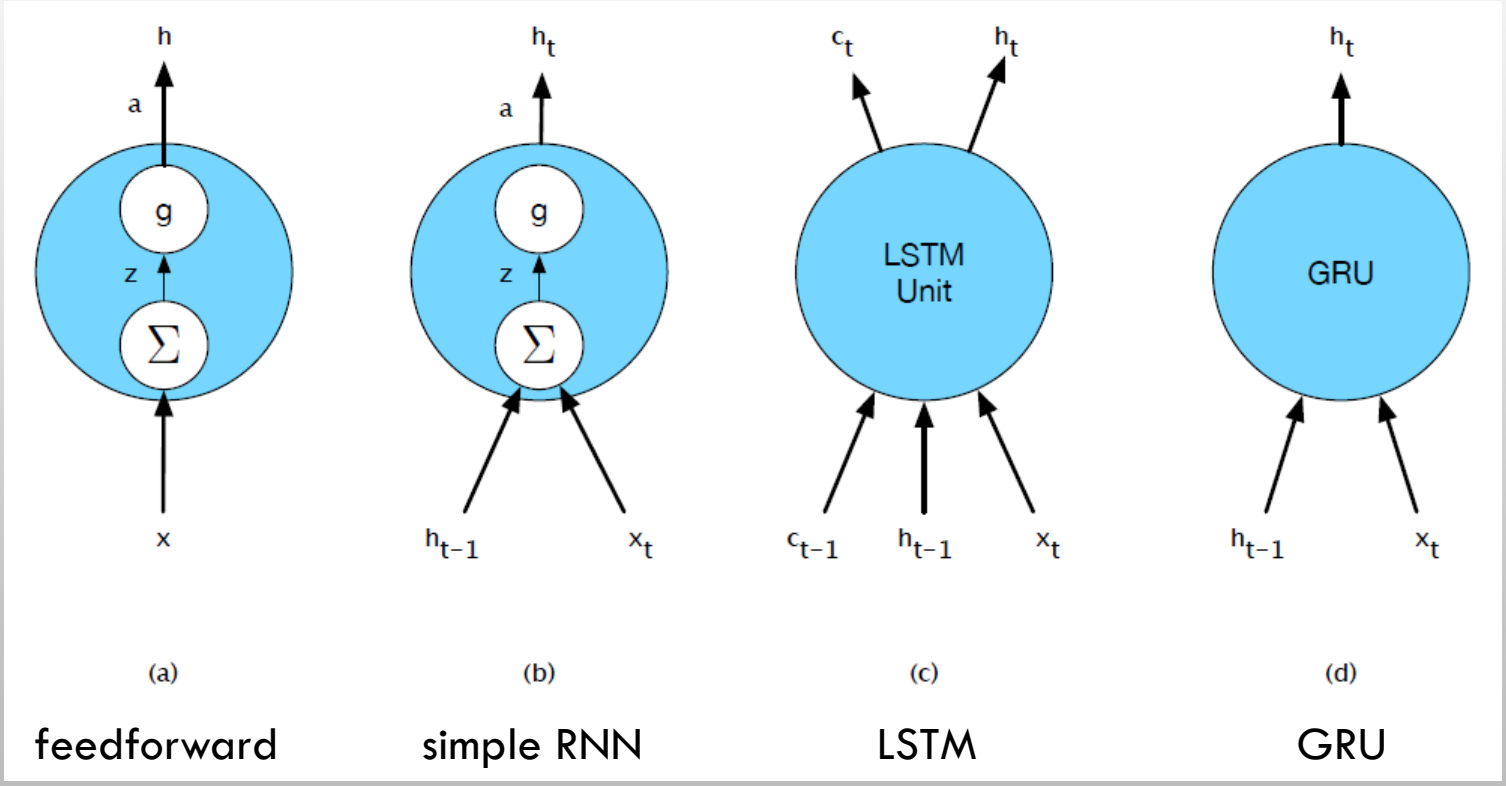
# Long Short-Term Memory (LSTM)

- LSTMs are designed to avoid the long-term dependency problem
  - *The flights the airline was cancelling were full.*



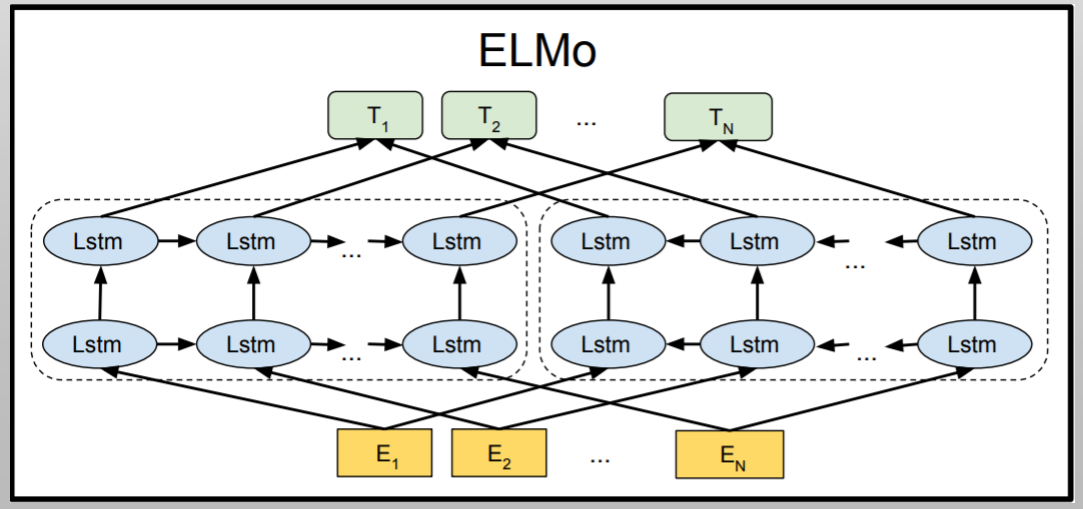
# Gated Units

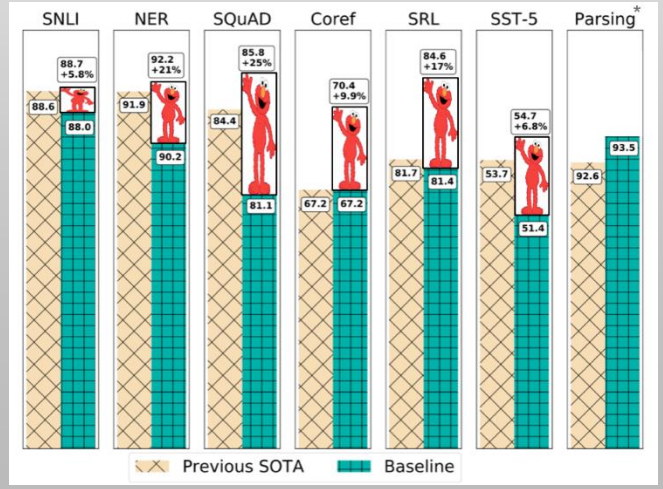

h
a
g
z
Σ
x
(a)
feedforward
$h_t$
a
g
z
Σ
$h_{t-1}$
$x_t$
(b)
simple RNN
$c_t$
$h_t$
LSTM Unit
$c_{t-1}$
$h_{t-1}$
$x_t$
(c)
LSTM
$h_t$
GRU
$h_{t-1}$
$x_t$
(d)
GRU

# Embeddings from Language Models (ELMo)

[Peters et al., 2018]

- Train a forward and a backward LSTM-based language model on some large corpus
- Use the hidden states for each token to compute a vector representation of each word



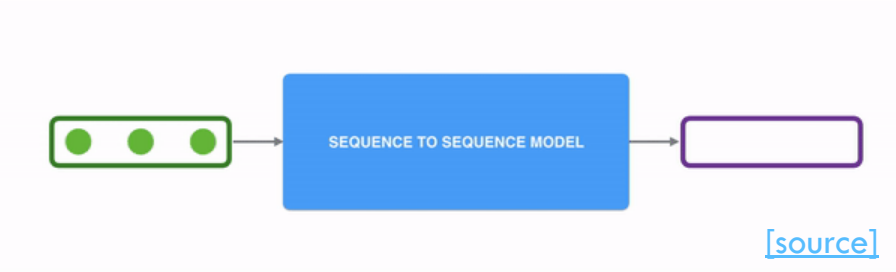
# Neural Language Models

- Word embeddings are context-agnostic
- Language models pretrained on a large corpus capture a lot of context-specific information

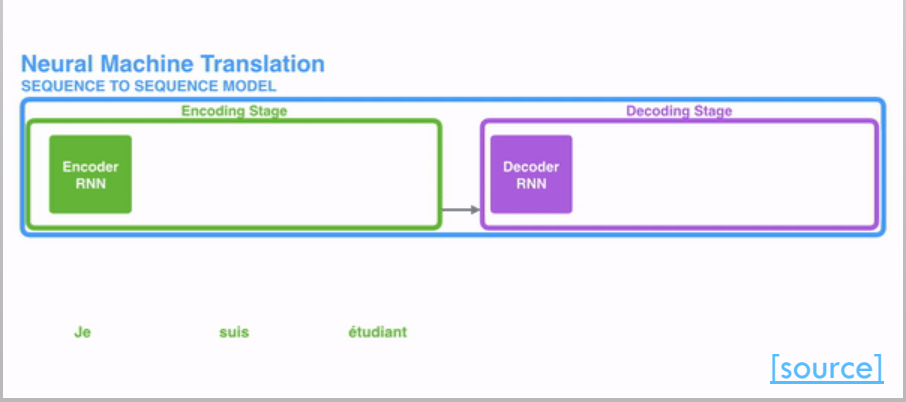
- Language model embeddings can be used as features in a target model
  - Contextual word embeddings
- Language models can be fine-tuned on target task data

# Sequence-to-Sequence Models

- A general framework for mapping one sequence to another using a neural network

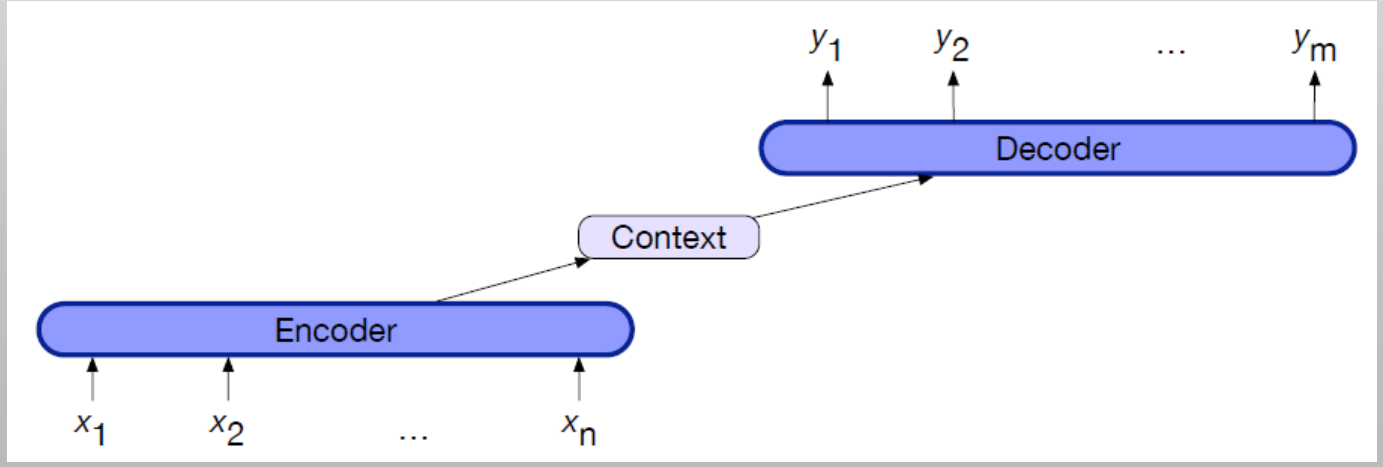



- Useful in many natural language generation tasks: machine translation, summarization, question answering, …

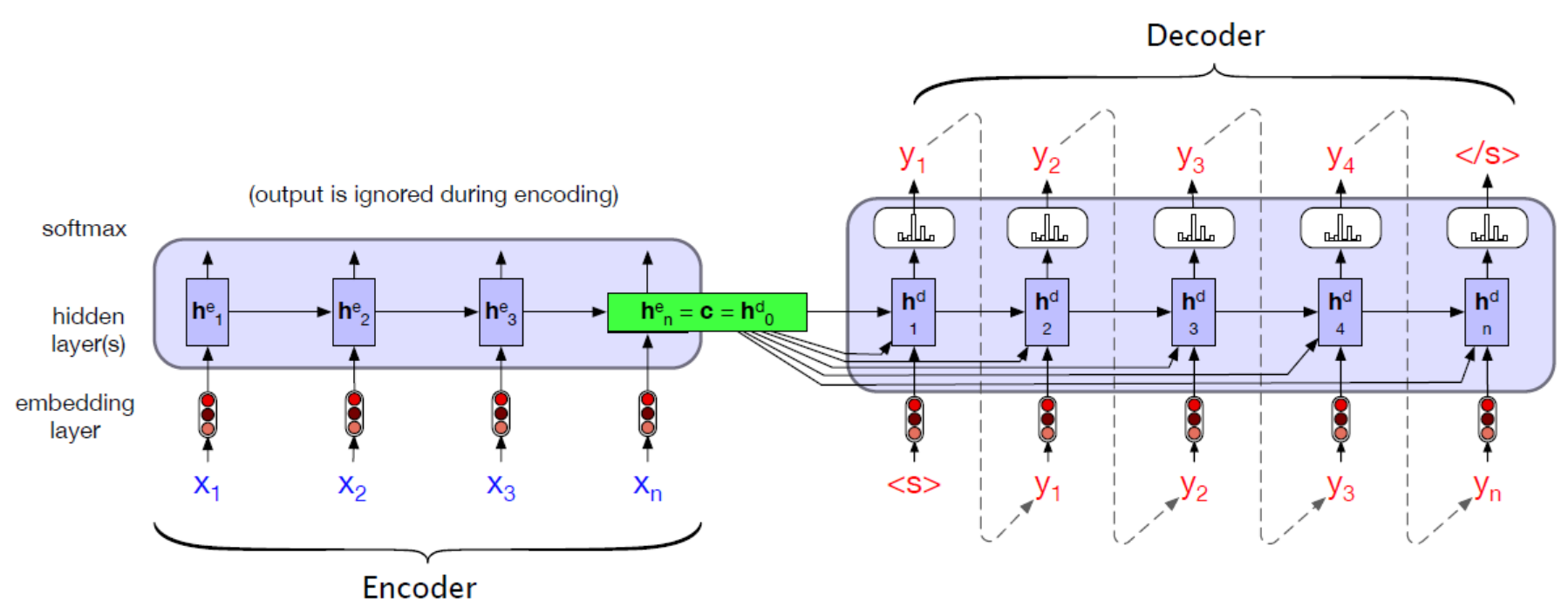


# Sequence-to-Sequence Models

- Main components:
  - Encoder network: processes a sentence token by token and compresses it into a vector representation
  - Decoder network: predicts the output token by token based on the encoder state, taking as input at every step the previously predicted token

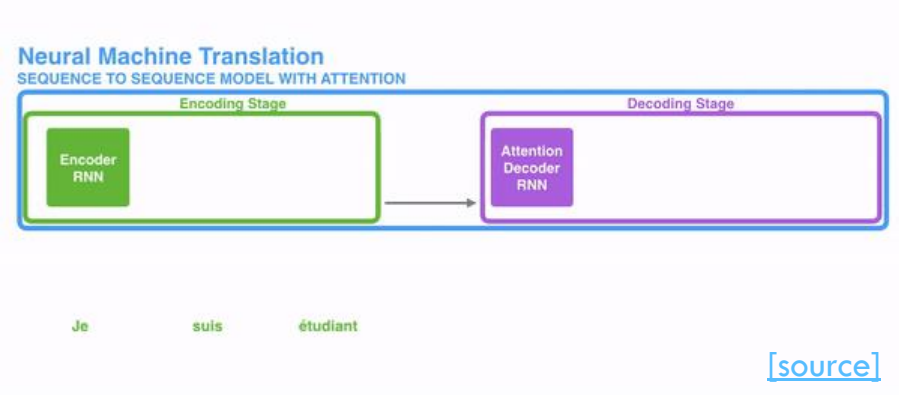


# Encoder-Decoder Networks

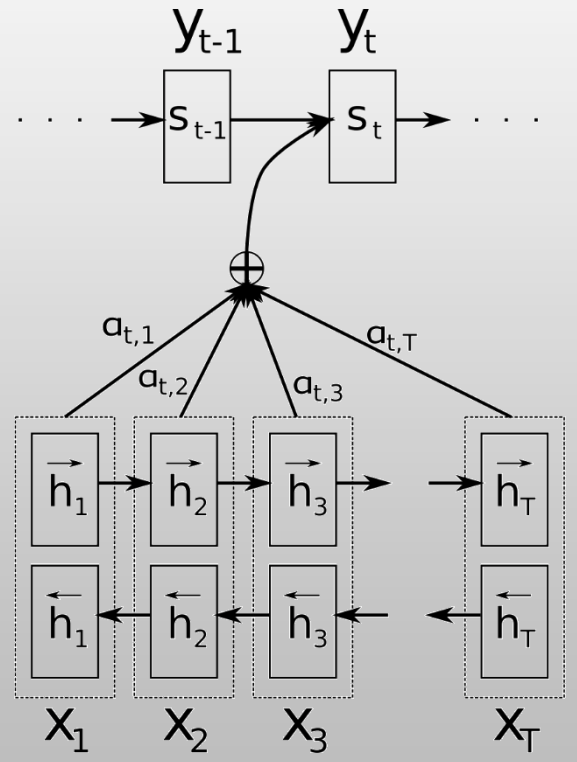


# Attention

- Attention alleviates the "bottleneck" problem of needing to compress the entire context, by allowing the decoder to look back at the source sequence hidden states
- It allows making decisions based on certain parts of the input

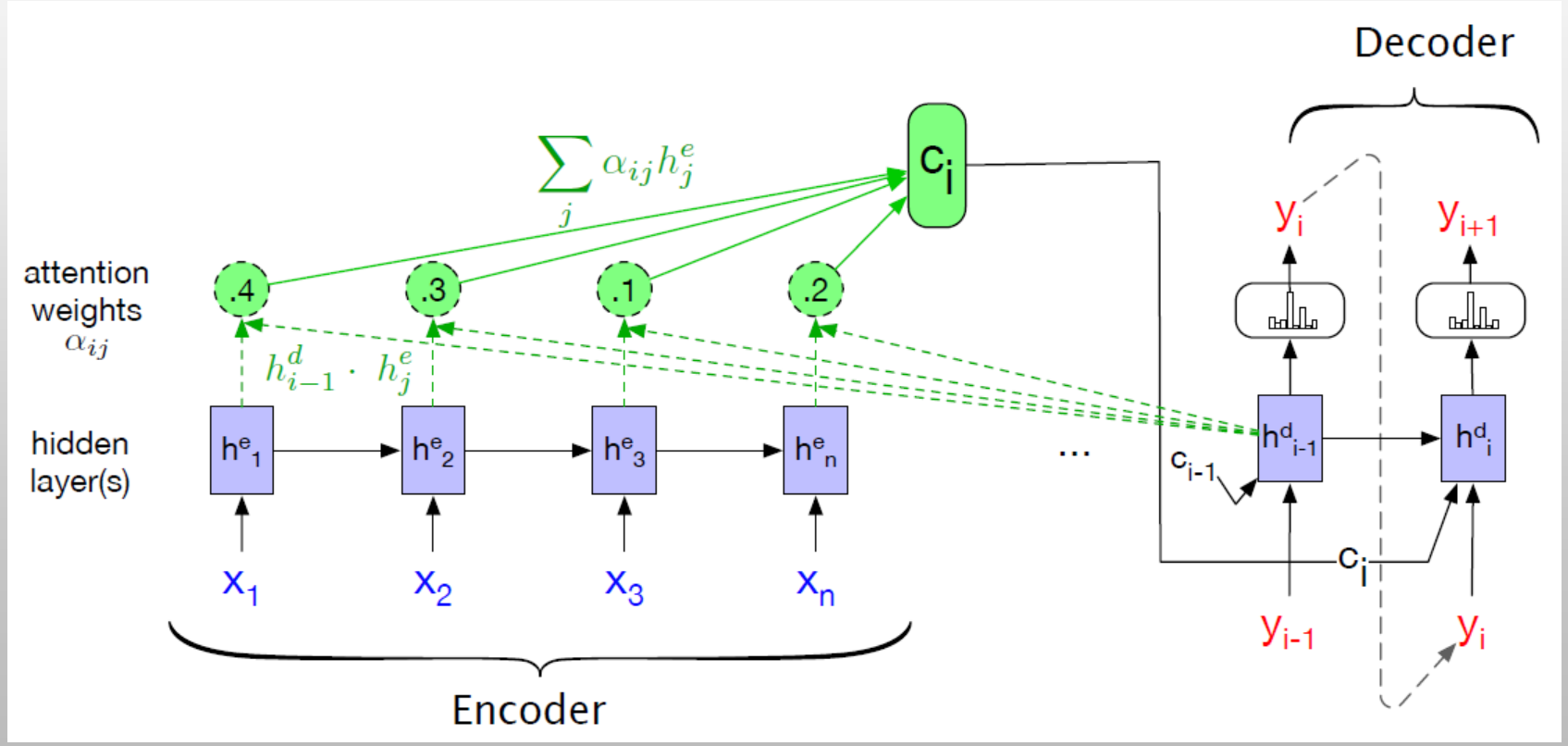



[source]



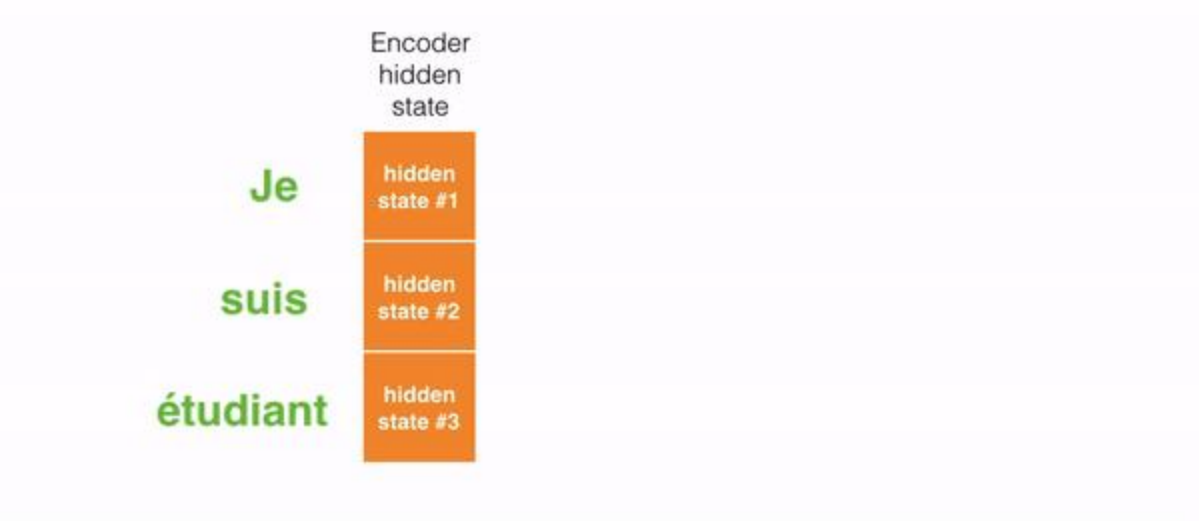

[Bahdanau et al., 2015]

# Attention

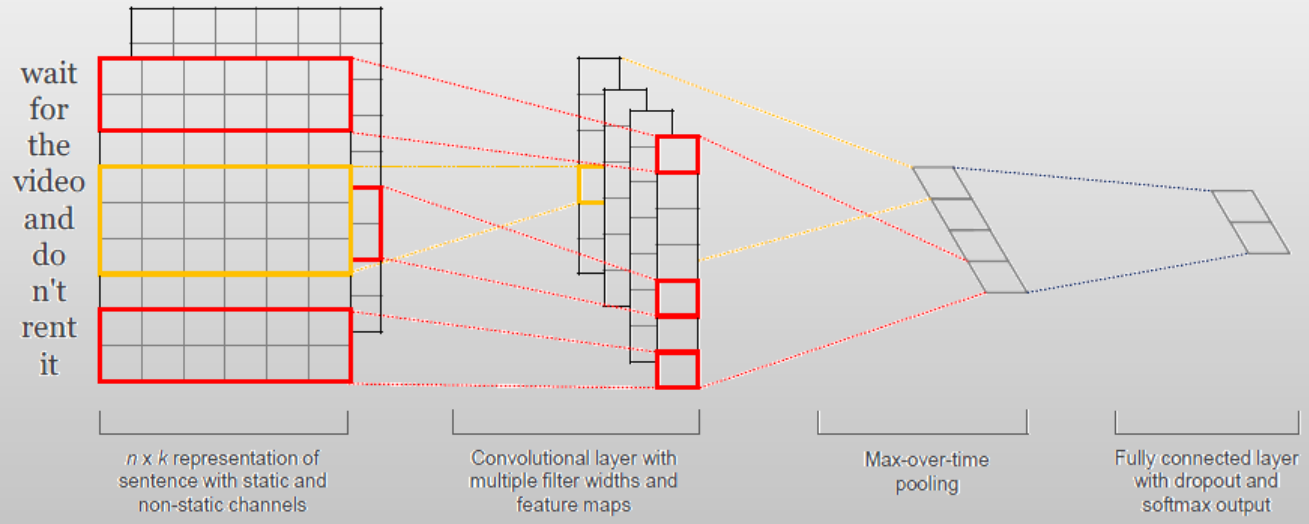

Decoder
$\sum_j \alpha_{ij} h_j^e$
$c_i$
attention weights $\alpha_{ij}$
.4
.3
.1
.2
$h_{i-1}^d \cdot h_j^e$
hidden layer(s)
$h^e_1$
$h^e_2$
$h^e_3$
$h^e_n$
...
$x_1$
$x_2$
$x_3$
$x_n$
Encoder
$y_i$
$y_{i+1}$
$c_{i-1}$
$h^d_{i-1}$
$h^d_i$
$c_i$
$y_{i-1}$
$y_i$

# Attention

- Attention learns the notion of alignment: which source words are more relevant to the current target word?

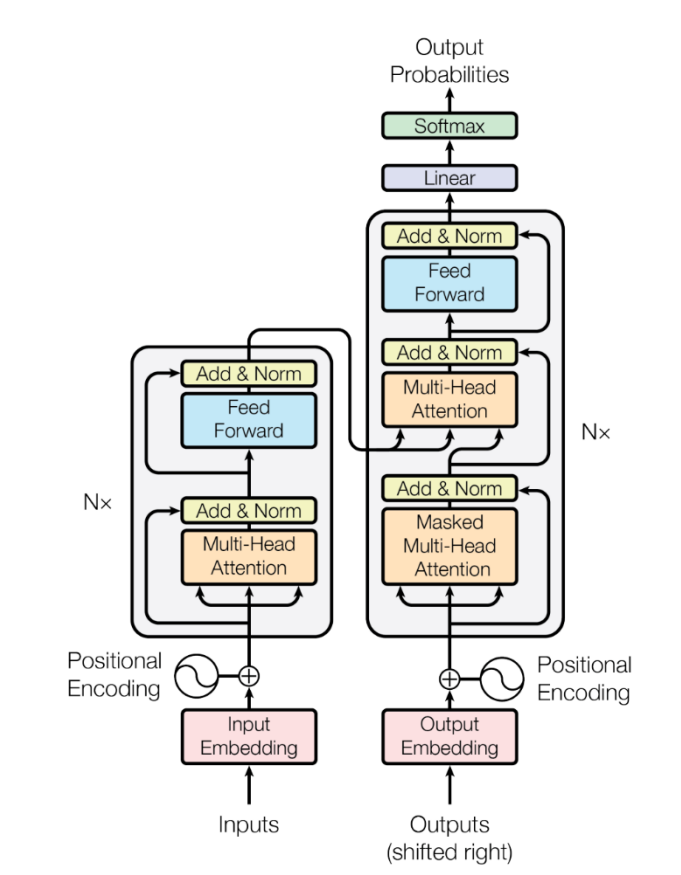


# Convolutional Neural Networks (CNN)

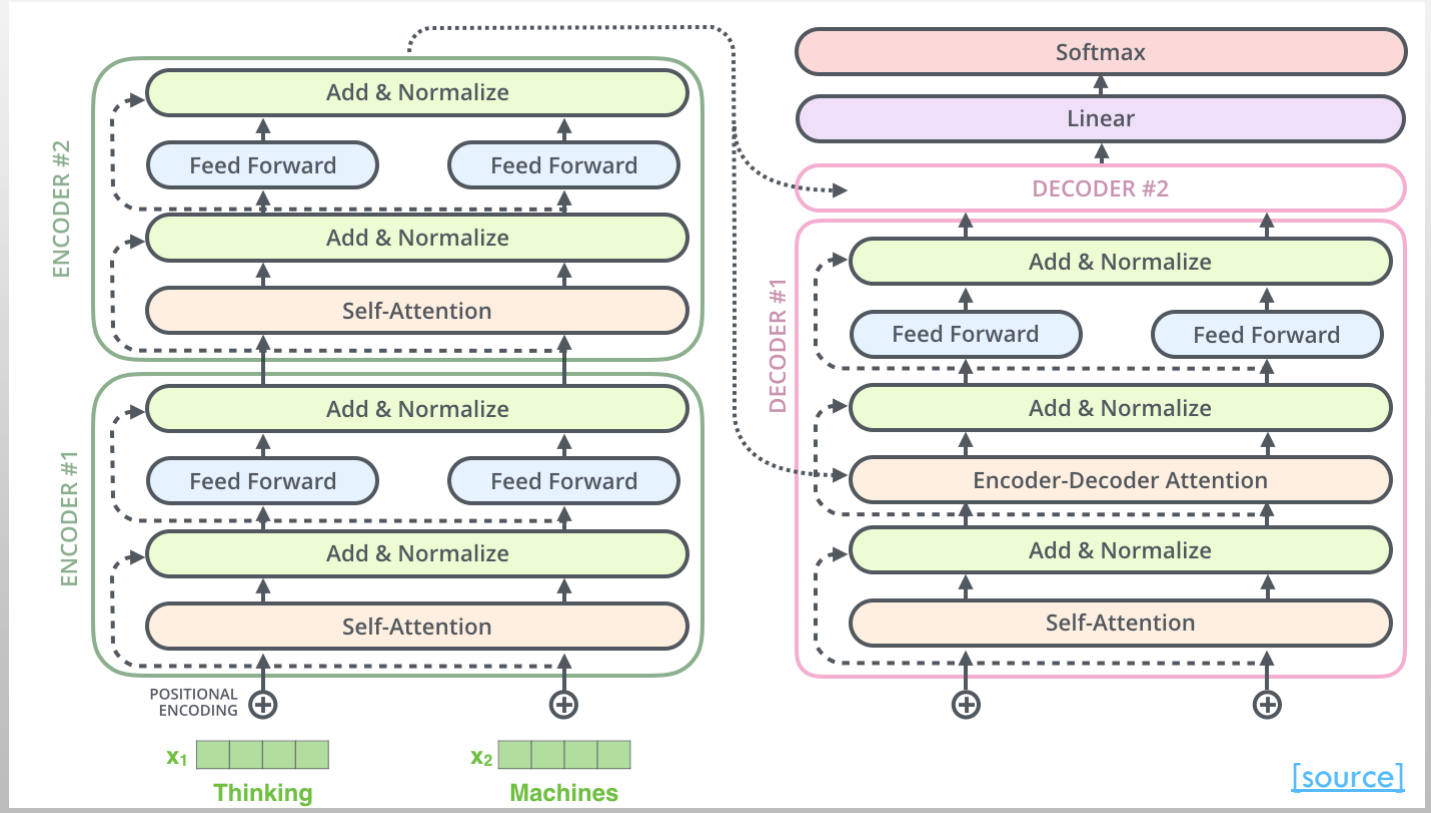



[Kim et al., 2014]

# Transformer

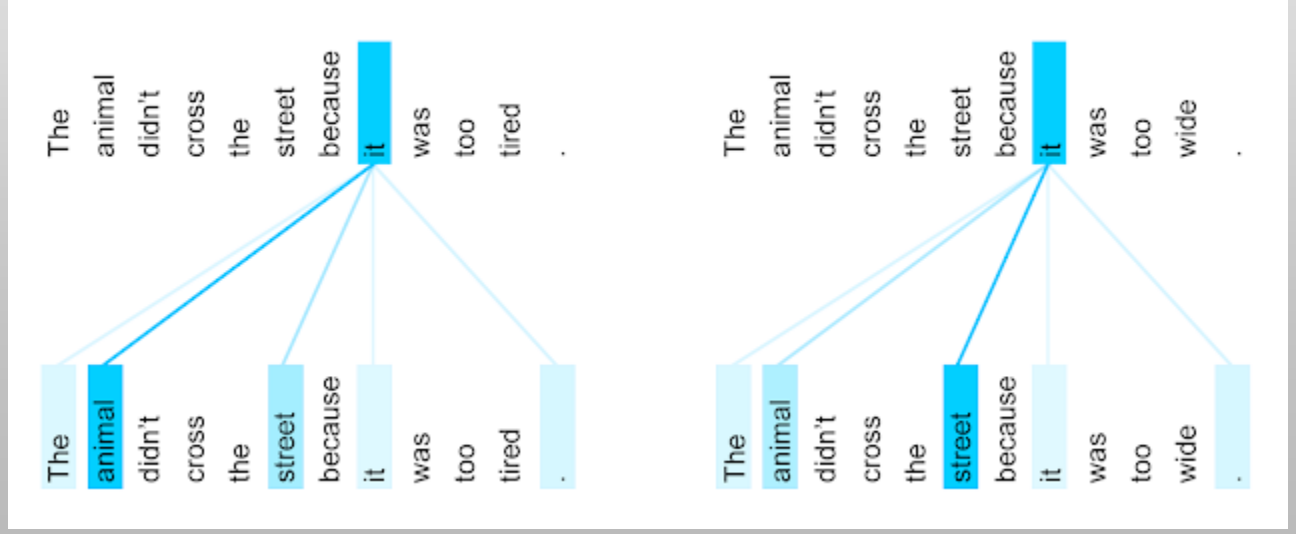



[Vaswani et al., 2017]



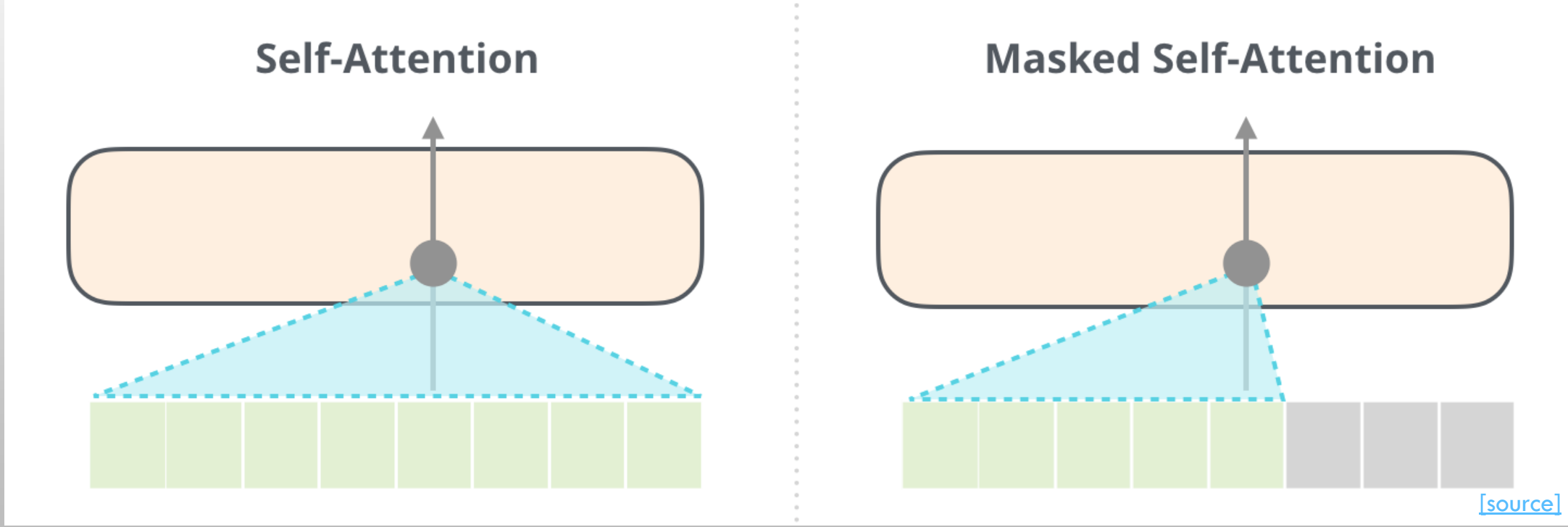
# Self-Attention

- Instead of using the decoder hidden-state to match with encoder hidden states, use each word's representation in the same sequence
- Model the relationships between all words inside a sentence
  - Encoder-encoder attention: each word attends to each other word within the input



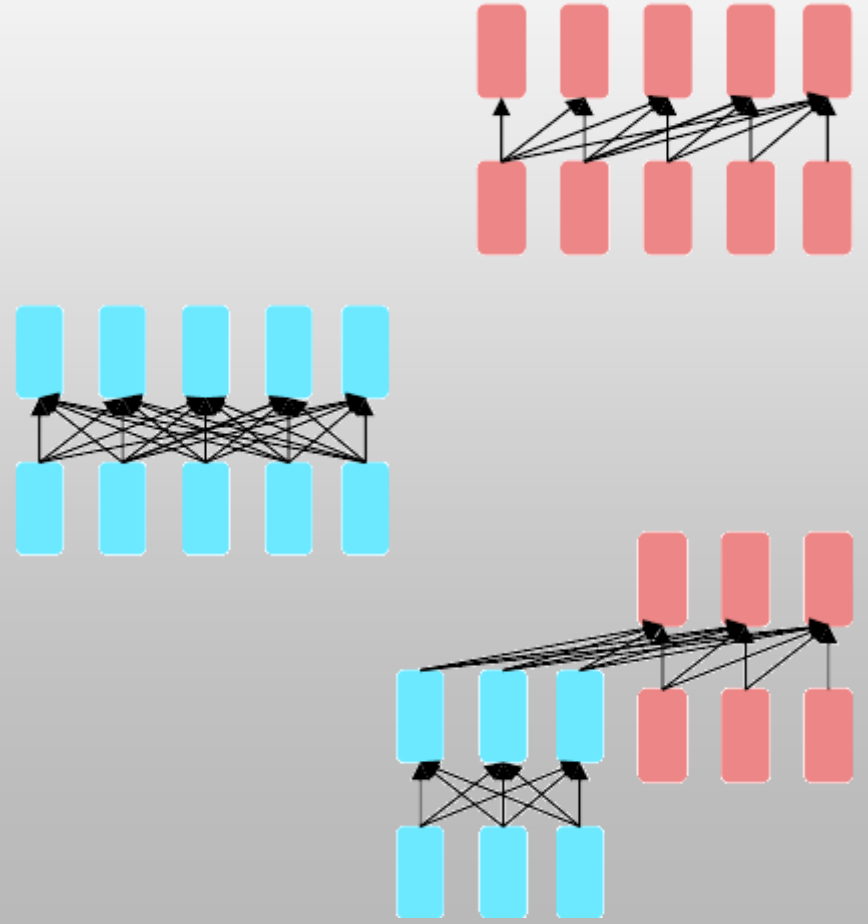

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

# Self-Attention



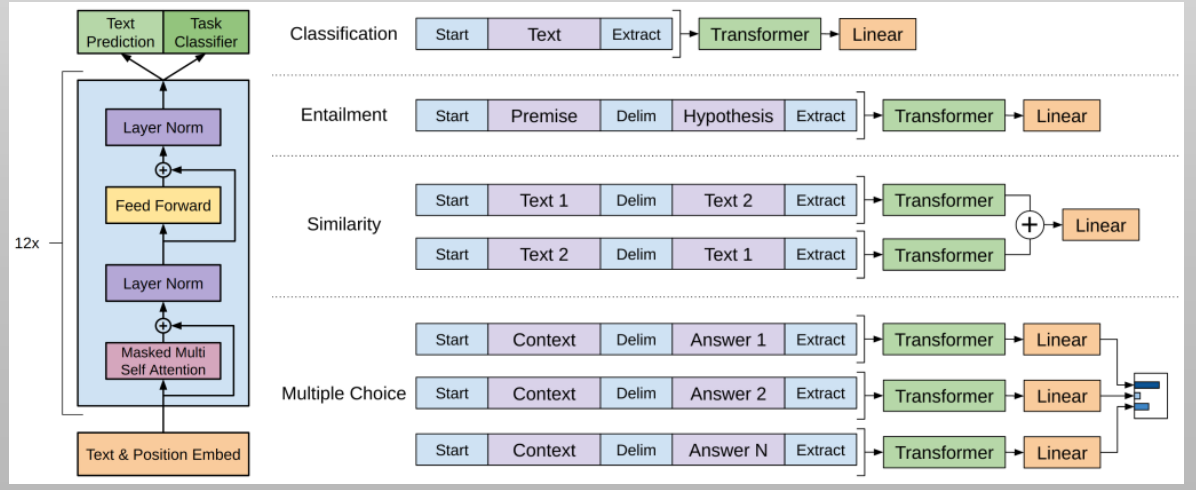
# Transformer-based Architectures

- Decoders: auto-regressive models
  - Language Models conditioned on previous words
  - Example: GPT, GPT-2, GPT-3
- Encoders: auto-encoding models
  - Auto-encoding from bidirectional context
  - Example: BERT and its variants (DistillBERT, RoBERTa, …)
- Encoder-Decoders: sequence-to-sequence models
  - Use the full Transformer model
  - Example: T5, BART



# GPT
# Generative Pre-trained Transformer
[Radford et al., 2018]

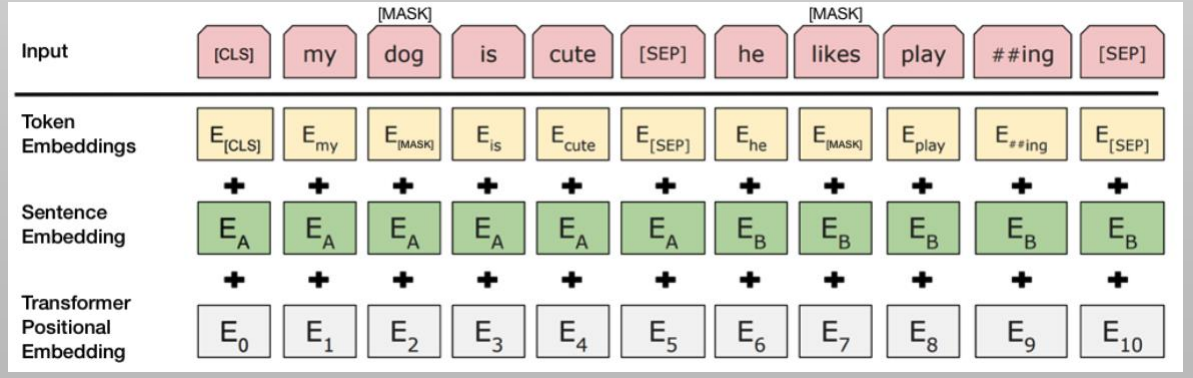
- **Autoregressive** model: Transformer decoder with 12 layers; Byte-pair encoding with 40k merges
- **Pre-trained** as a Language Model on the BooksCorpus (+7000 books, 800M words)
  - Predict next word given the previous words
  - Helpful for text generation tasks
- **Fine-tuning**
  - Task-specific input transformations



# BERT
# Bidirectional Encoder Representations from Transformers

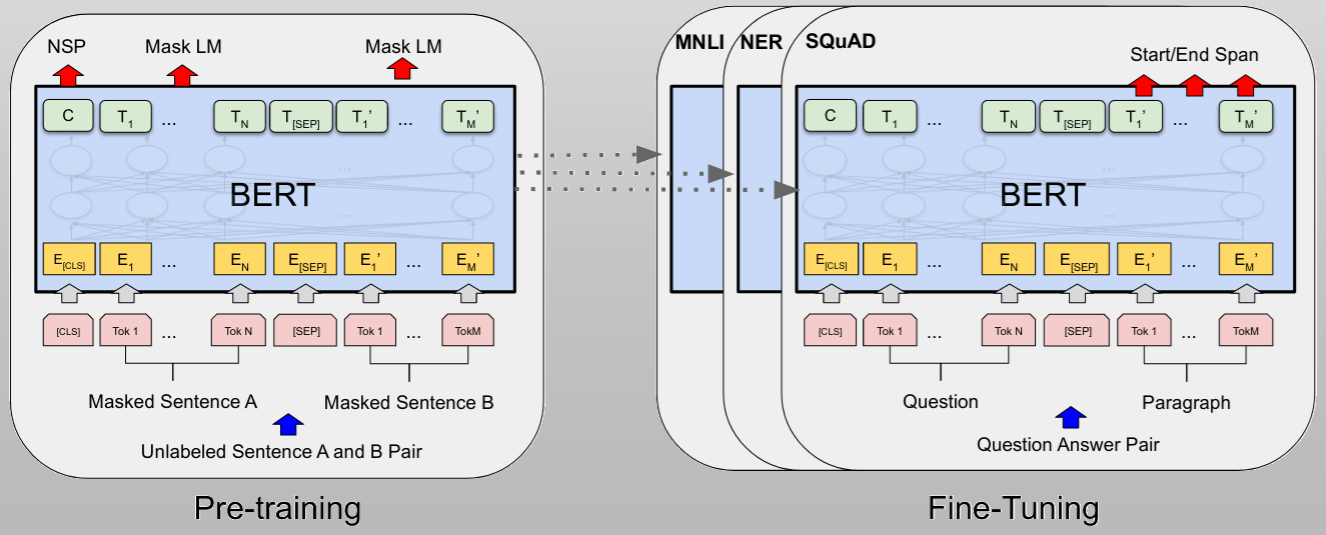
[Devlin et al., 2019]

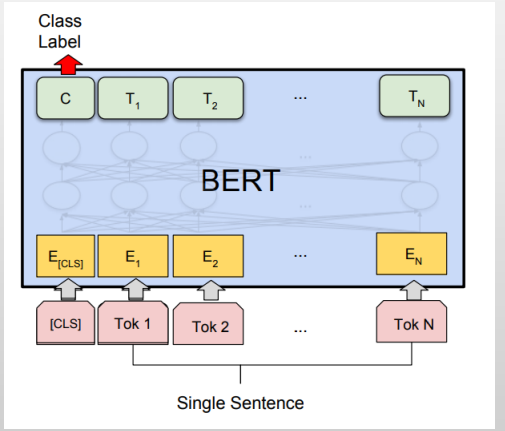
- **Autoencoding** model (bidirectional context) with 12 (BERT-base) or 24 (BERT-large) layers
- Trained on BooksCorpus (800M words) and English Wikipedia (2,500M words)
  - Masked Language Modeling
    - Replace some words with a special [MASK] token, and learn to predict those words
  - Next Sentence Prediction (abandoned in subsequent models, e.g. RoBERTa)
- Input representation:

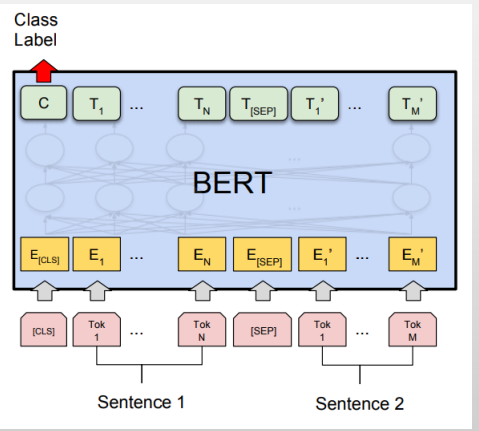


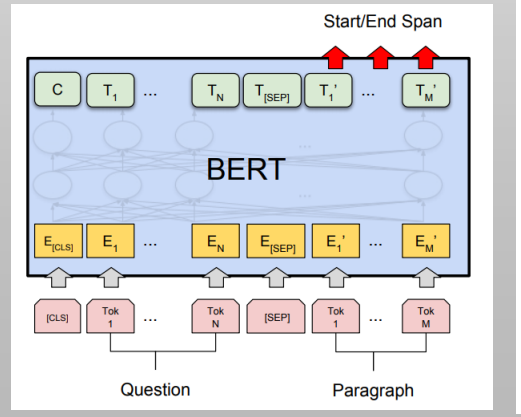
# BERT
# Bidirectional Encoder Representations from Transformers
[Devlin et al., 2019]

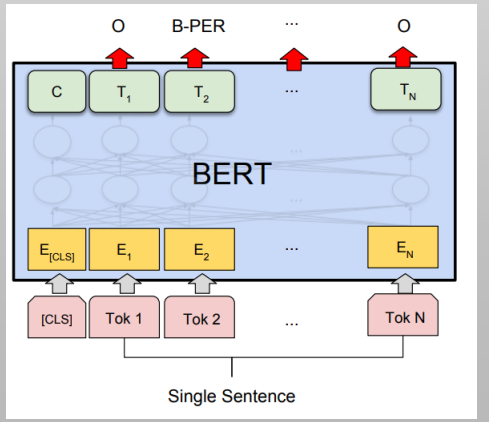
- **Pre-trained** with 16 (BERT-base) or 64 (BERT-large) TPU chips for 4 days
- **Fine-tuning** (doable on a single GPU in a few hours)



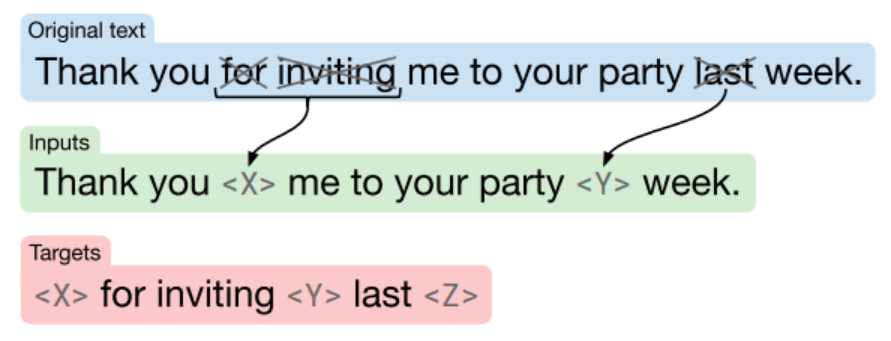



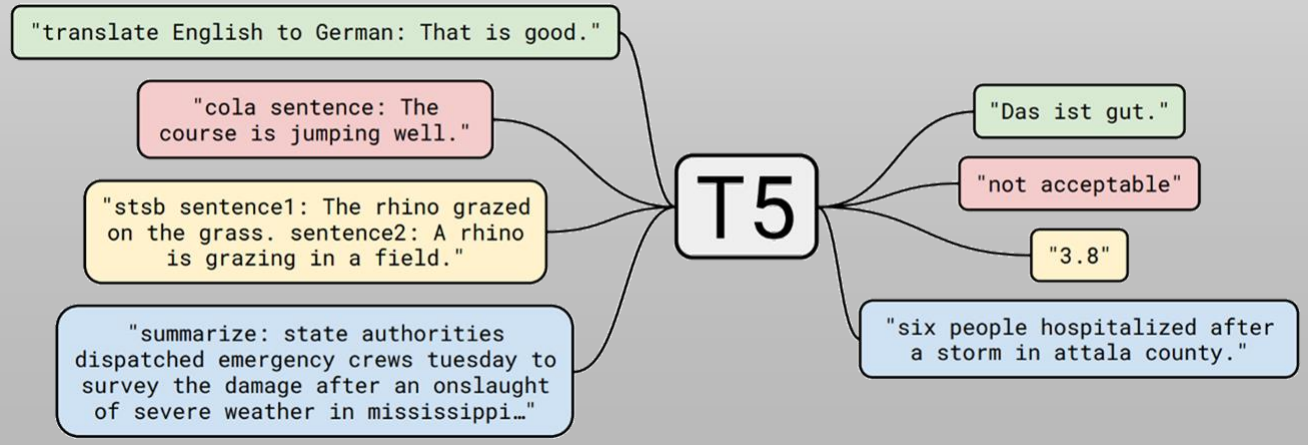



# T5
# Text-to-Text Transfer Transformer

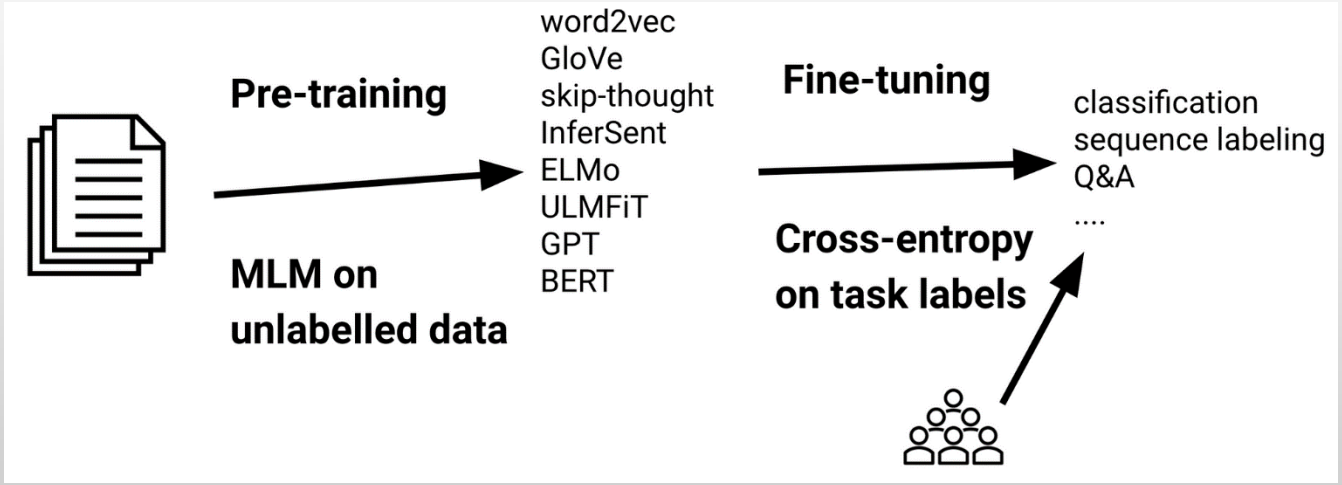
[Raffel et al., 2020]

- Encoder-decoder
- **Pre-trained** on 700GB C4 corpus
  - Language Model, but where the prefix of every input is provided (not predicted)
  - Span-corruption (masking)
- **Fine-tuned** on downstream tasks in a multi-task text-to-text format, with a task-specific (text) prefix added to the original input sequence



# Pre-training and Fine-tuning




https://ruder.io/recent-advances-lm-fine-tuning/

# Bias


## Language models are more capable than ever, but also more biased

Large language models are setting new records on technical benchmarks, but new data shows that larger models are also more capable of reflecting biases from their training data. **A 280 billion parameter model developed in 2021 shows a 29% increase in elicited toxicity over a 117 million parameter model considered the state of the art as of 2018.** The systems are growing significantly more capable over time, though as they increase in capabilities, so does the potential severity of their biases.

# Four paradigms in NLP

[Liu et al., 2021]

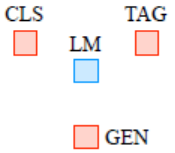
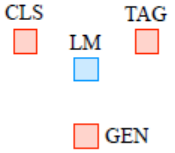
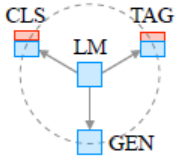
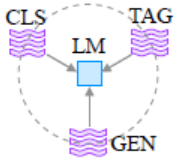
| Paradigm | Engineering | Task Relation |
|---|---|---|
| a. Fully Supervised Learning (Non-Neural Network) | Features (e.g. word identity, part-of-speech, sentence length) |   |
| b. Fully Supervised Learning (Neural Network) | Architecture (e.g. convolutional, recurrent, self-attentional) |   |
| c. Pre-train, Fine-tune | Objective (e.g. masked language modeling, next sentence prediction) |   |
| d. Pre-train, Prompt, Predict | Prompt (e.g. cloze, prefix) |   |

Table 1: Four paradigms in NLP. The "**engineering**" column represents the type of engineering to be done to build strong systems. The "**task relation**" column, shows the relationship between language models (LM) and other NLP tasks (CLS: classification, TAG: sequence tagging, GEN: text generation). : fully unsupervised training. : fully supervised training. : Supervised training combined with unsupervised training. indicates a textual prompt. Dashed lines suggest that different tasks can be connected by sharing parameters of pre-trained models. "LM→Task" represents *adapting LMs (objectives) to downstream tasks* while "Task→LM" denotes *adapting downstream tasks (formulations) to LMs.*

# Pretrain, Prompt, Predict

[Liu et al., 2021]

| Type | Task | Input ( [X] ) | Template | Answer ( [Z] ) |
|---|---|---|---|---|
| Text CLS | Sentiment | I love this movie. | [X] The movie is [Z]. | great<br>fantastic<br>... |
| | Topics | He prompted the LM. | [X] The text is about [Z]. | sports<br>science<br>... |
| | Intention | What is taxi fare to Denver? | [X] The question is about [Z]. | quantity<br>city<br>... |
| Text-span CLS | Aspect Sentiment | Poor service but good food. | [X] What about service? [Z]. | Bad<br>Terrible<br>... |
| Text-pair CLS | NLI | [X1]: An old man with ...<br>[X2]: A man walks ... | [X1]? [Z], [X2] | Yes<br>No<br>... |
| Tagging | NER | [X1]: Mike went to Paris.<br>[X2]: Paris | [X1][X2] is a [Z] entity. | organization<br>location<br>... |
| Text Generation | Summarization | Las Vegas police ... | [X] TL;DR: [Z] | The victim ...<br>A woman ...<br>... |
| | Translation | Je vous aime. | French: [X] English: [Z] | I love you.<br>I fancy you.<br>... |

# Conclusions

- Neural NLP models are the rule
- RNN → seq2seq → seq2seq + attention → Transformers
- Fully supervised → Pre-train and fine-tune → Prompt engineering
- Large Language Models
  - OpenAI: GPT, GPT-2, GPT-3, ChatGPT, GPT-4
  - DeepMind: Chinchilla, Sparrow
  - Google: BERT, T5, PaLM, Bard
  - FB/Meta: BART, OPT, LLaMA

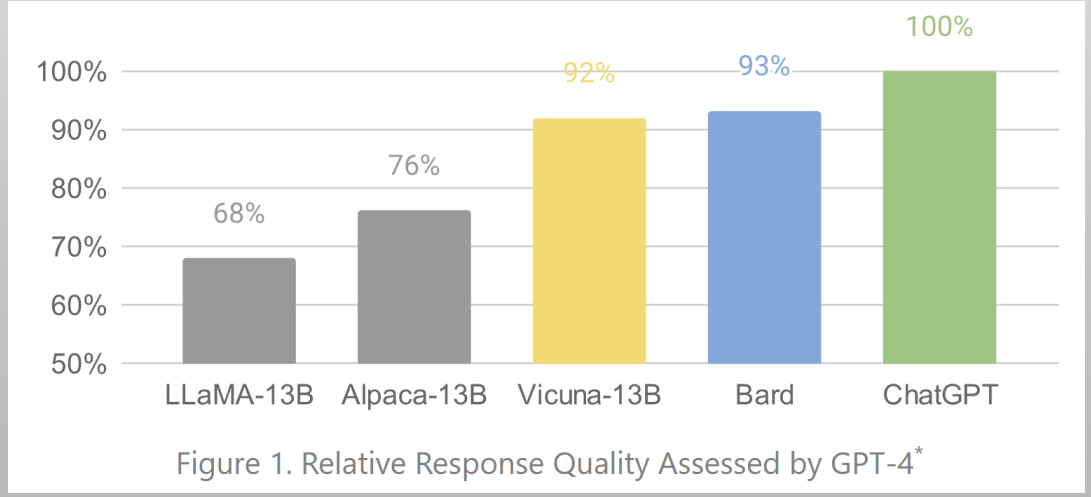



Figure 1. Relative Response Quality Assessed by GPT-4*

https://vicuna.lmsys.org/