

Deep Learning

08 - Transformers



SAPIENZA
UNIVERSITÀ DI ROMA

Fabrizio Silvestri

Modelling Sequence Data



SAPIENZA
UNIVERSITÀ DI ROMA

Examples of Sequence data

Speech Recognition

Machine Translation

Language Modeling

Named Entity Recognition

Sentiment Classification

Video Activity Analysis

WHAT'S
NEXT?

Input Data



Hello, I am Pankaj.

Recurrent neural ? based ? model

Pankaj lives in Munich

There is nothing to like in this movie.



Output

This is RNN

Hallo, ich bin Pankaj.
हैलो, मैं पंकज हूँ।

network

language

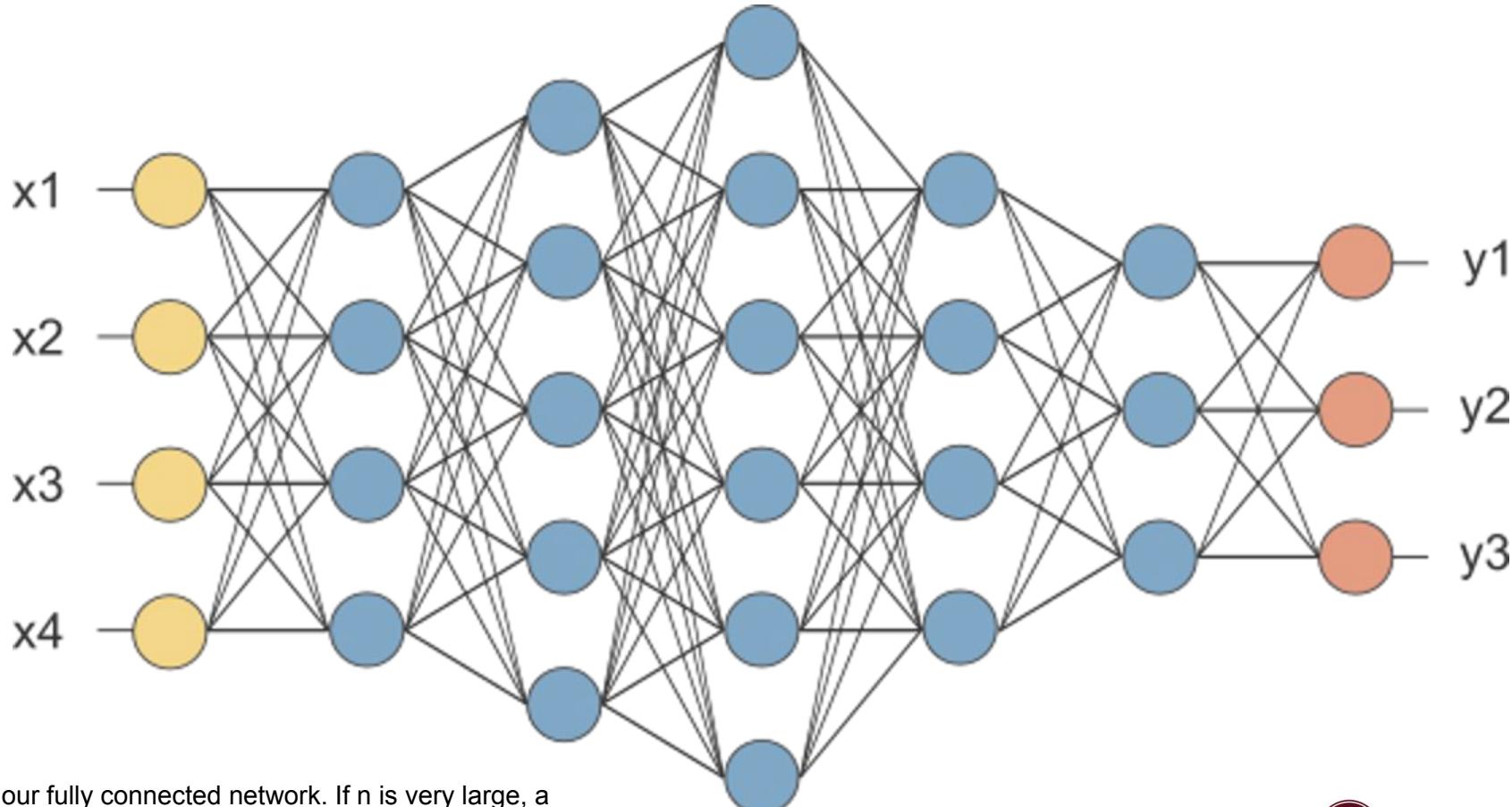
Pankaj lives in Munich
person location



Punching



SAPIENZA
UNIVERSITÀ DI ROMA



This is our fully connected network. If n is very large, a FCN would become prohibitively large. Idea: input one x_i at a time, and re-use the same edge weights



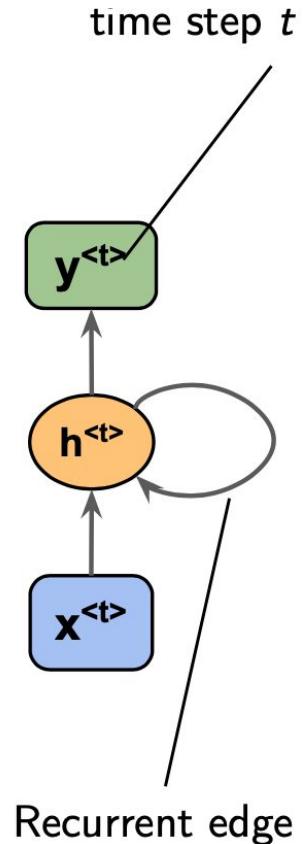
Recurrent Neural Networks



SAPIENZA
UNIVERSITÀ DI ROMA

The General Idea

- A recurrent neural network is a neural network that is specialized for processing a sequence of values x_1, \dots, x_T .
- Parameter sharing enables the model for examples of different lengths and generalize across them.



Unfolding

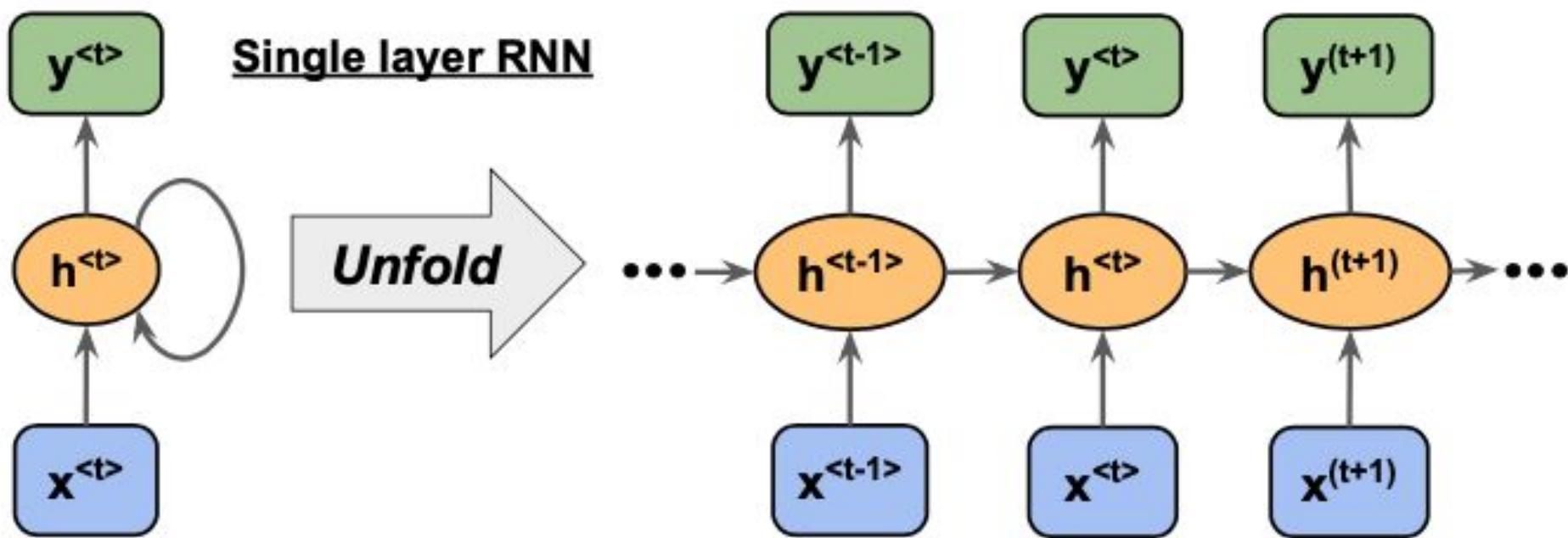
- Consider the classical form of a dynamical system:

$$\underbrace{h^{}}_{\text{cell state}} = f_W(\underbrace{h^{}}_{\text{old state}}, \underbrace{x^{}}_{\text{input vector}})$$

- **recurrent** because the definition of s at time t refers back to the same definition at time $t-1$.
- For a finite number of time steps τ , the formula can be **unfolded** by applying the definition $\tau-1$ times



Unfolding



Weight Matrices

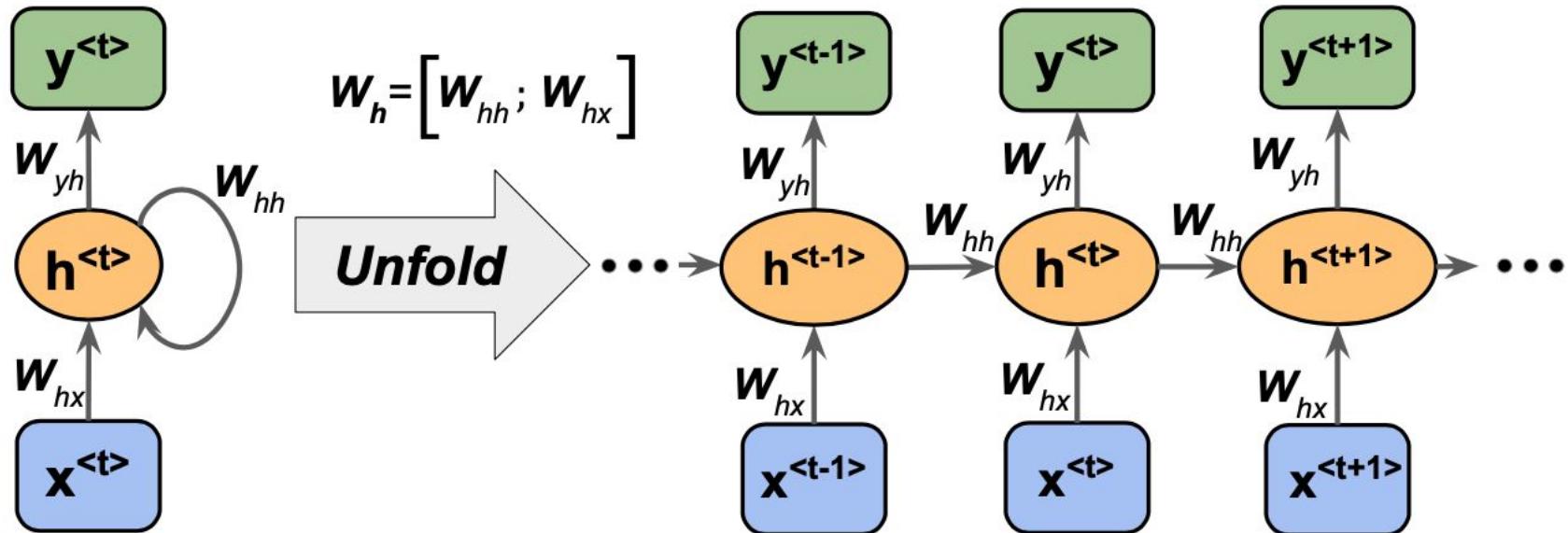


Figure: Sebastian Raschka, Vahid Mirjalili. *Python Machine Learning*. 3rd Edition. Birmingham, UK: Packt Publishing, 2019



Vanilla RNN

- A possible instantiation of:

$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}; \boldsymbol{\theta})$$

- Is the following

$$h_t = g_{\boldsymbol{\theta}}(h_{t-1}, x_t)$$
$$\downarrow$$

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t)$$

$$o_t = \text{softmax}(W_{ho} h_t)$$



Backpropagation Through Time



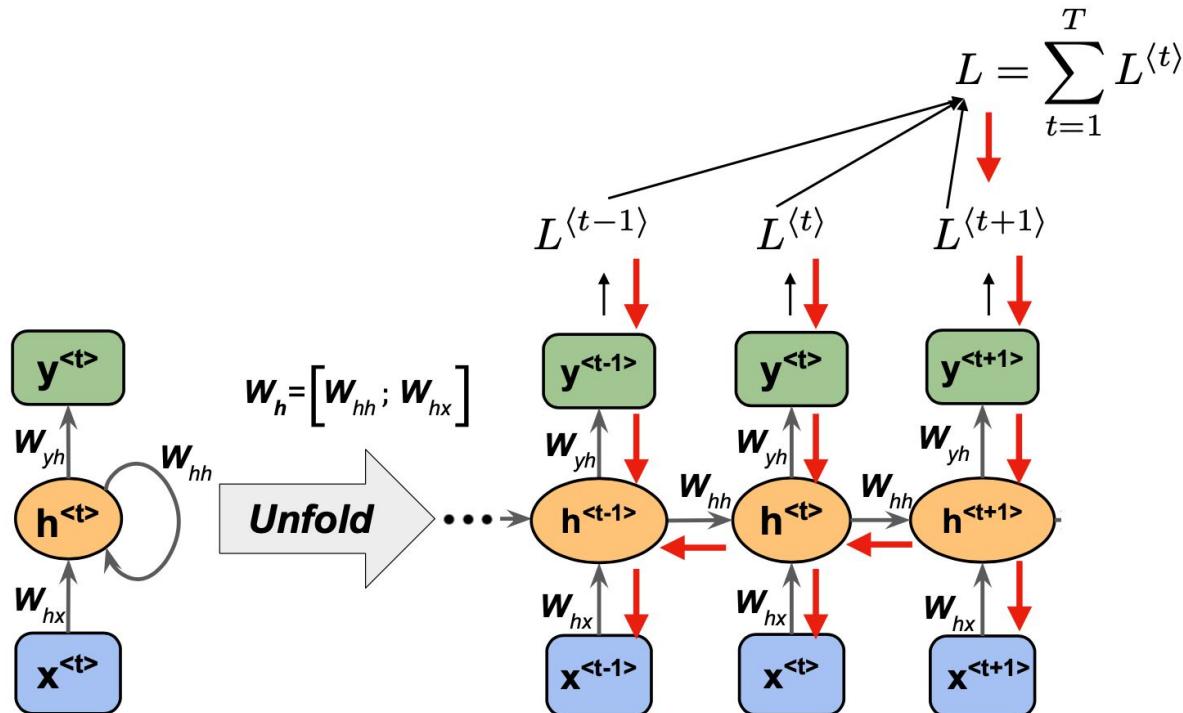
SAPIENZA
UNIVERSITÀ DI ROMA

How to Train a Recurrent NN?

- The training data for a recurrent neural network is an ordered sequence of k input-output pairs, $\langle x_0, y_0 \rangle, \langle x_1, y_1 \rangle, \dots, \langle x_{k-1}, y_{k-1} \rangle$.
 - An initial value must be specified for the hidden state h_0 .
 - Typically, a vector of all zeros is used for this purpose.
- We begin by unfolding a recurrent neural network in time.
 - Every copy of the network shares the same parameters.
- We can now use Backprop taking extra care of the shared parameters
 - **Backpropagation Through Time (BPTT)**.



Backpropagation Through Time



Sequence-to-Sequence (Seq2Seq)



SAPIENZA
UNIVERSITÀ DI ROMA

The General Idea

- RNNs are at the basis of a general idea of producing ML models that learn to **transform an input sequence into an output sequence**
- Examples:
 - Translation: I am Italian → Io sono italiano
 - Captioning:  → Picture of a flying plane
 - Aging:  → 



Types of Seq2Seq

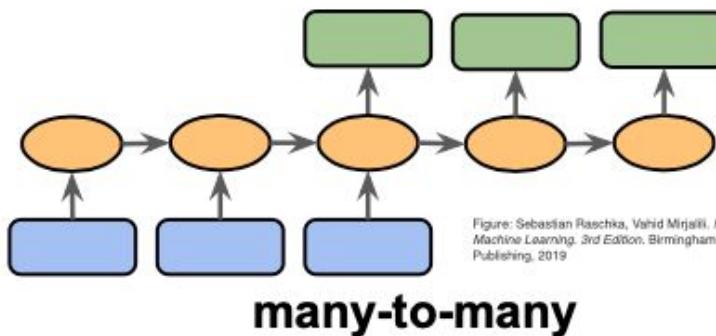
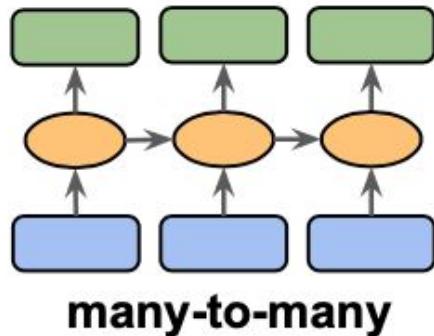
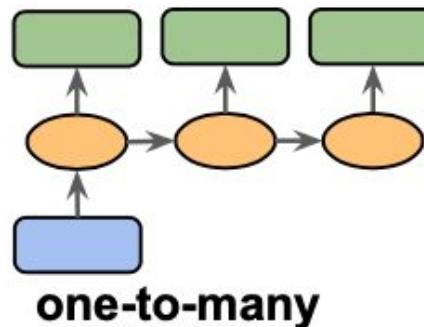
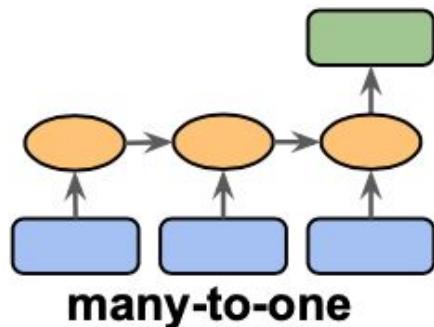
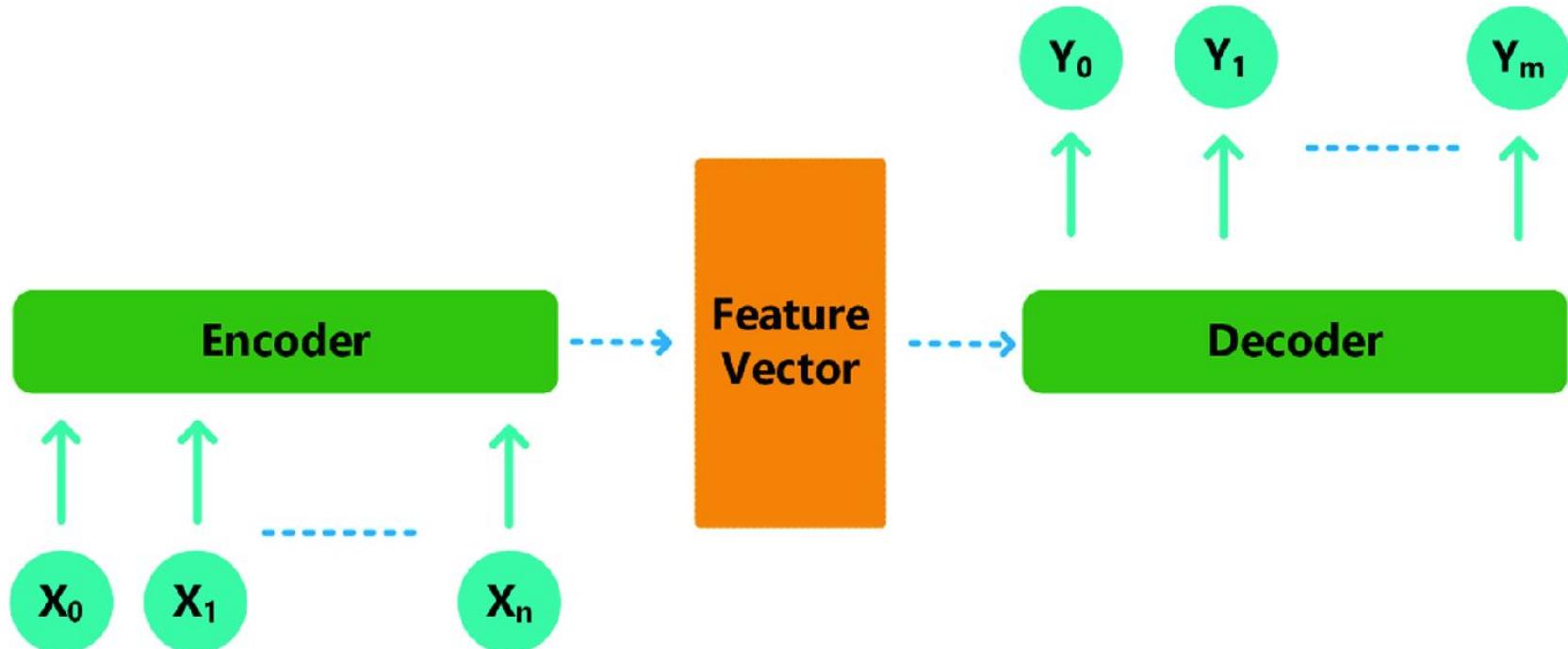


Figure: Sebastian Raschka, Vahid Mirjalili. Python
Machine Learning. 3rd Edition. Birmingham, UK: Packt
Publishing, 2019

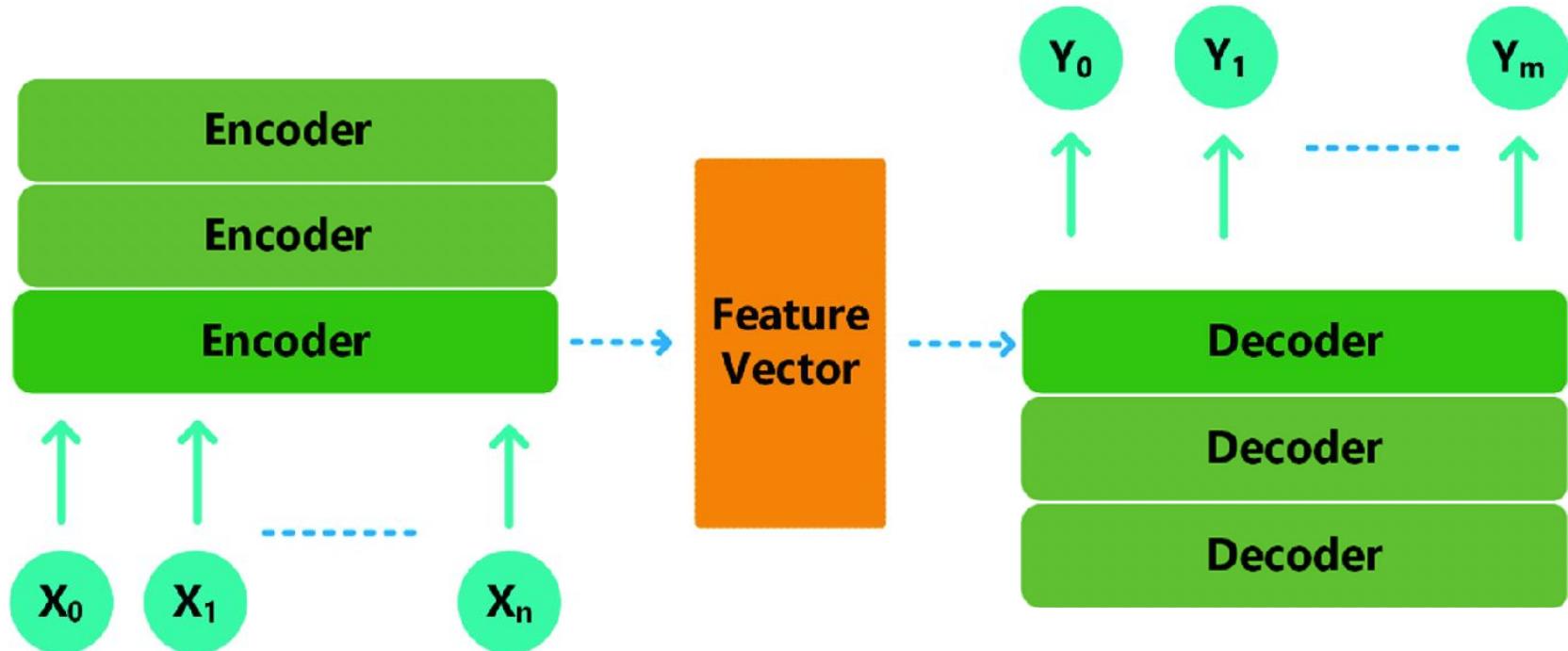


SAPIENZA
UNIVERSITÀ DI ROMA

The Encoder-Decoder Paradigm



The Encoder-Decoder Paradigm (DL Version)



Autoregressive Models



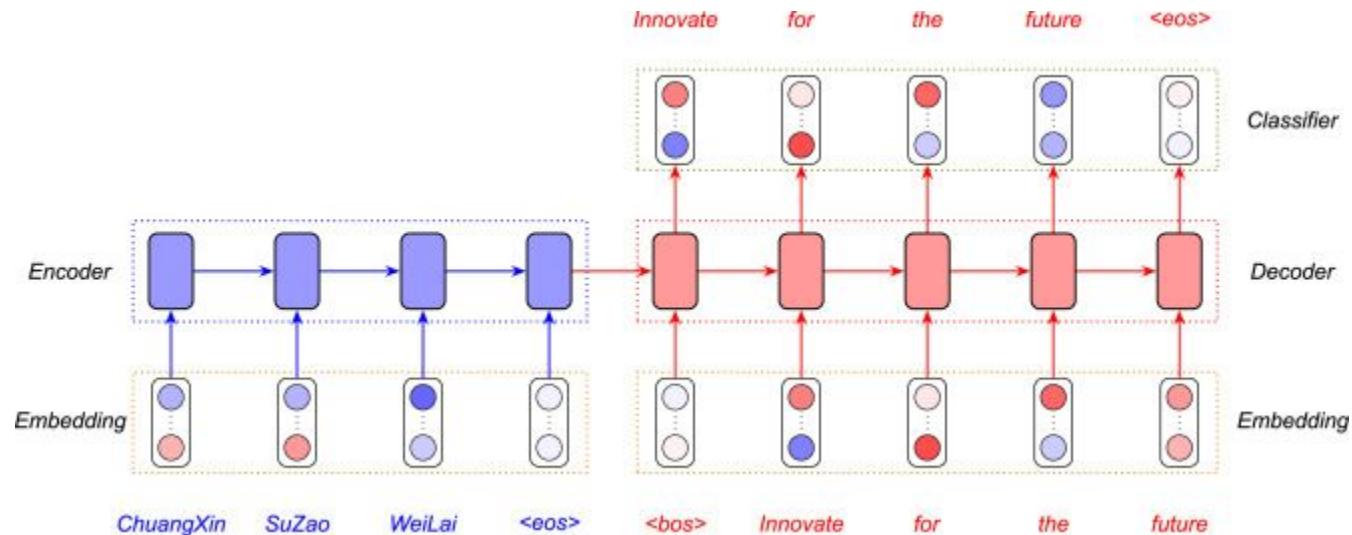
SAPIENZA
UNIVERSITÀ DI ROMA

What Happens after the Encoding Stage?

- The Encoder generates a representation for the input sequence.
- The decoder takes this representation and generates the output sequence.

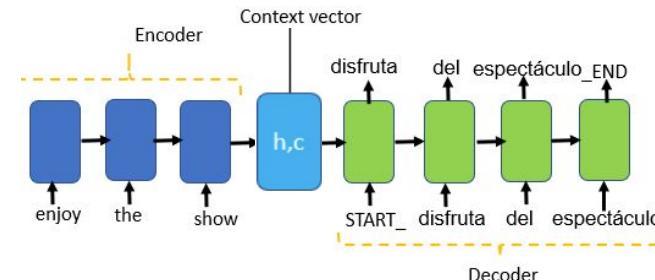
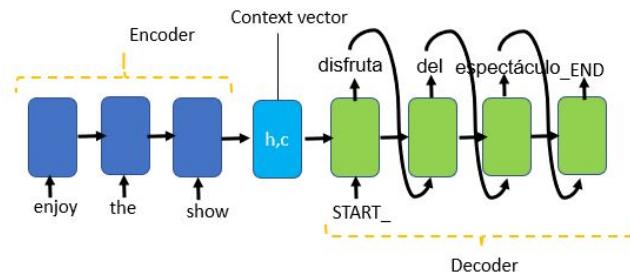


The Example of Language Translation



Teacher Forcing

- At decoding stage, the only input we have is that we generated so far...
 - ... not at training time.
- During training, with probability p feed in input the ground truth item of the sequence (with prob. 1-p the generated one)



Teacher forcing -Actual target words are sent as an input to the decoder



SAPIENZA
UNIVERSITÀ DI ROMA

Word Embeddings



SAPIENZA
UNIVERSITÀ DI ROMA

Vexilora (vek-sih-LOHR-ah)

- What does “Vexilora” (vek-sih-LOHR-ah) mean?
- Phonetically:
 - vek rhymes with "deck"
 - sih like the first syllable in "silver"
 - LOHR rhymes with "more" or "core," emphasized syllable
 - ah as in "sofa"
- IPA transcription: /vɛk.sɪ'lo:r.ə/



Vexilora (n)

- A rare mineral with magnetic properties
- Example: Vexilora embedded in a rock



magnetic
Vexilora



SAPIENZA
UNIVERSITÀ DI ROMA

Ve



ra,

Vexilora (n)

- An advanced quantum computing
- Example: potential o



um
ability.
the full



SAPIENZA
UNIVERSITÀ DI ROMA

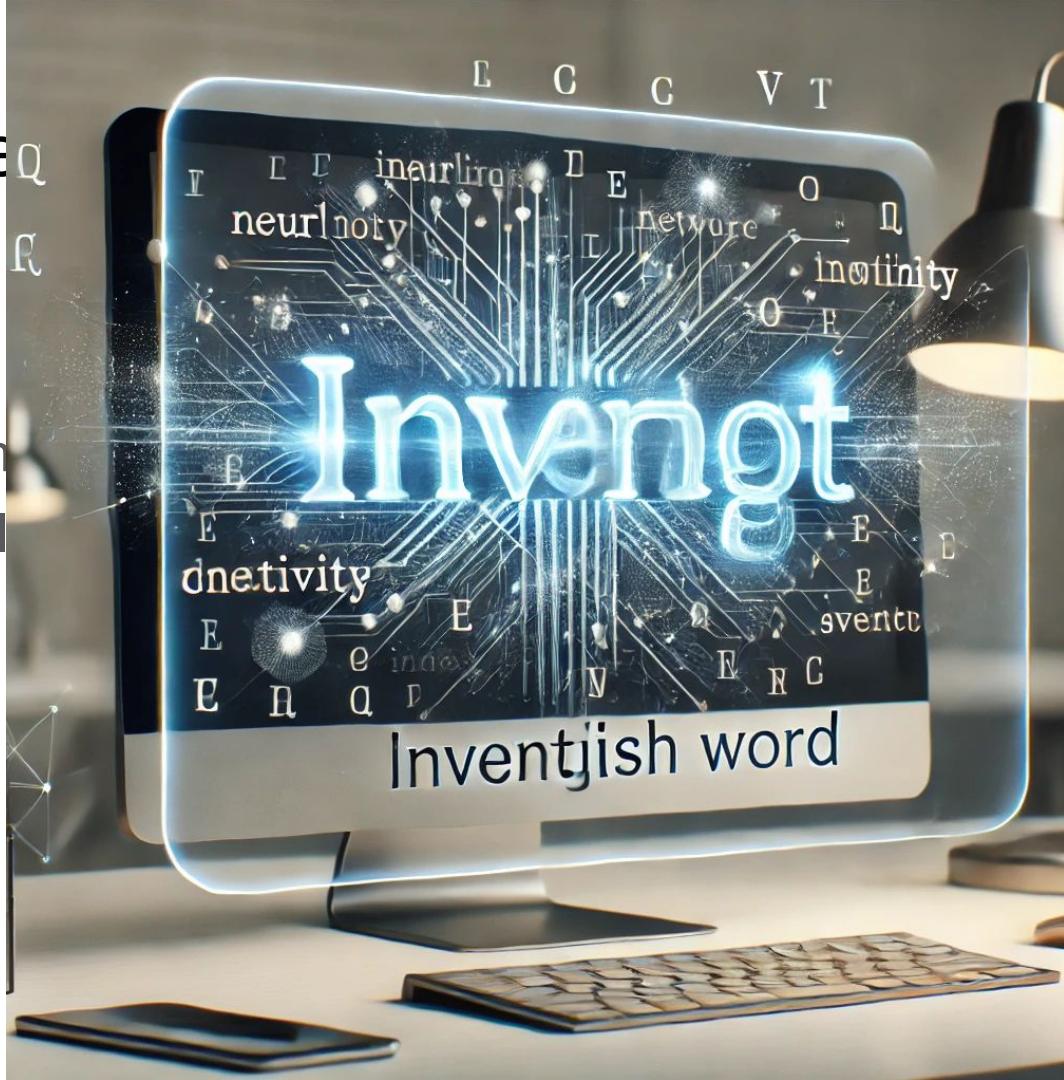
Vexilora (n)

- A mythical spirit with the ability to move
- Example: guides travelers safely through tre



Which one

- A mineral
- A dance
- A program
- A Mythical



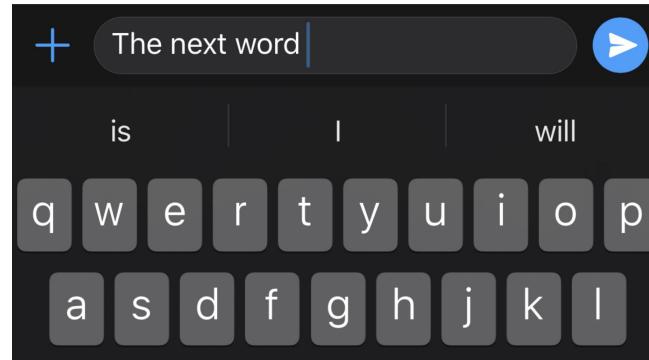
Word2Vec et al.



SAPIENZA
UNIVERSITÀ DI ROMA

A Digression: Language Models

- Let \mathbf{D} be a vocabulary of Words, and let $w_i \in \mathbf{D}$ be a word,
- A Language Model (LM) is a probability distribution over sequences $\langle w_1, w_2, \dots, w_n \rangle$ of words, i.e., $P(w_1, w_2, \dots, w_n)$
 - In other words given a language whose vocabulary is \mathbf{D} , we seek to compute the probability that a sentence occurs in that language.



Types of Language Models

- Unigram LMs: $P(w_1, w_2, \dots, w_n) = P(w_1)P(w_2) \dots P(w_n)$
- N-gram LMs: $\prod P(w_i | w_{i-(n-1)}, \dots, w_{i-1})$
- Neural LMs: $\prod P(w_i | \text{context}(w_i))$



Word Embeddings: Quiz Time!

- What is “tezguino”

- A bottle of _____ is on the table
- Everybody like _____
- Don't have _____ before you drive
- We make _____ out of corn

- loud
- motor oil
- tortillas
- choices
- wine



Distributional Statistics Matrix

	A bottle of _____ is on the table	Everybody like _____	Don't have _____ before you drive	We make _____ out of corn
tezgūino				
loud				
motor oil				
tortillas				
choices				
wine				



Enters Word2Vec



$$P(\text{Is} | \text{The, Cat, On, The})$$



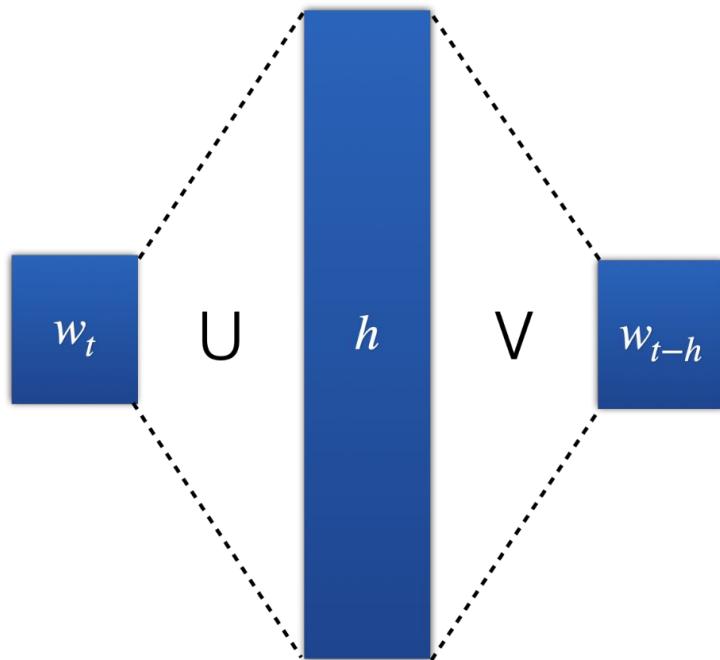
$$\begin{aligned} P(\text{The} | \text{Is}) \\ P(\text{Cat} | \text{Is}) \\ P(\text{On} | \text{Is}) \end{aligned}$$

CBOW | Skip-Gram



SAPIENZA
UNIVERSITÀ DI ROMA

Skip-Gram Word2Vec

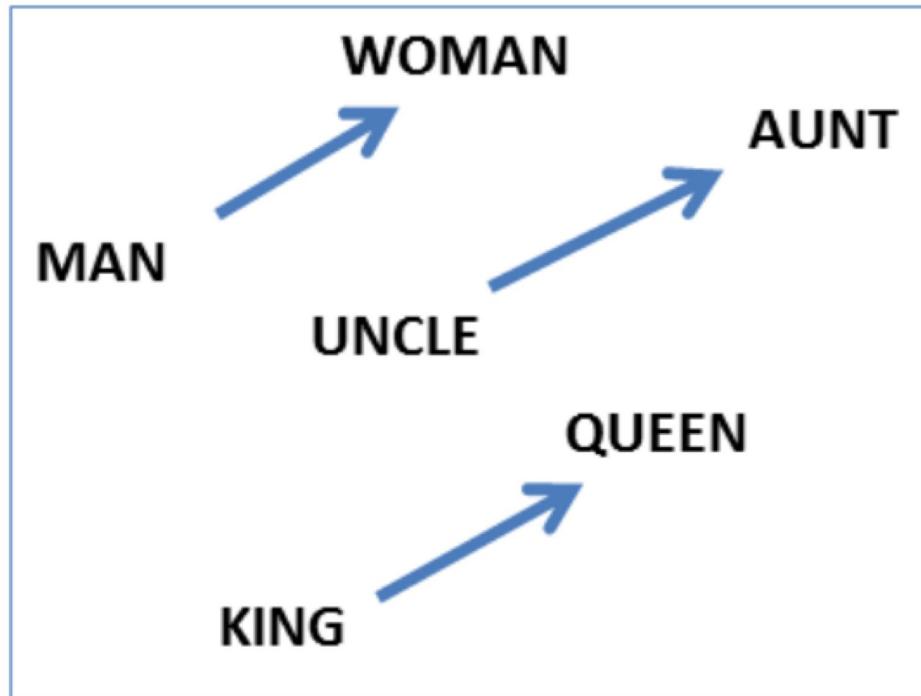


$$P(w_t | w_{t-h}, \dots, w_{t-1}, w_{t+1}, w_{t+h}) = \\ = \text{softmax}(w_t U V^T)$$

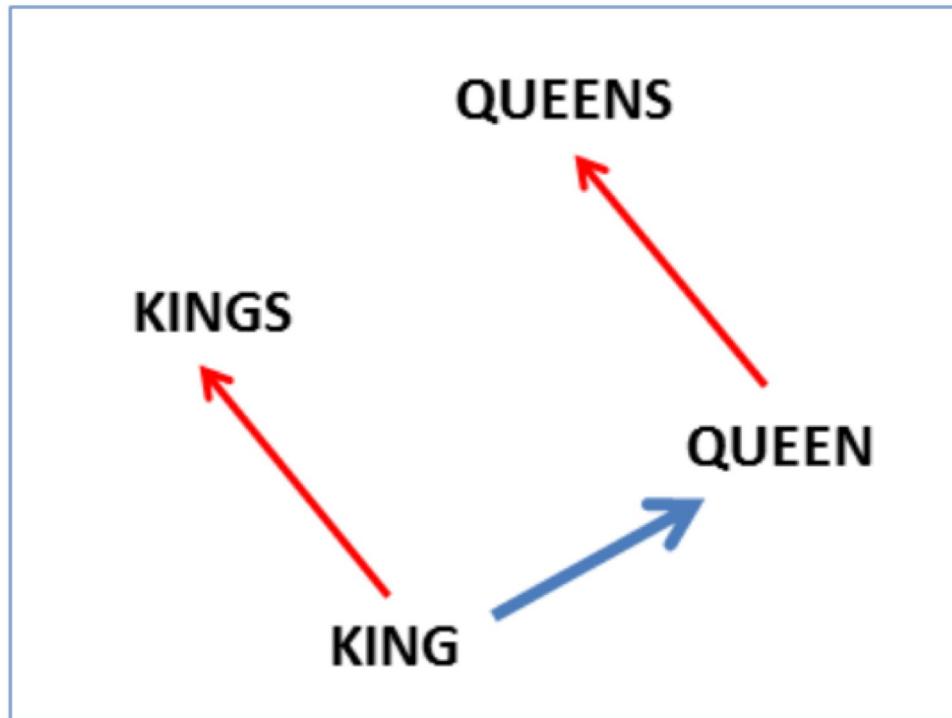
$$U, V \in \mathbb{R}^{n \times d}$$
$$n = |D|$$



Some Cool Properties of Word2Vec



Some Cool Properties of Word2Vec



Some Cool Properties of Word2Vec

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza



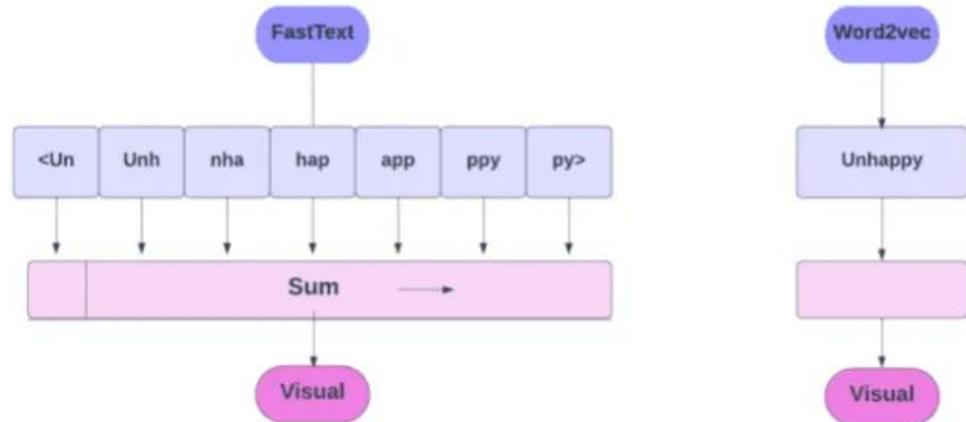
A Big Drawback of Word2Vec

- Word2Vec (or word embeddings in general) associate one vector to each symbol.
- One vector!
- How about:
 - Feet:
 - My feet hurt
 - This tree is 6 feet tall
 - Rock:
 - Rock paper and scissor
 - Rock 'n' roll
 - Break (in English) has more than 70 different meanings!!!



FastText

- **Efficient Word Representation**
 - It is an extension of Word2Vec that represents words as bags of character n-grams, allowing it to capture subword information.
 - This helps in better handling of rare or out-of-vocabulary words.
- **Fast Training**
 - FastText is designed to be faster and more efficient than traditional word embeddings.
- **Handling Morphological Rich Languages**
 - By breaking down words into subword components (n-grams), FastText is particularly effective for languages with rich morphology or languages with a high number of compound words.



FastText's Loss

- For a chosen context c of a word at position t , we have

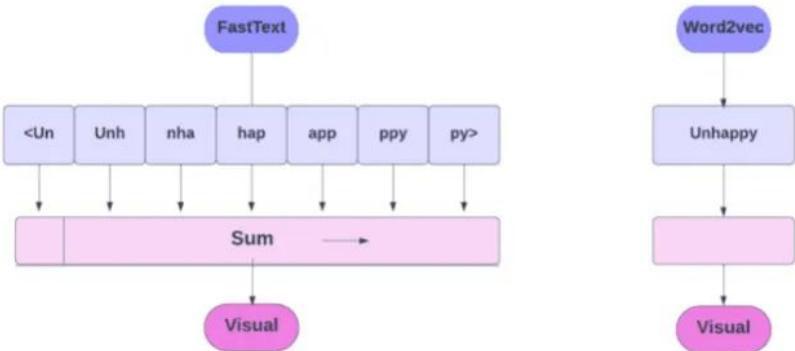
$$\log \left(1 + e^{-s(w_t, w_c)} \right) + \sum_{n \in \mathcal{N}_{t,c}} \log \left(1 + e^{s(w_t, n)} \right)$$

where $\mathcal{N}_{t,c}$ is a set of negative examples sampled from the vocabulary.

- By denoting the logistic loss $\ell : x \mapsto \log(1 + e^{-x})$
- We can rewrite it as:

$$\sum_{t=1}^T \left[\sum_{c \in \mathcal{C}_t} \ell(s(w_t, w_c)) + \sum_{n \in \mathcal{N}_{t,c}} \ell(-s(w_t, n)) \right]$$

- where $s(w, c) = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g^\top \mathbf{v}_c$.



Suppose that you are given a dictionary of n -grams of size G . Given a word w , let us denote by $\mathcal{G}_w \subset \{1, \dots, G\}$ the set of n -grams appearing in w . We associate a vector representation \mathbf{z}_g to each n -gram g .



FastText Performance

	POS	Chunking	NER	SA(IMDb)			SA(SST)			NMT Perplexity
				Bi-LSTM	CNN	FastText	Bi-LSTM	CNN	FastText	
SGNS	94.54	88.21	87.12	85.36	88.78	68.52	64.08	66.93	55.53	79.14
CBOW	93.79	84.91	83.83	86.93	85.88	69.55	65.63	65.06	60.73	102.33
GloVe	93.32	84.11	85.3	70.41	87.56	67.94	65.16	65.15	58.29	84.20
FastText	94.36	87.96	87.10	73.97	83.69	67.00	50.01	63.25	52.77	82.60



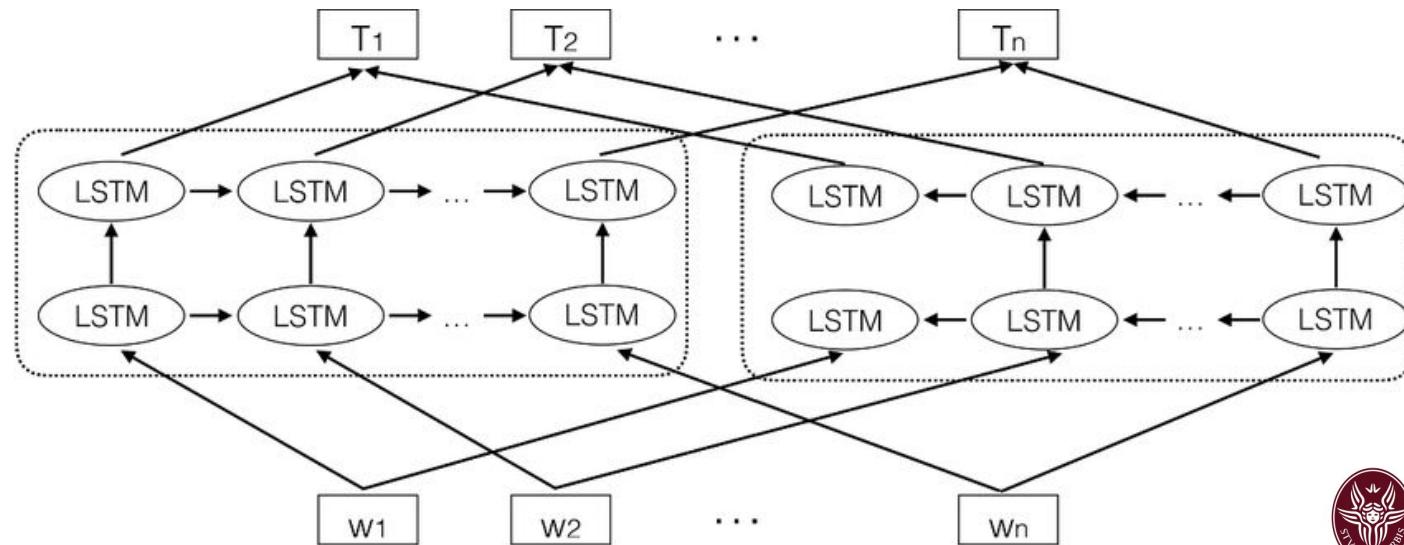
ELMo



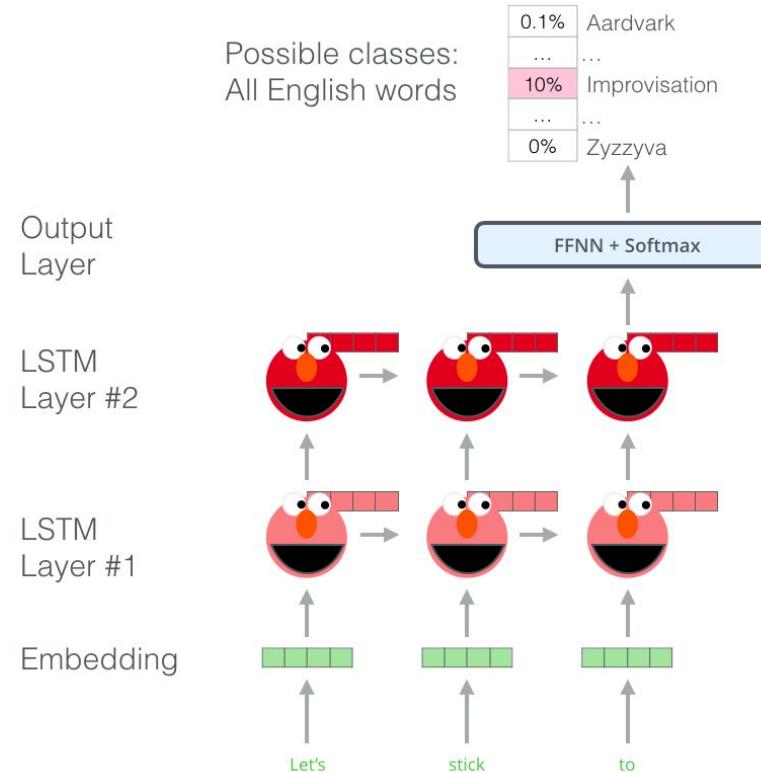
SAPIENZA
UNIVERSITÀ DI ROMA

Enters EIMo

- It trains contextual word embeddings whose value is not static but is computed every time on the basis of the context (i.e., surrounding words)

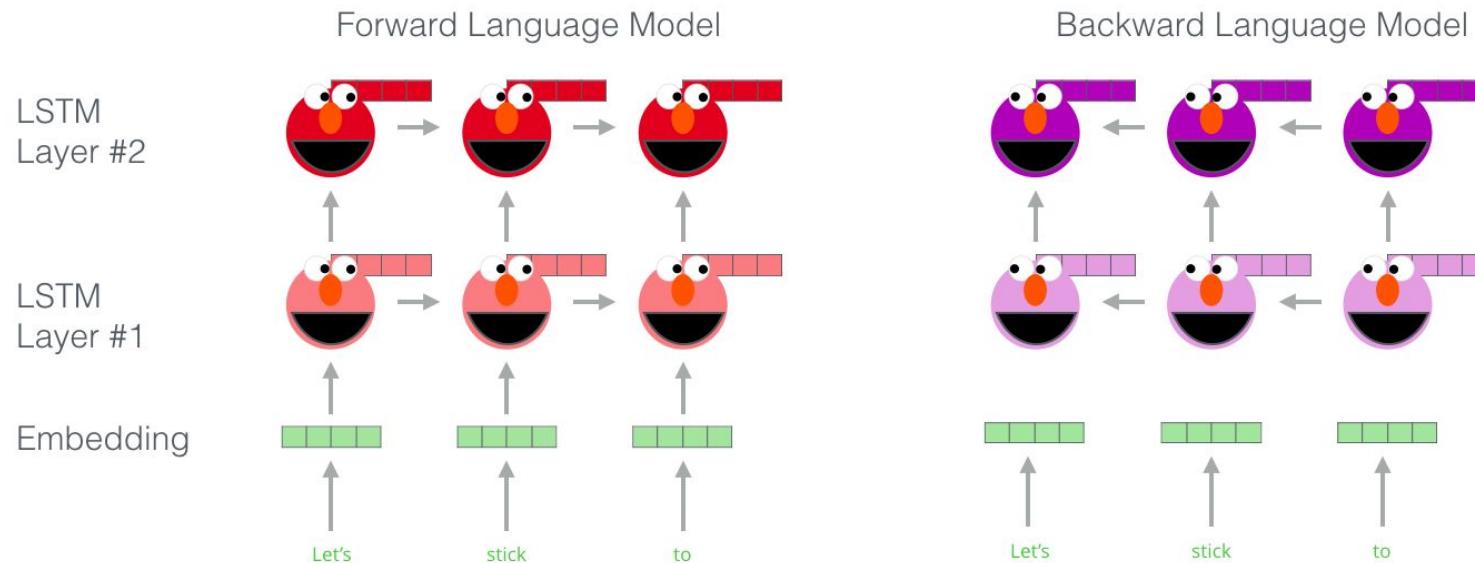


An Example



An Example

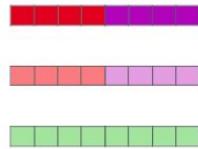
Embedding of “stick” in “Let’s stick to” - Step #1



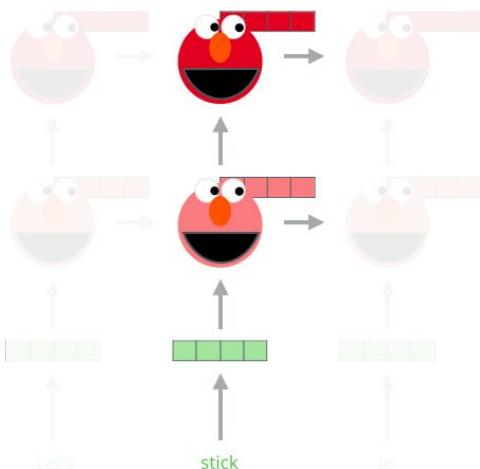
An Example

Embedding of “stick” in “Let’s stick to” - Step #2

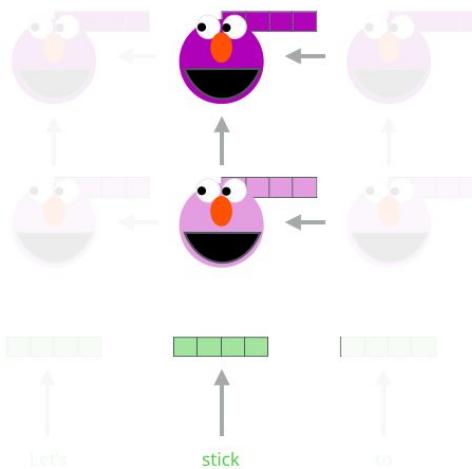
1- Concatenate hidden layers



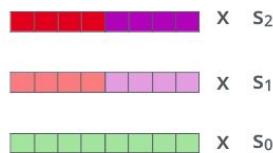
Forward Language Model



Backward Language Model



2- Multiply each vector by a weight based on the task



3- Sum the (now weighted) vectors



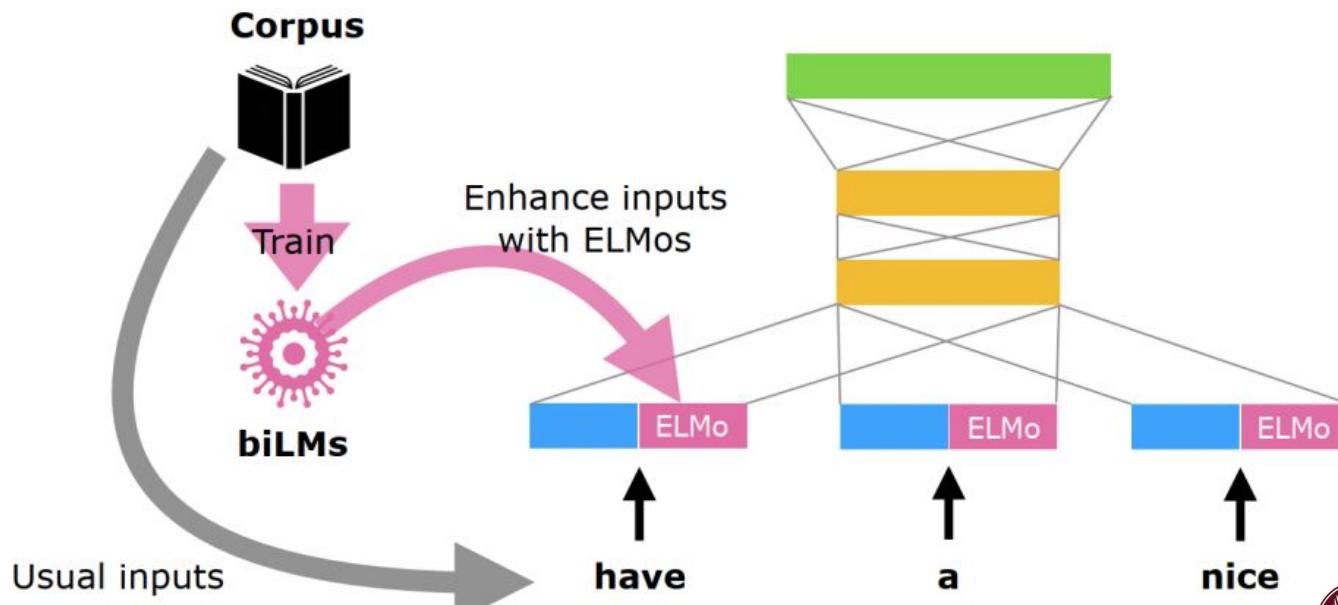
ELMo embedding of “stick” for this task in this context



SAPIENZA
UNIVERSITÀ DI ROMA

ELMo as Word Embedding

ELMo can be integrated to almost all neural NLP tasks with simple concatenation to the embedding layer



Experiments

TASK	PREVIOUS SOTA		OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F₁ for SQuAD, SRL and NER; average F₁ for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The “increase” column lists both the absolute and relative improvements over our baseline.



No... Not that kind of Transformers



SAPIENZA
UNIVERSITÀ DI ROMA

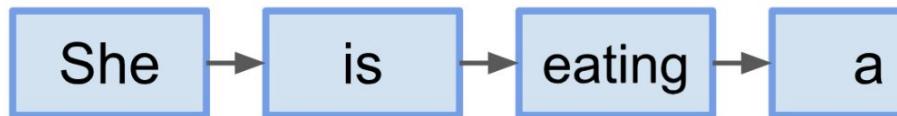
Attention Mechanism



SAPIENZA
UNIVERSITÀ DI ROMA

Seq2Seq Model

Encoder



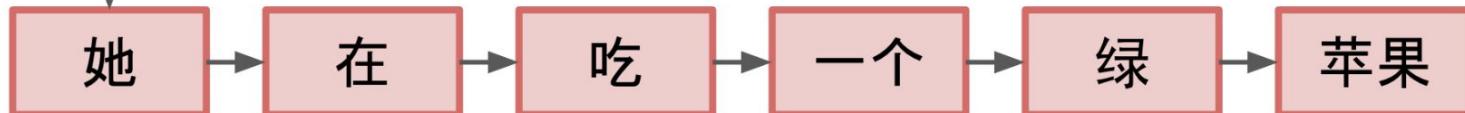
A critical and apparent disadvantage of this fixed-length context vector design is incapability of remembering long sentences.

Often it has forgotten the first part once it completes processing the whole input.

Context vector (length: 5)

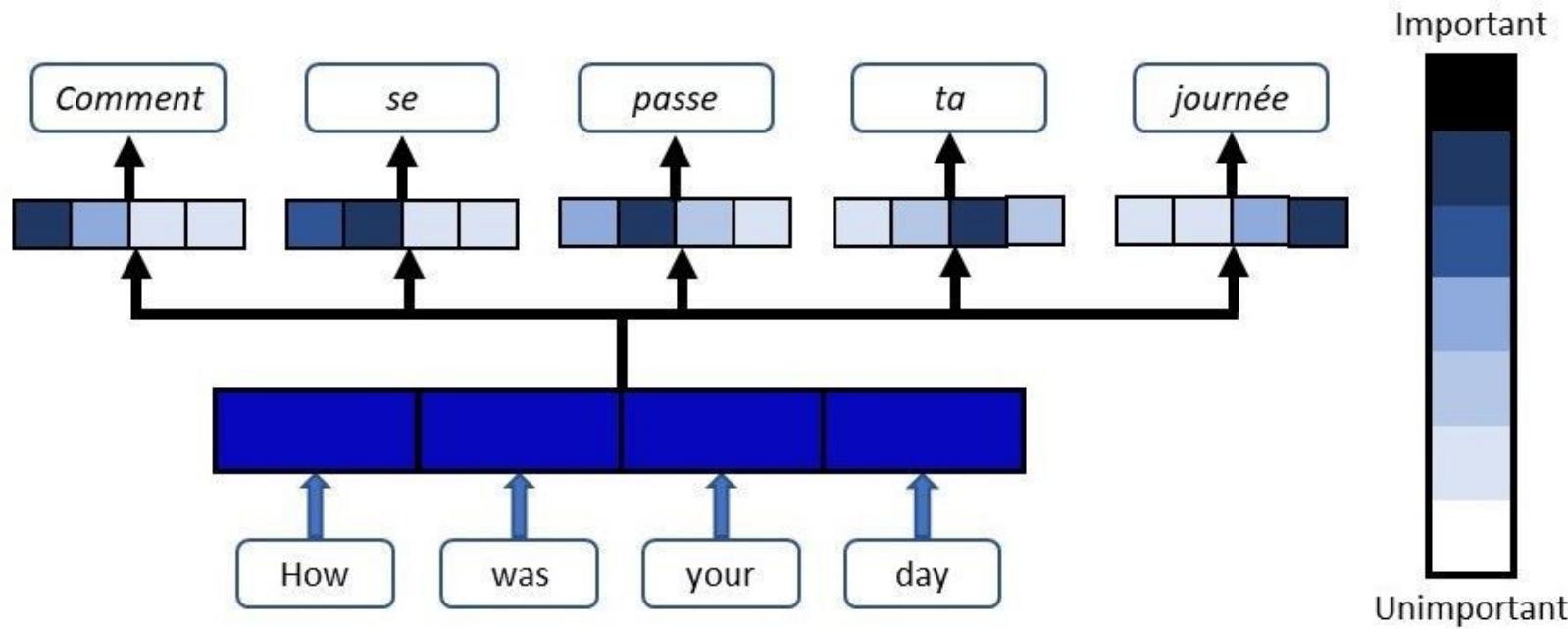
[0.1, -0.2, 0.8, 1.5, -0.3]

Decoder

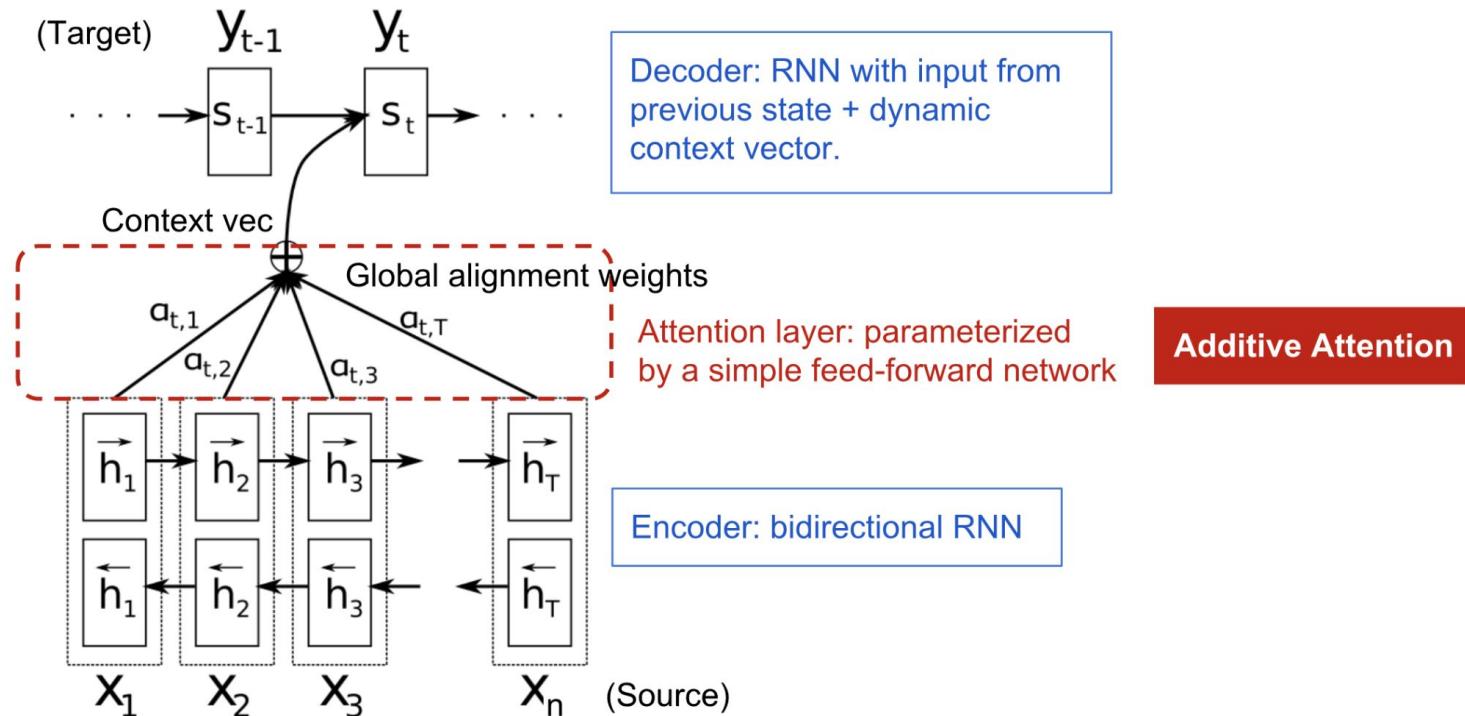


SAPIENZA
UNIVERSITÀ DI ROMA

Let's focus on a single output



The Attention Mechanism



The Attention Mechanism

- Say, we have a source sequence x of length n and try to output a target sequence y of length m :

$$\mathbf{x} = [x_1, x_2, \dots, x_n]$$

$$\mathbf{y} = [y_1, y_2, \dots, y_m]$$

- The encoder is a bidirectional RNN with a forward hidden state $\vec{\mathbf{h}}_i$ and a backward one $\overleftarrow{\mathbf{h}}_i$.
 - A simple concatenation of two represents the encoder state.
 - The motivation is to include both the preceding and following words in the annotation of one word.

$$\mathbf{h}_i = [\vec{\mathbf{h}}_i^T; \overleftarrow{\mathbf{h}}_i^T]^T, i = 1, \dots, n$$



The Attention Mechanism

- The decoder network has hidden state $\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t)$ for the output word at position t , $t=1, \dots, m$, where the context vector \mathbf{c}_t is a sum of hidden states of the input sequence, weighted by alignment scores:

$$\mathbf{c}_t = \sum_{i=1}^n \alpha_{t,i} \mathbf{h}_i \quad ; \text{ Context vector for output } y_t$$

$$\begin{aligned} \alpha_{t,i} &= \text{align}(y_t, x_i) && ; \text{ How well two words } y_t \text{ and } x_i \text{ are aligned.} \\ &= \frac{\exp(\text{score}(\mathbf{s}_{t-1}, \mathbf{h}_i))}{\sum_{i'=1}^n \exp(\text{score}(\mathbf{s}_{t-1}, \mathbf{h}_{i'}))} && ; \text{ Softmax of some predefined alignment score..} \end{aligned}$$

The alignment model assigns a score $\alpha_{t,i}$ to the pair of input at position i and output at position t , (y_t, x_i) , based on how well they match. The set of $\{\alpha_{t,i}\}$ are weights defining how much of each source hidden state should be considered for each output.



The Attention Mechanism

- The alignment score α is parametrized by a feed-forward network with a single hidden layer and this network is jointly trained with other parts of the model.
- The score function is therefore in the following form, given that \tanh is used as the non-linear activation function:

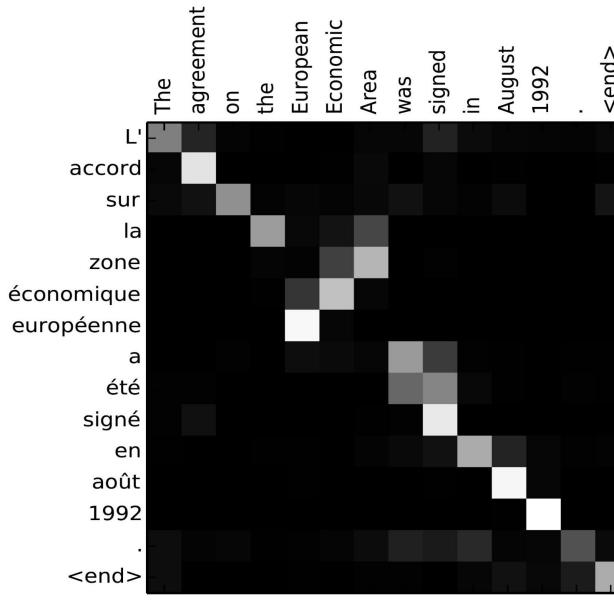
$$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\mathbf{s}_t; \mathbf{h}_i])$$

where both \mathbf{v}_a and \mathbf{W}_a are weight matrices to be learned in the alignment model.



The Attention Mechanism

- The matrix of alignment scores is a nice byproduct to explicitly show the correlation between source and target words.



A Family of Attention Scores

Name	Alignment score function	Citation
Content-base attention	$\text{score}(s_t, \mathbf{h}_i) = \text{cosine}[s_t, \mathbf{h}_i]$	Graves2014
Additive(*)	$\text{score}(s_t, \mathbf{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[s_t; \mathbf{h}_i])$	Bahdanau2015
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a s_t)$ Note: This simplifies the softmax alignment to only depend on the target position.	Luong2015
General	$\text{score}(s_t, \mathbf{h}_i) = s_t^\top \mathbf{W}_a \mathbf{h}_i$ where \mathbf{W}_a is a trainable weight matrix in the attention layer.	Luong2015
Dot-Product	$\text{score}(s_t, \mathbf{h}_i) = s_t^\top \mathbf{h}_i$	Luong2015
Scaled Dot-Product(^)	$\text{score}(s_t, \mathbf{h}_i) = \frac{s_t^\top \mathbf{h}_i}{\sqrt{n}}$ Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.	Vaswani2017



Self-Attention



SAPIENZA
UNIVERSITÀ DI ROMA

Self-Attention

[Self-attention](#), also known as [intra-attention](#), is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence.

It has been shown to be very useful in machine reading, abstractive summarization, or image description generation.

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

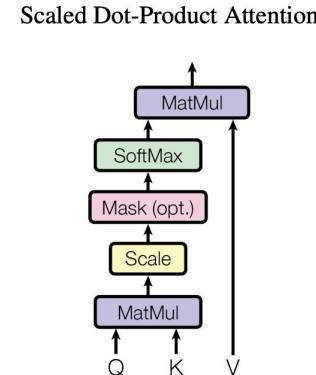
The current word is in red and the size of the blue shade indicates the activation level. (Image source: [Cheng et al., 2016](#))



Key, Value and Query

- The transformer views the encoded representation of the input as a set of key-value pairs, (K, V) , both of dimension n (the input sequence length)
 - For example, in the context of NMT, both the keys and values are the encoder hidden states.
- In the decoder, the previous output is compressed into a query Q (of dimension m) and the next output is produced by mapping this query and the set of keys and values.
- The transformer adopts the scaled dot-product attention: the output is a weighted sum of the values, where the weight assigned to each value is determined by the dot-product of the query with all the keys:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{n}}\right)V$$



Transformer: a seq2seq model

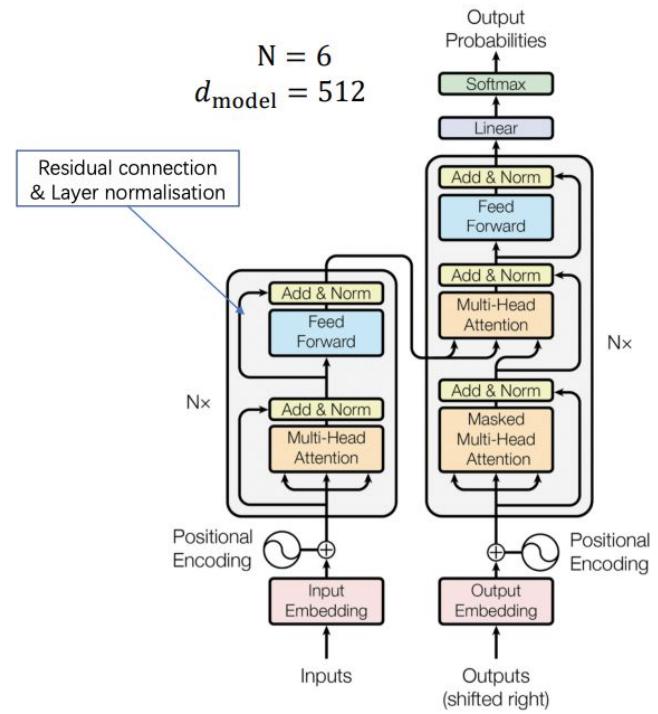
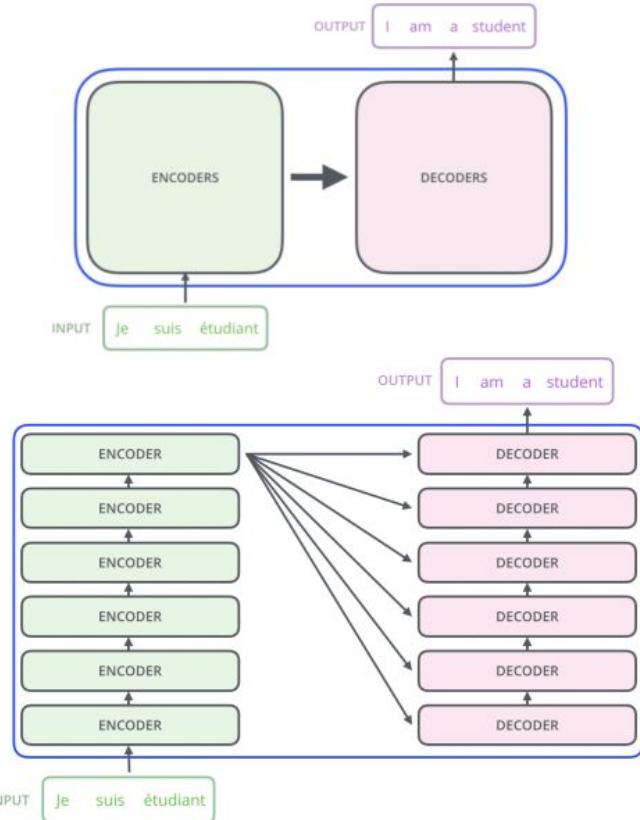


Figure 1: The Transformer - model architecture.

Figure in (Vaswani et al., 2017)

["Attention is All you Need"](#) (Vaswani, et al., 2017), without a doubt, is one of the most impactful and interesting paper published in 2017.

It makes it possible to do [seq2seq modeling](#) without recurrent network units.



Key, Value and Query: An Example

- Consider this phrase – “[Action gets results](#)”.
- To calculate the self-attention for the first word “Action”, we will calculate scores for all the words in the phrase with respect to “Action”.
- This score determines the importance of other words when we are encoding a certain word in an input sequence.



Key, Value and Query: An Example

- The score for the first word is calculated by taking the dot product of the Query vector (q_1) with the keys vectors (k_1, k_2, k_3) of all the words.

Word	q vector	k vector	v vector	score
Action	q_1	k_1	v_1	$q_1 \cdot k_1$
gets		k_2	v_2	$q_1 \cdot k_2$
results		k_3	v_3	$q_1 \cdot k_3$



Key, Value and Query: An Example

- These scores are divided by 8 which is the square root of the dimension of the key vector

Word	q vector	k vector	v vector	score	score / 8
Action	q_1	k_1	v_1	$q_1 \cdot k_1$	$q_1 \cdot k_1 / 8$
gets		k_2	v_2	$q_1 \cdot k_2$	$q_1 \cdot k_2 / 8$
results		k_3	v_3	$q_1 \cdot k_3$	$q_1 \cdot k_3 / 8$



Key, Value and Query: An Example

- Next, these scores are normalized using the softmax activation function.

Word	q vector	k vector	v vector	score	score / 8	Softmax
Action	q_1	k_1	v_1	$q_1 \cdot k_1$	$q_1 \cdot k_1 / 8$	x_{11}
gets		k_2	v_2	$q_1 \cdot k_2$	$q_1 \cdot k_2 / 8$	x_{12}
results		k_3	v_3	$q_1 \cdot k_3$	$q_1 \cdot k_3 / 8$	x_{13}



Key, Value and Query: An Example

- These normalized scores are then multiplied by the value vectors (v_1, v_2, v_3) and sum up the resultant vectors to arrive at the final vector (z_1).
- This is the output of the self-attention layer. It is then passed on to the feed-forward network as input.

Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum
Action	q_1	k_1	v_1	$q_1 \cdot k_1$	$q_1 \cdot k_1 / 8$	x_{11}	$x_{11} * v_1$	z_1
gets		k_2	v_2	$q_1 \cdot k_2$	$q_1 \cdot k_2 / 8$	x_{12}	$x_{12} * v_2$	
results		k_3	v_3	$q_1 \cdot k_3$	$q_1 \cdot k_3 / 8$	x_{13}	$x_{13} * v_3$	



Key, Value and Query: An Example

- So, z_1 is the self-attention vector for the first word of the input sequence “Action gets results”.
- We can get the vectors for the rest of the words in the input sequence in the same fashion.

Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum [#]
Action		k_1	v_1	$q_2 \cdot k_1$	$q_2 \cdot k_1 / 8$	x_{21}	$x_{21} * v_1$	
gets	q_2	k_2	v_2	$q_2 \cdot k_2$	$q_2 \cdot k_2 / 8$	x_{22}	$x_{22} * v_2$	z_2
results		k_3	v_3	$q_2 \cdot k_3$	$q_2 \cdot k_3 / 8$	x_{23}	$x_{23} * v_3$	

Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum [#]
Action		k_1	v_1	$q_3 \cdot k_1$	$q_3 \cdot k_1 / 8$	x_{31}	$x_{31} * v_1$	
gets		k_2	v_2	$q_3 \cdot k_2$	$q_3 \cdot k_2 / 8$	x_{32}	$x_{32} * v_2$	
results	q_3	k_3	v_3	$q_3 \cdot k_3$	$q_3 \cdot k_3 / 8$	x_{33}	$x_{33} * v_3$	z_3

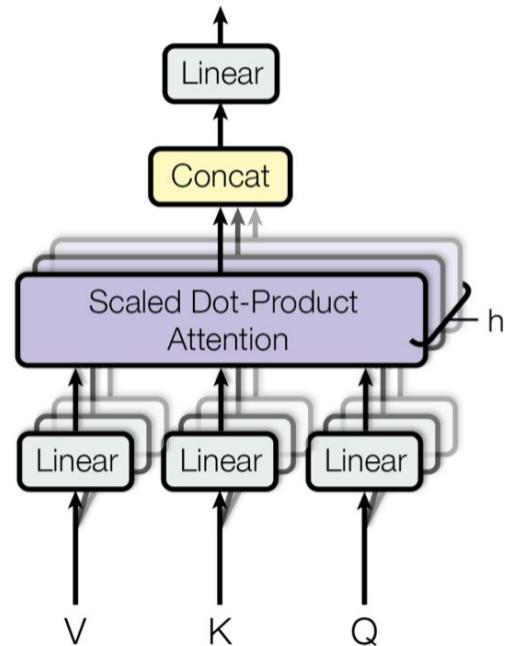


Multi-head Self-Attention

- Rather than only computing the attention once, **the multi-head mechanism runs through the scaled dot-product attention multiple times in parallel.**
- “multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this.”

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = [\text{head}_1; \dots; \text{head}_h] \mathbf{W}^O$$
$$\text{where } \text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$$

where \mathbf{W}_i^Q , \mathbf{W}_i^K , \mathbf{W}_i^V , and \mathbf{W}^O are parameter matrices to be learned.



The Transformer Architecture

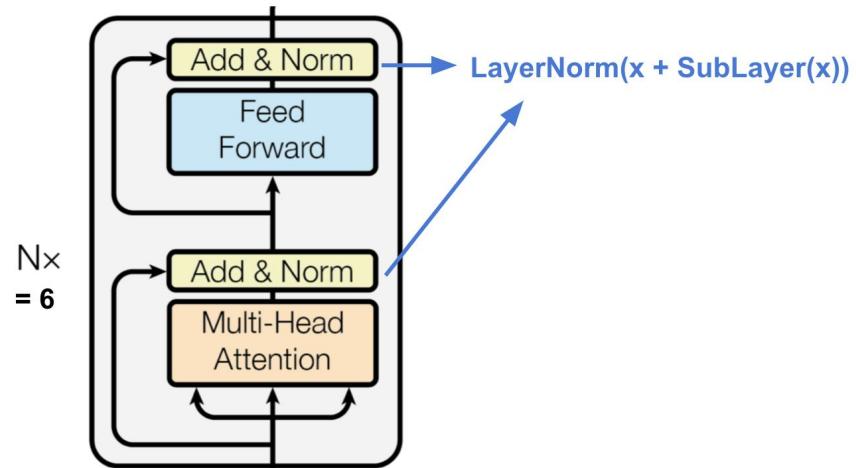


SAPIENZA
UNIVERSITÀ DI ROMA

Encoder

The encoder generates an attention-based representation with capability to locate a specific piece of information from a potentially infinitely-large context.

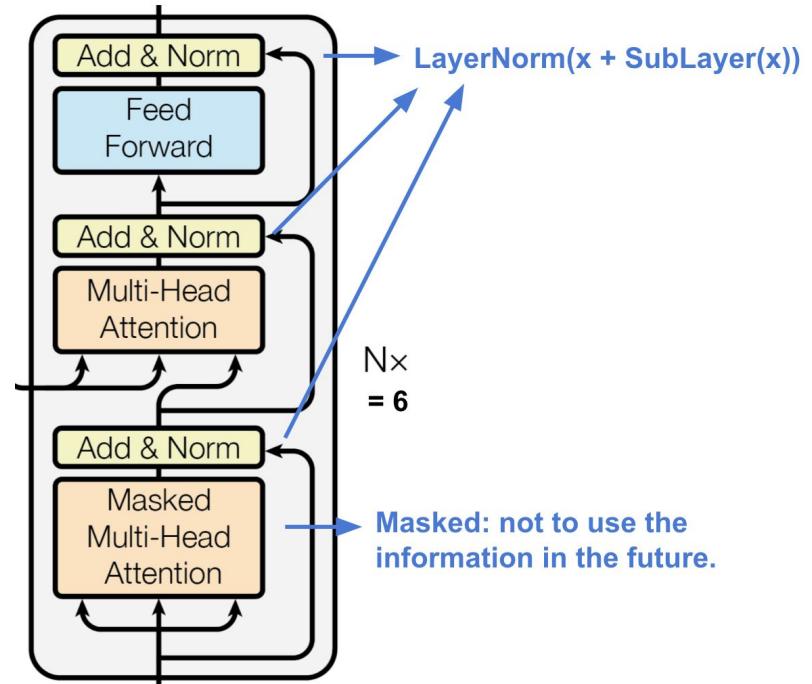
- A stack of N=6 identical layers.
- Each layer has a multi-head self-attention layer and a simple position-wise fully connected feed-forward network.
- Each sub-layer adopts a residual connection and a layer normalization.
- All the sub-layers output data of the same dimension $d_{\text{model}} = 512$.



Decoder

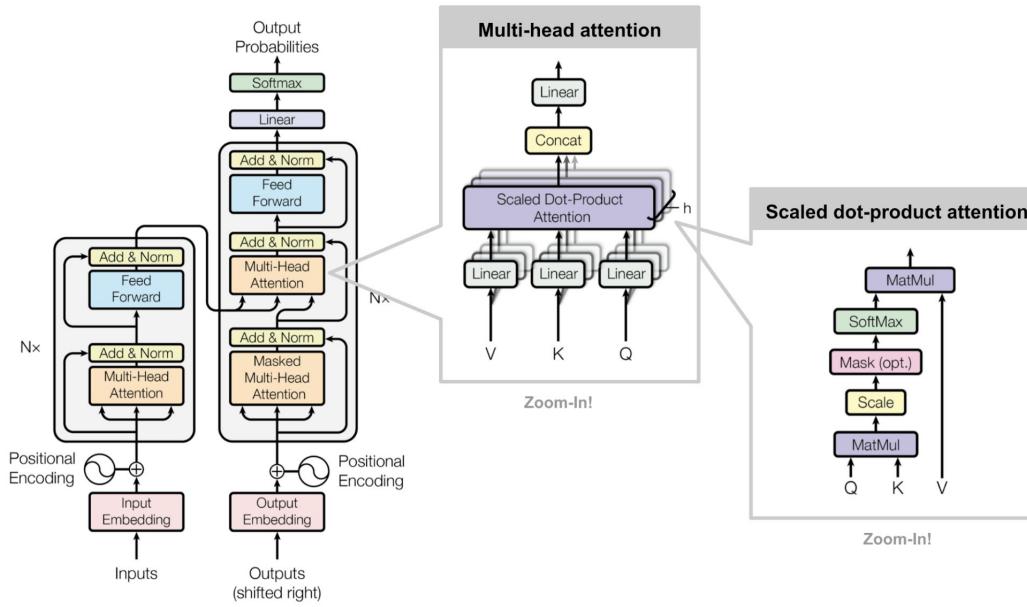
The decoder is able to retrieval from the encoded representation.

- A stack of $N = 6$ identical layers
- Each layer has two sub-layers of multi-head attention mechanisms and one sub-layer of fully-connected feed-forward network.
- Similar to the encoder, each sub-layer adopts a residual connection and a layer normalization.
- The first multi-head attention sub-layer is **modified** to prevent positions from attending to subsequent positions, as we don't want to look into the future of the target sequence when predicting the current position.



The Full Architecture

- Both the source and target sequences first go through embedding layers to produce data of the same dimension $d_{model} = 512$.
- To preserve the position information, a sinusoid-wave-based positional encoding is applied and summed with the embedding output.
- A softmax and linear layer are added to the final decoder output.
- The output from the last encoder in the encoder-stack **is passed to all the decoders in the decoder-stack**



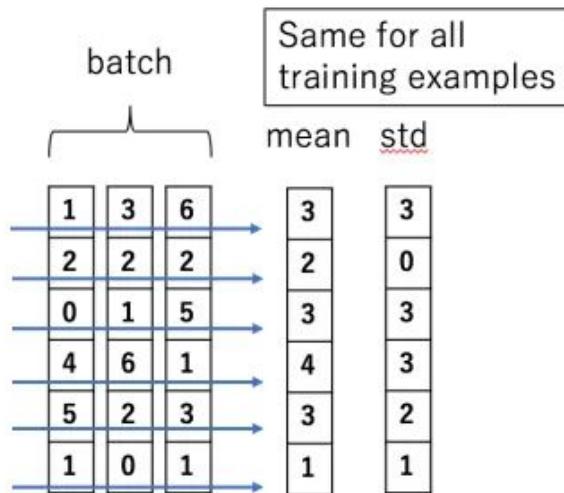
Normalization



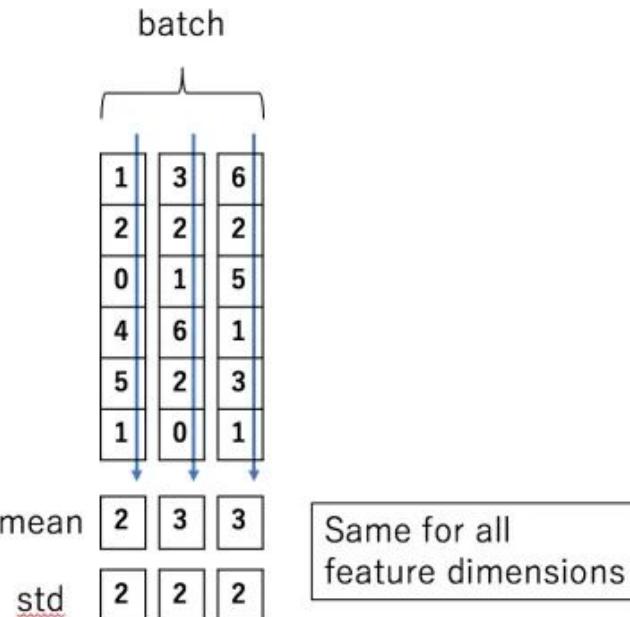
SAPIENZA
UNIVERSITÀ DI ROMA

Layer Normalization

Batch Normalization



Layer Normalization



Correct Normalization Position

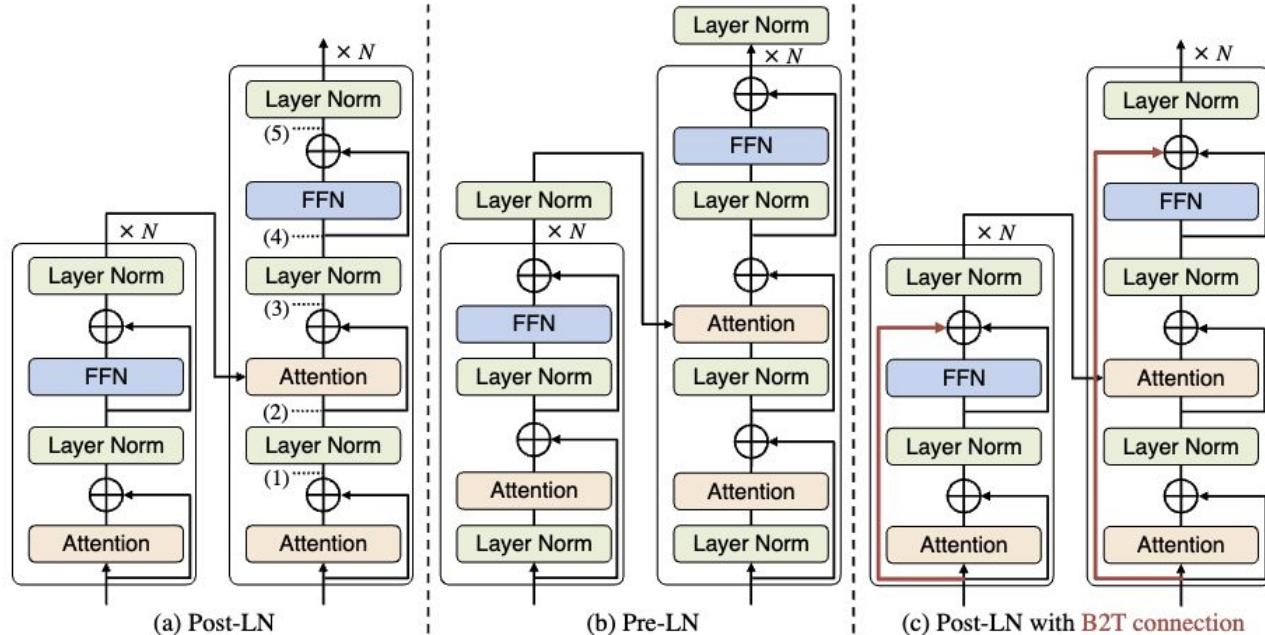


Figure 2: Transformer-based encoder-decoder architectures for (a) Post-LN, (b) Pre-LN, and (c) Post-LN with our introduced connection.

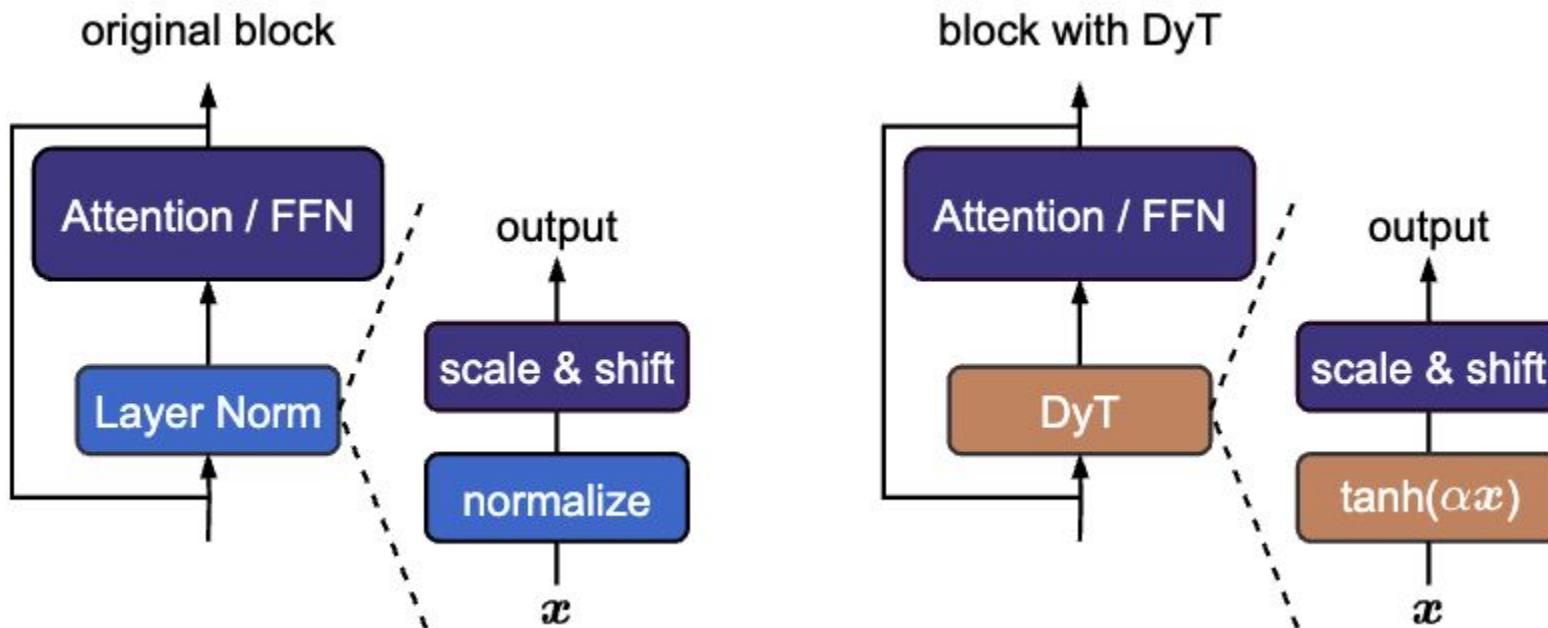
From: <https://arxiv.org/pdf/2206.00330v1>

Correct Normalization Position. Is it Worth?

Table 2: F-1 based ROUGE-1, 2, and L (R-1, R-2, and R-L respectively) scores on headline generation [21].

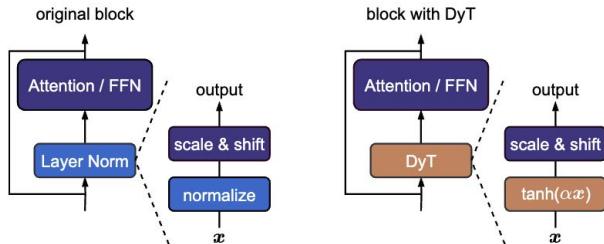
Method	R-1	R-2	R-L
Enc-Dec: 6L-6L			
Post-LN	38.57	19.37	35.79
Pre-LN	38.27	19.29	35.39
B2T connection	38.43	19.37	35.72
Enc-Dec: 18L-18L			
Post-LN	Training failed		
Pre-LN	38.97	19.94	35.99
B2T connection	39.61	20.28	36.66

(Do not use) Layer Normalization



From “Transformers without Normalization”: <https://arxiv.org/pdf/2503.10622>

Why? Because it's more efficient!

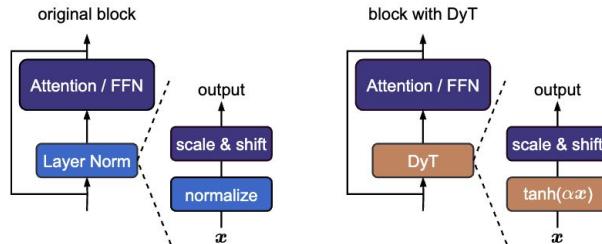


LLaMA 7B	inference		training	
	layer	model	layer	model
RMSNorm	2.1s	14.1s	8.3s	42.6s
DyT	1.0s	13.0s	4.8s	39.1s
reduction	↓ 52.4%	↓ 7.8%	↓ 42.2%	↓ 8.2%

From "Transformers without Normalization": <https://arxiv.org/pdf/2503.10622>

How?

$$\text{DyT}(x) = \gamma * \tanh(\alpha x) + \beta$$



Algorithm 1 Pseudocode of DyT layer.

```
# input x has the shape of [B, T, C]
# B: batch size, T: tokens, C: dimension

class DyT(Module):
    def __init__(self, C, init_alpha):
        super().__init__()
        self.alpha = Parameter(ones(1) * init_alpha)
        self.gamma = Parameter(ones(C))
        self.beta = Parameter(zeros(C))

    def forward(self, x):
        x = tanh(self.alpha * x)
        return self.gamma * x + self.beta
```

From “Transformers without Normalization”: <https://arxiv.org/pdf/2503.10622>

Why Transformers



SAPIENZA
UNIVERSITÀ DI ROMA

Why Transformers are Preferable to RNNs?

- The total computational complexity per layer
- The amount of computation that can be parallelized
 - minimum number of sequential operations required
- The path length between long-range dependencies in the network.
 - Learning long-range dependencies is a key challenge in many sequence transduction tasks.

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(\log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$



Experiments on NMT

The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	41.29	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$3.3 \cdot 10^{18}$	
Transformer (big)	28.4	41.0	$2.3 \cdot 10^{19}$	



The Annotated Transformer



SAPIENZA
UNIVERSITÀ DI ROMA

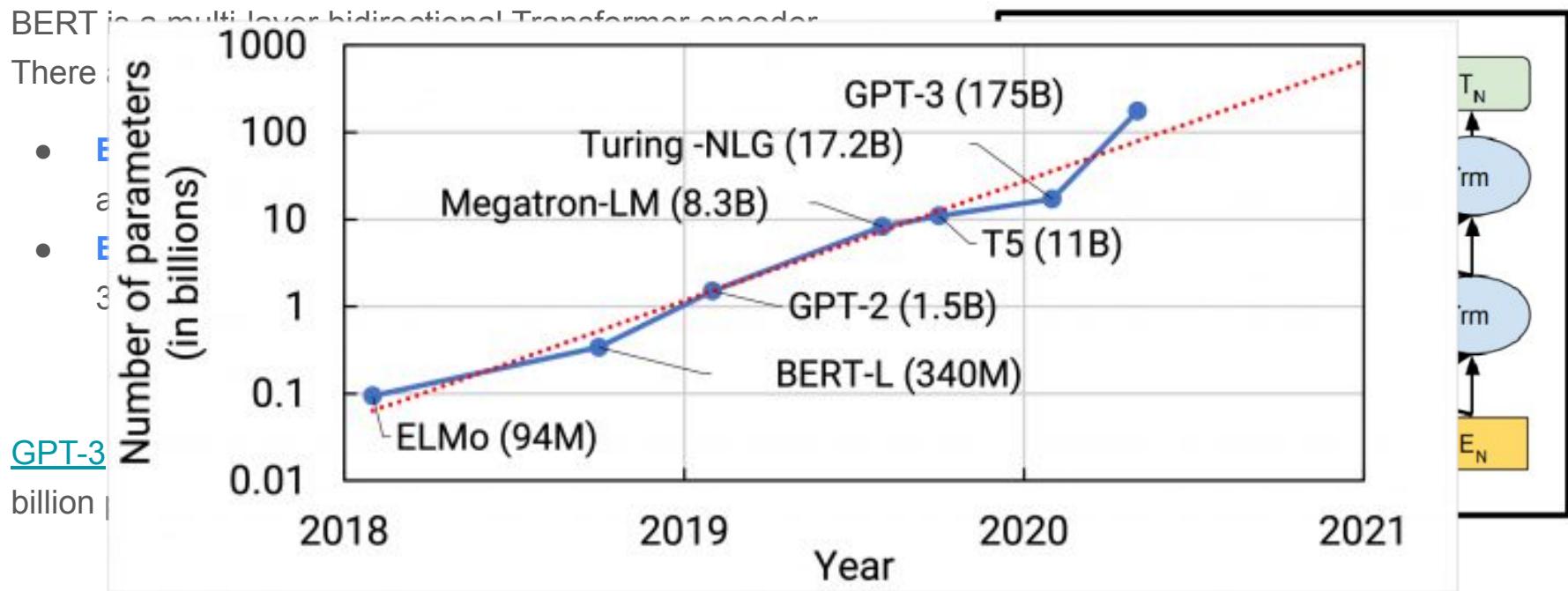
BERT

Bidirectional Encoder Representations from Transformers



SAPIENZA
UNIVERSITÀ DI ROMA

The Architecture



Pretraining BERT: Masked LMs and Next Sentence Prediction

- In **masked language modeling** instead of predicting every next token, a percentage of input tokens is masked at random and only those masked tokens are predicted.
 - Bi-directional models are more powerful than uni-directional language models.
 - But in a multi-layered model bi-directional models do not work because the lower layers leak information and allow a token to see itself in later layers.
- Similarly, **next sentence prediction** takes a pair of sentences and try to predict if one follows the other in the training dataset.



Masked Language Model

- 80% of the time: Replace the word with the [MASK] token, e.g., my dog is hairy → my dog is [MASK]
- 10% of the time: Replace the word with a random word, e.g., my dog is hairy → my dog is apple
- 10% of the time: Keep the word unchanged, e.g., my dog is hairy → my dog is hairy. The purpose of this is to bias the representation towards the actual observed word.



Next Sequence Prediction

Input = [CLS] the man went to [MASK] store [SEP]

he bought a gallon [MASK] milk [SEP]

Label = IsNext

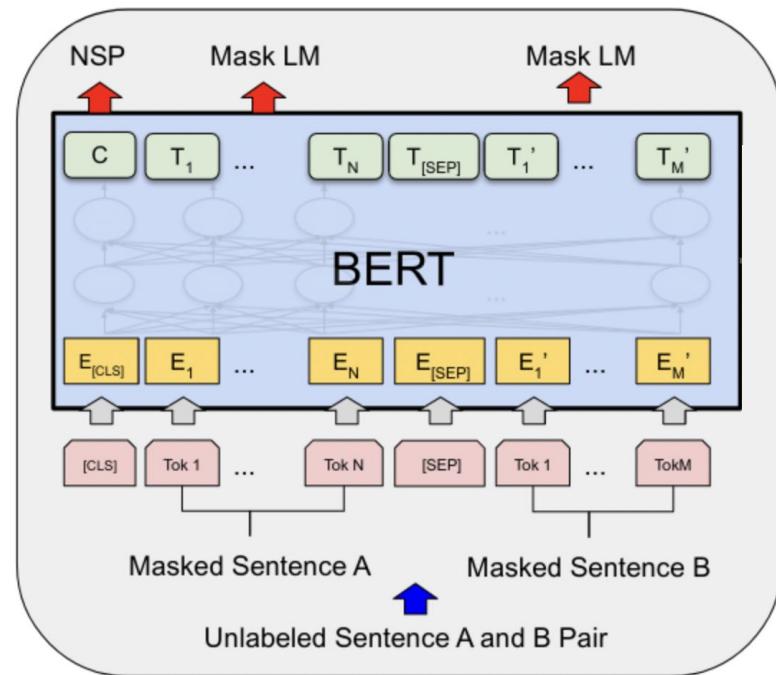
Input = [CLS] the man [MASK] to the store [SEP]

penguin [MASK] are flight ##less birds [SEP]

Label = NotNext



Pretraining BERT: Masked LMs and Next Sentence Prediction



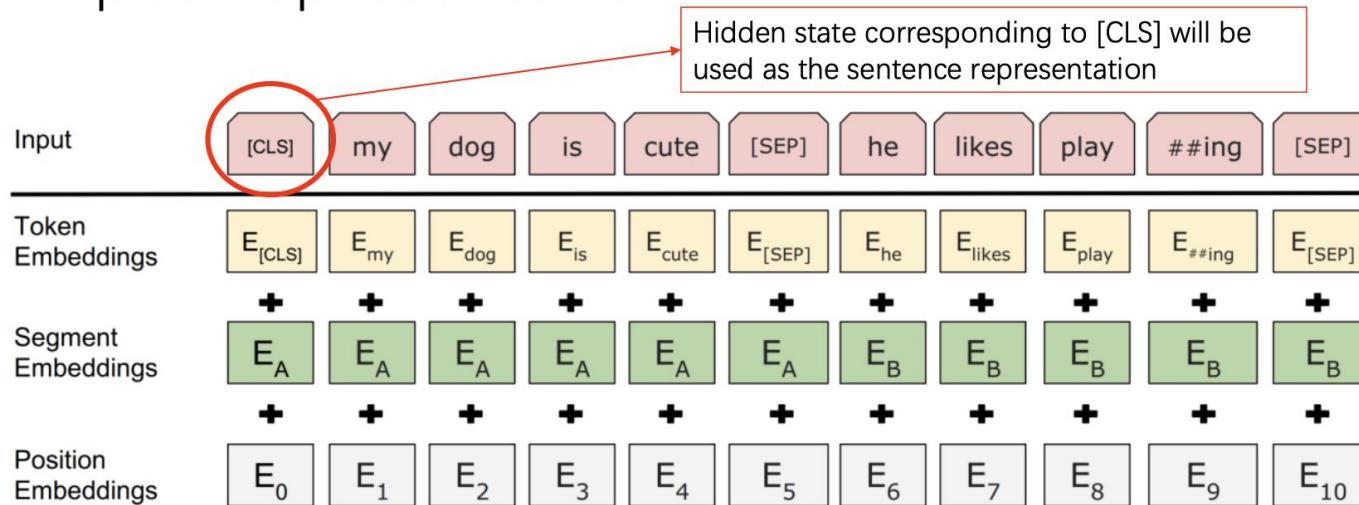
Pre-training



SAPIENZA
UNIVERSITÀ DI ROMA

Input Representation

Input Representation



- Token Embeddings: WordPiece embedding (Wu *et al.*, 2016)
- Segment Embeddings: randomly initialized and learned; single sentence input only adds E_A
- Position embeddings: randomly initialized and learned

What you see

Words vs Tokens

A word is most likely what you think it is - the most simple form or unit of language as understood by humans. In the sentence, "I like cats", there are three words - "I", "like", and "cats." We can think of words as the primary building blocks of language; the fundamental pieces of language that we are taught from a very young age.

A token is a bit more complex. *Tokenization* is the process of converting pieces of language into bits of data that are usable for a program, and a *tokenizer* is an algorithm or function that performs this process, i.e., takes language and converts it into these usable bits of data. Thus, a token is a unit of text that is intentionally segmented for a large language model to process efficiently. These units can be words or any other subset of language - parts of words, combinations of words, or punctuation.

There are a variety of different tokenizers out there which reflect a variety of trade offs. Well-known tokenizers include NLTK (Natural Language Toolkit), Spacy, BERT tokenizer and Keras. Whether or not to select one of these or a different tokenizer depends upon your specific use case. On average, there are roughly 0.75 words per token, but there can be meaningful differences among tokenizers.

What LLM sees





```
✓ 6s
1 from transformers import AutoTokenizer
2
3 # Load the tokenizer for a publicly available LLM (e.g., GPT-2)
4 tokenizer = AutoTokenizer.from_pretrained("gpt2")
5
6 # Sample text to tokenize
7 text = "Hugging Face's transformers library makes NLP easy!"
8
9 # Tokenize the text
10 tokens = tokenizer.tokenize(text)
11
12 # Convert tokens to IDs
13 token_ids = tokenizer.convert_tokens_to_ids(tokens)
14
15 # Print results
16 print("Original Text:", text)
17 print("Tokens:", tokens)
18 print("Token IDs:", token_ids)
19
20 # Decode to check if the process is reversible
21 decoded_text = tokenizer.decode(token_ids)
22 print("Decoded Text:", decoded_text)
```



tokenizer_config.json: 100%  26.0/26.0 [00:00<00:00, 521B/s]

config.json: 100%  665/665 [00:00<00:00, 46.8kB/s]

vocab.json: 100%  1.04M/1.04M [00:00<00:00, 18.5MB/s]

merges.txt: 100%  456k/456k [00:00<00:00, 8.84MB/s]

tokenizer.json: 100%  1.36M/1.36M [00:00<00:00, 22.4MB/s]

Original Text: Hugging Face's transformers library makes NLP easy!

Tokens: ['Hug', 'ging', 'Face', "'s", 'transform', 'ers', 'library', 'makes', 'NLP', 'easy', '!']

Token IDs: [48098, 2667, 15399, 338, 6121, 364, 5888, 1838, 399, 19930, 2562, 0]

Decoded Text: Hugging Face's transformers library makes NLP easy!



✓



```
1 from transformers import AutoTokenizer
2
3 # Load the tokenizer for a publicly available LLM (e.g., GPT-2)
4 tokenizer = AutoTokenizer.from_pretrained("gpt2")
5
6 # Sample text to tokenize
7 text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
8
9 # Tokenize the text
10 tokens = tokenizer.tokenize(text)
11
12 # Convert tokens to IDs
13 token_ids = tokenizer.convert_tokens_to_ids(tokens)
14
15 # Print results
16 print("Original Text:", text)
17 print("Tokens:", tokens)
18 print("Token IDs:", token_ids)
19
20 # Decode to check if the process is reversible
21 decoded_text = tokenizer.decode(token_ids)
22 print("Decoded Text:", decoded_text)
```



Original Text: ABCDEFGHIJKLMNOPQRSTUVWXYZ

Tokens: ['ABC', 'DEF', 'GH', 'IJ', 'K', 'LM', 'N', 'OP', 'Q', 'R', 'ST', 'U', 'VW', 'XY', 'Z']

Token IDs: [24694, 32988, 17511, 23852, 42, 31288, 45, 3185, 48, 49, 2257, 52, 30133, 34278, 57]

Decoded Text: ABCDEFGHIJKLMNOPQRSTUVWXYZ



SAPIENZA
UNIVERSITÀ DI ROMA



```
1 from transformers import AutoTokenizer
2
3 # Load the tokenizer for a publicly available LLM (e.g., GPT-2)
4 tokenizer = AutoTokenizer.from_pretrained("gpt2")
5
6 # Sample text to tokenize
7 text = "Key in Italian is chiave"
8
9 # Tokenize the text
10 tokens = tokenizer.tokenize(text)
11
12 # Convert tokens to IDs
13 token_ids = tokenizer.convert_tokens_to_ids(tokens)
14
15 # Print results
16 print("Original Text:", text)
17 print("Tokens:", tokens)
18 print("Token IDs:", token_ids)
19
20 # Decode to check if the process is reversible
21 decoded_text = tokenizer.decode(token_ids)
22 print("Decoded Text:", decoded_text)
```

→ Original Text: Key in Italian is chiave
Tokens: ['Key', 'In', 'Italian', 'is', 'chiave']
Token IDs: [9218, 287, 8200, 318, 442, 544, 303]
Decoded Text: Key in Italian is chiave



✓
0s

▶

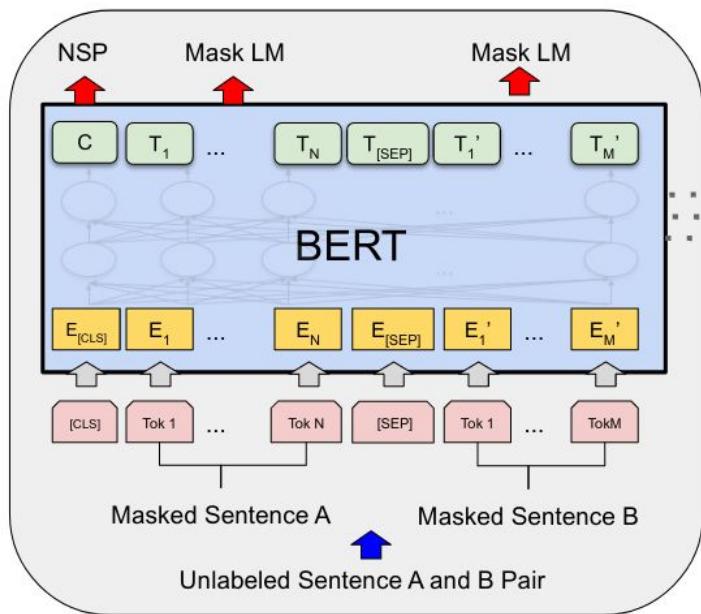
```
1 from transformers import AutoTokenizer
2
3 # Load the tokenizer for a publicly available LLM (e.g., GPT-2)
4 tokenizer = AutoTokenizer.from_pretrained("gpt2")
5
6 # Sample text to tokenize
7 text = "missppeellinggs are tokenized as well"
8
9 # Tokenize the text
10 tokens = tokenizer.tokenize(text)
11
12 # Convert tokens to IDs
13 token_ids = tokenizer.convert_tokens_to_ids(tokens)
14
15 # Print results
16 print("Original Text:", text)
17 print("Tokens:", tokens)
18 print("Token IDs:", token_ids)
19
20 # Decode to check if the process is reversible
21 decoded_text = tokenizer.decode(token_ids)
22 print("Decoded Text:", decoded_text)
```

→ Original Text: missppeellinggs are tokenized as well
Tokens: ['miss', 'ppe', 'elling', 'gs', 'Gare', 'Gtoken', 'ized', 'Gas', 'Gwell']
Token IDs: [3927, 27768, 9417, 14542, 389, 11241, 1143, 355, 880]
Decoded Text: missppeellinggs are tokenized as well

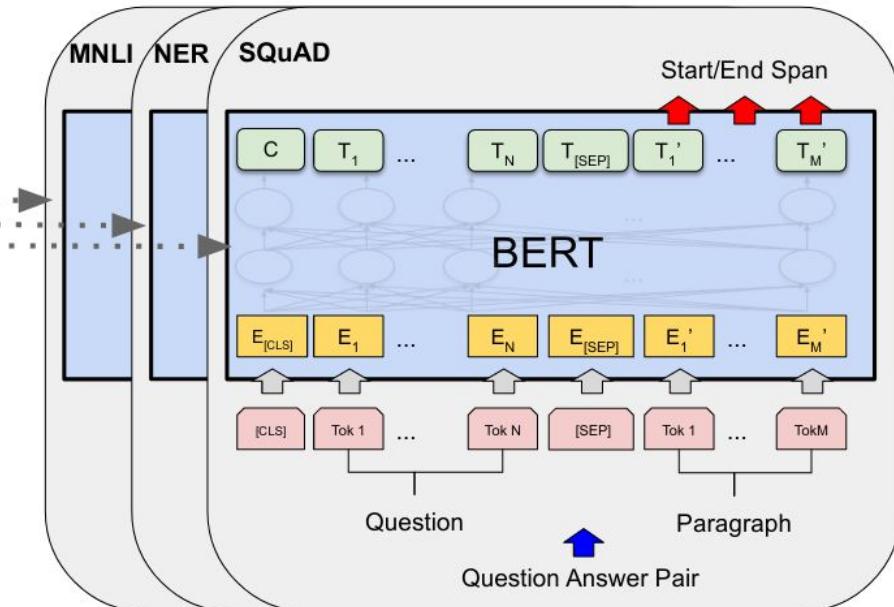


SAPIENZA
UNIVERSITÀ DI ROMA

Fine Tuning BERT



Pre-training



Fine-Tuning



SAPIENZA
UNIVERSITÀ DI ROMA

Fine Tuning BERT: Sequence Classification

The pre-trained model is trained on a supervised dataset to predict the class of a given sequence. Since the output of the BERT (Transformer Encoder) model is the hidden state for all the tokens in the sequence, the output needs to be pooled to obtain only one label. The Classification token ([CLS] token) is used here. The output of this token is considered as the classifier pooled output and it is further put into a fully-connected classification layer for obtaining the labeled output.



Fine Tuning BERT: Named Entity Recognition (NER)

The hidden state outputs are directly put into a classifier layer with the number of tags as the output units for each of the token. Then these logits are used to obtain the predicted class of each token using argmax.

Fine Tuning BERT: Natural Language Inference (NLI) or Textual Entailment

We train BERT the same as in the NSP task, with both the sentences i.e. the text and the hypothesis separated using [SEP] token, and are identified using the Segment Embeddings. The [CLS] token is used to obtain the classification result as explained in the Sequence Classification part.

Fine Tuning BERT: Grounded Common Sense Inference

Given a sentence, the task is to choose the most plausible continuation among four choices. We take 4 input sequences, each containing the original sentence and the corresponding choice concatenated to it. Here too, we use the [CLS] token state and feed it to a fully-connected layer to obtain the scores for each of the choices which are then normalized using a softmax layer.



Fine Tuning BERT: Question Answering

A paragraph is used as one sequence and the question is used as another. There are 2 cases in this task:

1. The answer is expected to be found within the paragraph. Here, we intend to find the Start and the End token of the answer from the paragraph. For this, we take the dot product of each token T_i and the start token S to obtain the probability of the token i to be the start of the answer. Similarly, we obtain the probability of the end token j . The score of a candidate span from position i to position j is defined as $S.T_i + E.T_j$, and the maximum scoring span where $j \geq i$ is used as a prediction.
2. we consider the possibility that there may be no short answer to the question present in the paragraph, which is a more realistic case. For this, we consider the probability scores for the start and end of the answer at the [CLS] token itself. We call this as s_{null} . We compare this s_{null} with the max score obtained at the best candidate span (i.e. the best score for the first case). If this obtained score is greater than s_{null} by a sufficient threshold τ , then we use the candidate best score as the answer. The threshold τ can be tuned to maximize the dev set F1-score.



Experiments: GLUE Tests

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Average
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT _{BASE}	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT _{LARGE}	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1

Table 1: GLUE Test results, scored by the evaluation server (<https://gluebenchmark.com/leaderboard>). The number below each task denotes the number of training examples. The “Average” column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.⁸ BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

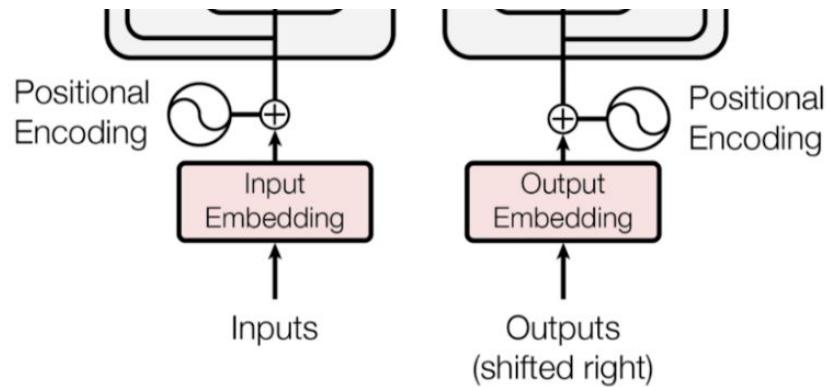


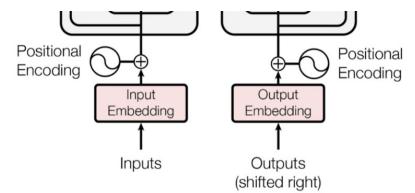
Positional Embeddings



SAPIENZA
UNIVERSITÀ DI ROMA

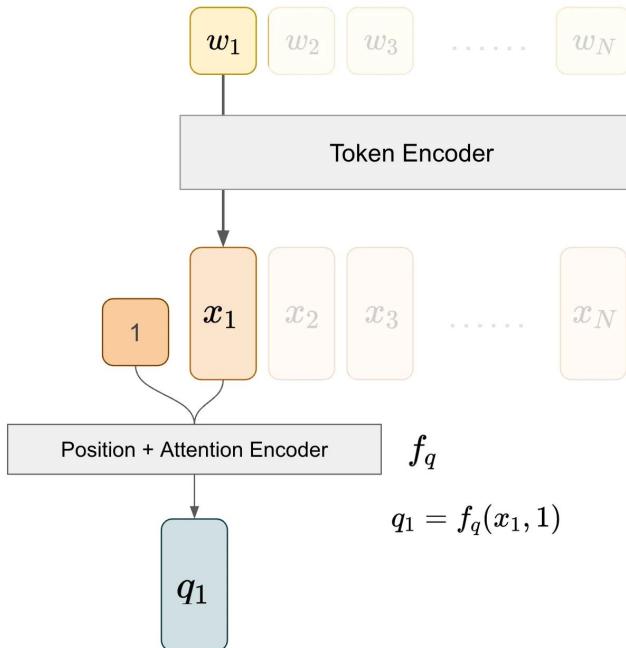
How to Encode the Position of a Token?

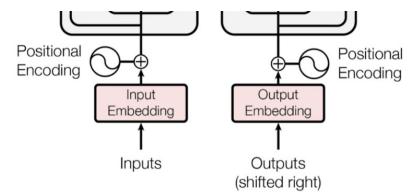




Let's Get There Step-by-Step

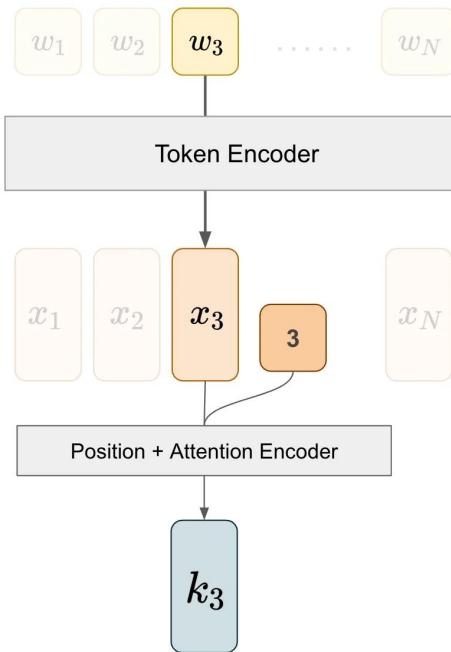
- We obtain the attention score by taking the inner product of q and k , and this inner product is key to how position embeddings function.
- For example, to get the attention score for the pair $(1, 3)$, we get the query vector from token 1 and the key vector from token 3.
 - We obtain the query vector q_1 by first extracting its token embedding through the token encoder. Then, we feed this embedding and its positional information into the position+attention encoder, which integrates position information and projects the result to produce the key vector.

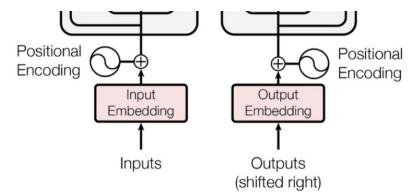




Let's Get There Step-by-Step

- We perform a similar process for the third token to obtain k_3 , the key vector corresponding to token 3.





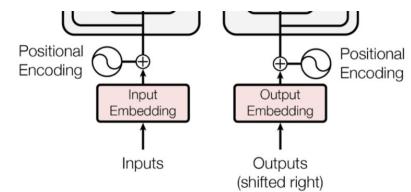
Let's Get There Step-by-Step

- Finally, we take the inner product of q_1 and k_3 to determine the attention score for (1, 3).
- Angle brackets $\langle \rangle$ denote the inner product, x represents the token embedding, and f is the attention+position encoder.

$$q_1 \cdot k_3 = (1, 3)$$

$$\langle f_q(\mathbf{x}_m, \mathbf{m}), f_k(\mathbf{x}_n, \mathbf{n}) \rangle = score(m, n)$$





Original Positional Encoding

$$P(k, 2i) = \sin\left(\frac{k}{n^{2i/d}}\right)$$

$$P(k, 2i + 1) = \cos\left(\frac{k}{n^{2i/d}}\right)$$

- L : is the length of the sequence
- k : Position of an object in the input sequence, k in $[0, L/2]$
- d : Dimension of the output embedding space
- $P(k,j)$: Position function for mapping a position in the input sequence to index of the positional matrix
- n : User-defined scalar, set to 10,000 by the authors of [Attention Is All You Need](#).
- i : Used for mapping to token indices, with a single value of maps to both sine and cosine functions



$$P(k, 2i) = \sin\left(\frac{k}{n^{2i/d}}\right)$$

$$P(k, 2i + 1) = \cos\left(\frac{k}{n^{2i/d}}\right)$$

Original Positional Encoding

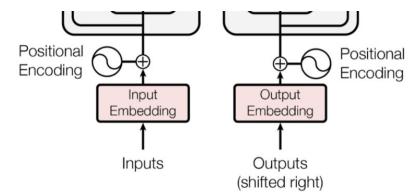
Index
of token,
 k

Positional Encoding
Matrix with $d=4$, $n=100$

Sequence	k	$i=0$	$i=0$	$i=1$	$i=1$
		$P_{00}=\sin(0)$ = 0	$P_{01}=\cos(0)$ = 1	$P_{02}=\sin(0)$ = 0	$P_{03}=\cos(0)$ = 1
I	0	$P_{10}=\sin(1/1)$ = 0.84	$P_{11}=\cos(1/1)$ = 0.54	$P_{12}=\sin(1/10)$ = 0.10	$P_{13}=\cos(1/10)$ = 1.0
am	1	$P_{20}=\sin(2/1)$ = 0.91	$P_{21}=\cos(2/1)$ = -0.42	$P_{22}=\sin(2/10)$ = 0.20	$P_{23}=\cos(2/10)$ = 0.98
a	2	$P_{30}=\sin(3/1)$ = 0.14	$P_{31}=\cos(3/1)$ = -0.99	$P_{32}=\sin(3/10)$ = 0.30	$P_{33}=\cos(3/10)$ = 0.96
Robot	3				

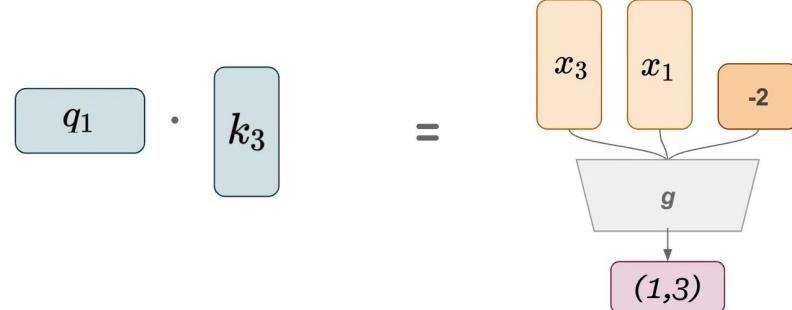
Positional Encoding Matrix for the sequence 'I am a robot'





Once We Get Here: Position is Already In

- the relative positional information is encoded before the inner product: it's inherently tied to the token embedding
- Is there another way to encode relative positional information only at the moment we perform the $\langle q, k \rangle$ inner product?
 - Or equivalently, can we find a function g that takes only the token embeddings and their positions as input?



$$\langle f_q(\mathbf{x}_m, \mathbf{m}), f_k(\mathbf{x}_n, \mathbf{n}) \rangle = g(\mathbf{x}_m, \mathbf{x}_n, \mathbf{m} - \mathbf{n}).$$



RoFORMER: ENHANCED TRANSFORMER WITH ROTARY POSITION EMBEDDING

Jianlin Su

Zhuiyi Technology Co., Ltd.
Shenzhen
bojonesu@wezhuiyi.com

Yu Lu

Zhuiyi Technology Co., Ltd.
Shenzhen
julianlu@wezhuiyi.com

Shengfeng Pan

Zhuiyi Technology Co., Ltd.
Shenzhen
nickpan@wezhuiyi.com

Ahmed Murtadha

Zhuiyi Technology Co., Ltd.
Shenzhen
mengjiayi@wezhuiyi.com

Bo Wen

Zhuiyi Technology Co., Ltd.
Shenzhen
brucewen@wezhuiyi.com

Yunfeng Liu

Zhuiyi Technology Co., Ltd.
Shenzhen
glenliu@wezhuiyi.com

November 9, 2023

ABSTRACT

Position encoding recently has shown effective in the transformer architecture. It enables valuable supervision for dependency modeling between elements at different positions of the sequence. In this paper, we first investigate various methods to integrate positional information into the learning process of transformer-based language models. Then, we propose a novel method named Rotary Position Embedding(RoPE) to effectively leverage the positional information. Specifically, the proposed RoPE encodes the absolute position with a rotation matrix and meanwhile incorporates the explicit relative position dependency in self-attention formulation. Notably, RoPE enables valuable properties, including the flexibility of sequence length, decaying inter-token dependency with increasing relative distances, and the capability of equipping the linear self-attention with relative position encoding. Finally, we evaluate the enhanced transformer with rotary position embedding, also called RoFormer, on various long text classification benchmark datasets. Our experiments show that it consistently overcomes its alternatives. Furthermore, we provide a theoretical analysis to explain some experimental results. RoFormer is already integrated into Huggingface: https://huggingface.co/docs/transformers/model_doc/roformer.

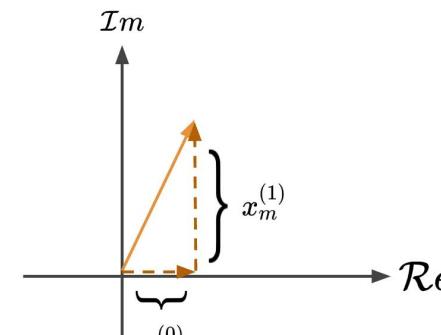


SAPIENZA
UNIVERSITÀ DI ROMA

A 2D Intuition

- Token embeddings and attention vectors (query, key) all reside in 2D space.
- For convenience, these 2D vectors can also be represented using a complex number (as shown in the figure).

Token Embedding Dimension = 2

$$x_m = \begin{pmatrix} x_m^{(0)} \\ x_m^{(1)} \end{pmatrix} = x_m^{(0)} + x_m^{(1)} j =$$


(0) $x_m^{(1)}$

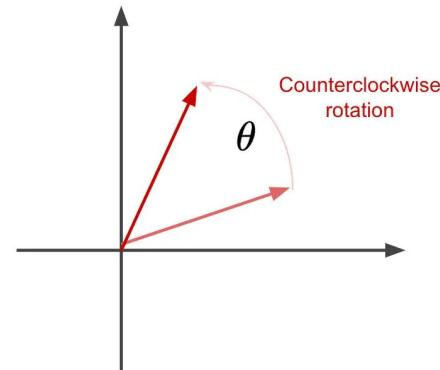


Rotation Matrices

- counterclockwise rotation matrix can be expressed both in a matrix form and in an exponential form.

Token Embedding Dimension = 2

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = e^{j\theta} =$$



Building the Function g

- We want to satisfy the following condition:

$$\langle f_q(\mathbf{x}_m, \mathbf{m}), f_k(\mathbf{x}_n, \mathbf{n}) \rangle = g(\mathbf{x}_m, \mathbf{x}_n, \mathbf{m} - \mathbf{n})$$

- We can pick f_q and f_k such that

$$f_q(\mathbf{x}_m, m) = (\mathbf{W}_q \mathbf{x}_m) e^{im\theta}$$

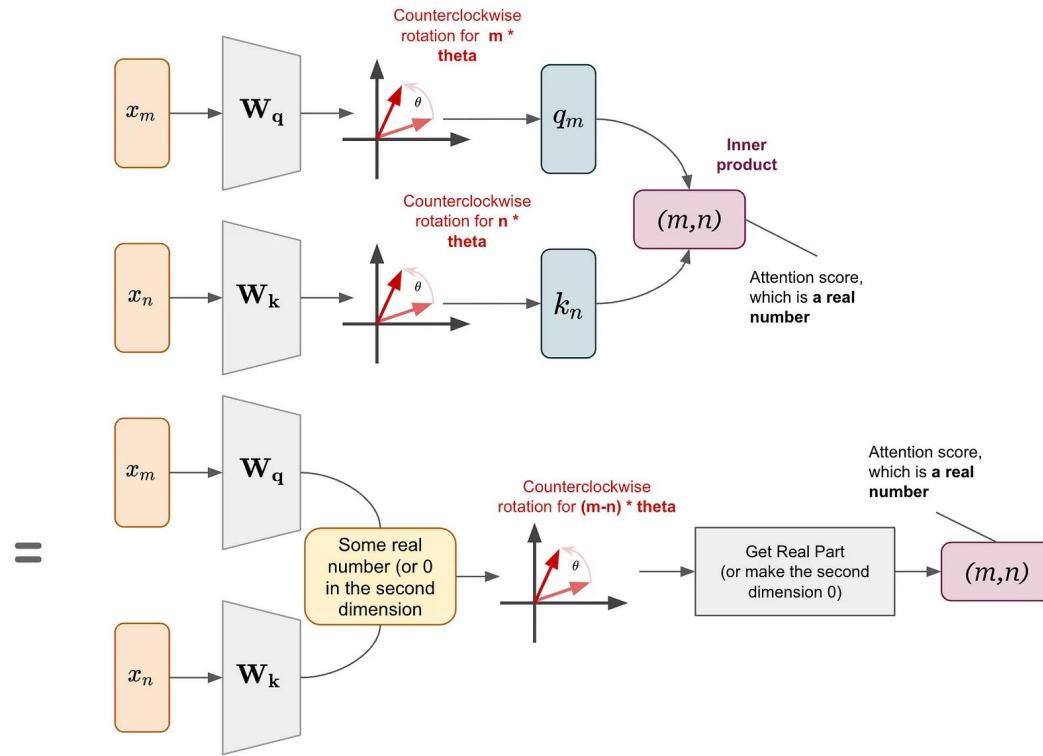
$$f_k(\mathbf{x}_n, n) = (\mathbf{W}_k \mathbf{x}_n) e^{in\theta}$$

- This way we have the following g

$$g(\mathbf{x}_m, \mathbf{x}_n, m - n) = \operatorname{Re}[(\mathbf{W}_q \mathbf{x}_m)(\mathbf{W}_k \mathbf{x}_n)^* e^{i(m-n)\theta}]$$



Building the Function g (in Graphical Form)



We only need $(m-n)$ for the rotation, which signifies that this is a type of relative position embedding.



Generalization to any Dimension

$$\mathbf{R}_{\Theta,m}^d = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix}$$

$$f_{\{q,k\}}(\mathbf{x}_m, m) = \mathbf{R}_{\Theta,m}^d \mathbf{W}_{\{q,k\}} \mathbf{x}_m$$



Effects of RoPE

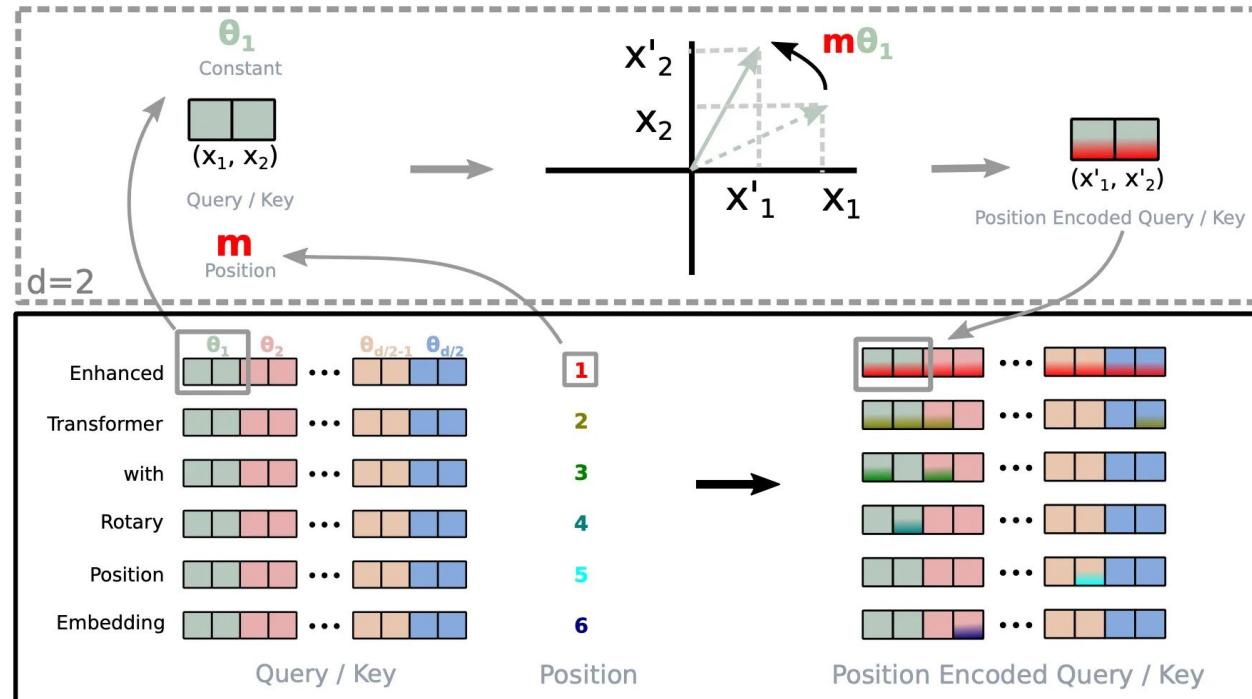
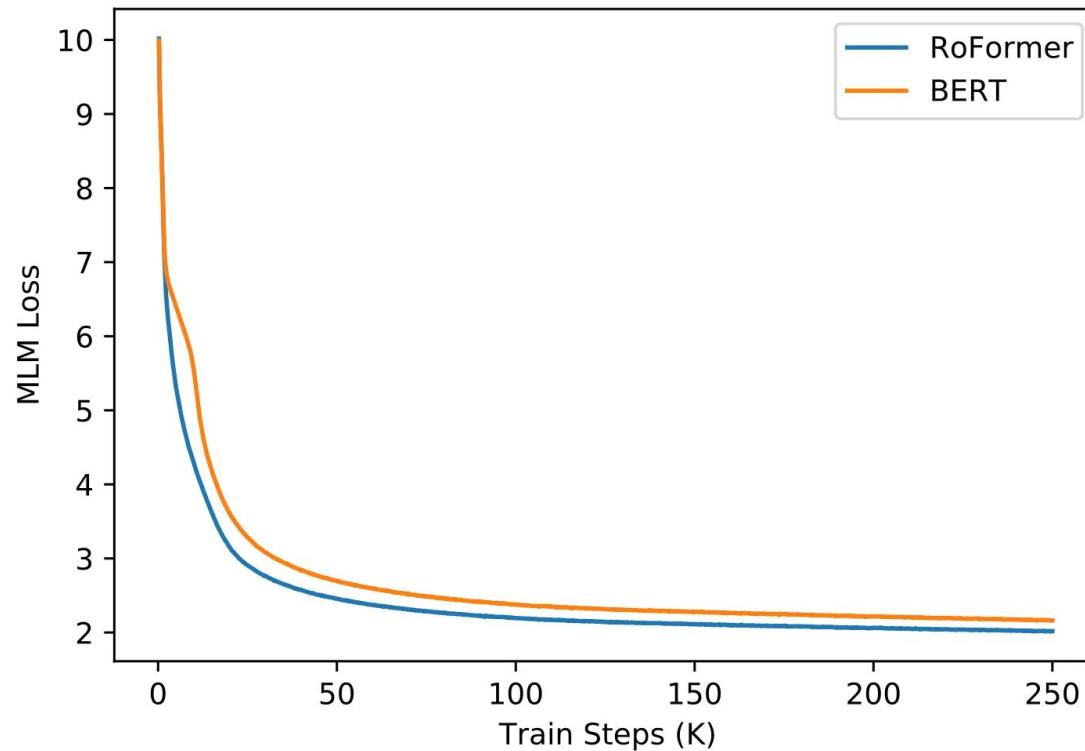


Figure 1: Implementation of Rotary Position Embedding(RoPE).



Performance



Performance

Table 2: Comparing RoFormer and BERT by fine tuning on downstream GLEU tasks.

Model	MRPC	SST-2	QNLI	STS-B	QQP	MNLI(m/mm)
BERT Devlin et al. [2019]	88.9	93.5	90.5	85.8	71.2	84.6/83.4
RoFormer	89.5	90.7	88.0	87.0	86.4	80.2/79.8

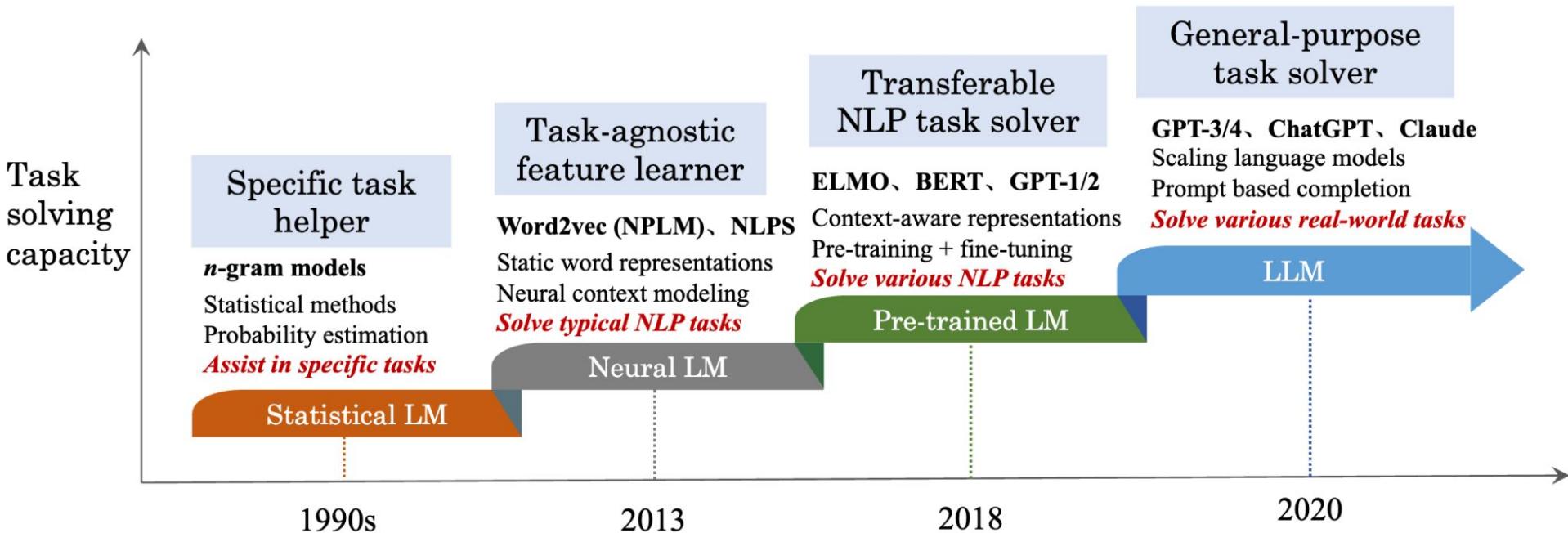


Decoder Only LLMs



SAPIENZA
UNIVERSITÀ DI ROMA

A Quick Recap on Where we are



From: <https://arxiv.org/pdf/2303.18223>



Where (almost) everything started...

Language Models are Few-Shot Learners

Tom B. Brown* Benjamin Mann* Nick Ryder* Melanie Subbiah*

Jared Kaplan† Prafulla Dhariwal Arvind Neelakantan Pranav Shyam

Girish Sastry Amanda Askell Sandhini Agarwal Ariel Herbert-Voss

Gretchen Krueger Tom Henighan Rewon Child Aditya Ramesh

Daniel M. Ziegler Jeffrey Wu Clemens Winter

Christopher Hesse Mark Chen Eric Sigler Mateusz Litwin Scott Gray

Benjamin Chess Jack Clark Christopher Berner

Sam McCandlish Alec Radford Ilya Sutskever Dario Amodei

From: Language Models are Few-Shot Learners



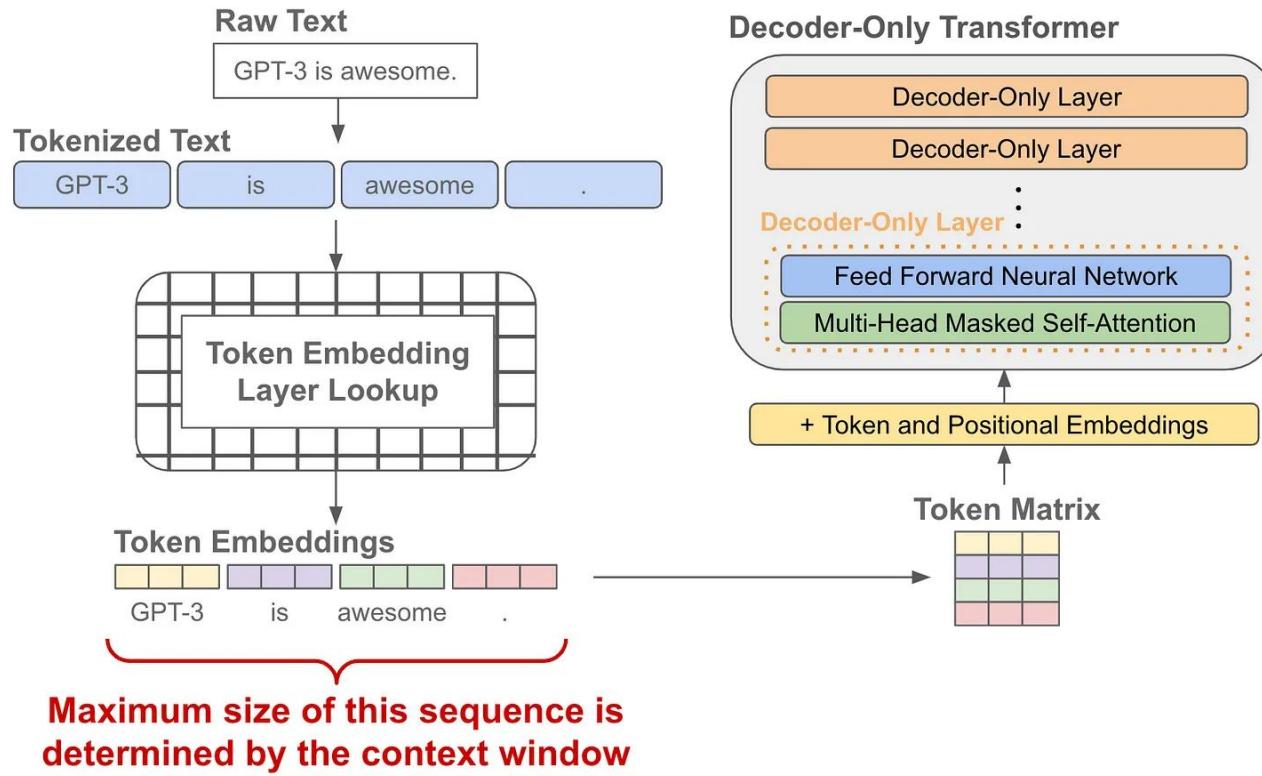
SAPIENZA
UNIVERSITÀ DI ROMA

Where (almost) everything started...

- **Model Architecture:**
 - Introduced GPT-3, an autoregressive language model with 175 billion parameters, significantly larger than previous models.
- **Few-Shot Learning:**
 - Demonstrated that GPT-3 can perform various NLP tasks—such as translation, question-answering, and cloze tasks—with task-specific fine-tuning, relying instead on few-shot, one-shot, or zero-shot learning through textual prompts.
- **Performance:**
 - Achieved strong performance on many NLP benchmarks, sometimes matching or exceeding state-of-the-art models that were fine-tuned on specific tasks.
- **Limitations:**
 - Identified areas where GPT-3's few-shot learning struggles, particularly in tasks requiring robust reasoning or domain-specific knowledge, and highlighted concerns about biases and inaccuracies inherited from training data.



GPT is Decoder-Only (implements causal LMs)



Parameter Size

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}



Dataset Used

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4



Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description  
2 cheese => ..... ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description  
2 sea otter => loutre de mer ← example  
3 cheese => ..... ← prompt
```

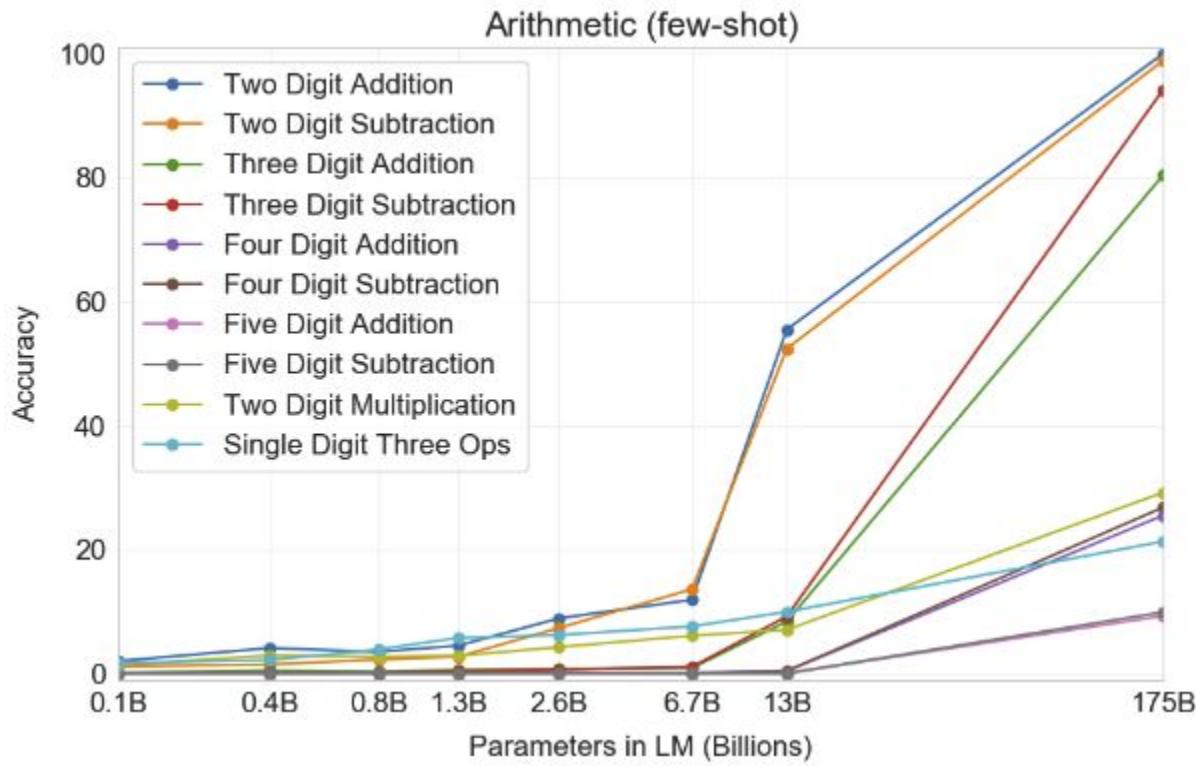
Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description  
2 sea otter => loutre de mer ← examples  
3 peppermint => menthe poivrée  
4 plush girafe => girafe peluche  
5 cheese => ..... ← prompt
```



A Novel Task: Arithmetic



A Novel Task: Arithmetic

- **3 digit subtraction (3D-)** – Same as 2 digit subtraction, except numbers are uniformly sampled from [0, 1000).
- **4 digit addition (4D+)** – Same as 3 digit addition, except uniformly sampled from [0, 10000).
- **4 digit subtraction (4D-)** – Same as 3 digit subtraction, except uniformly sampled from [0, 10000).
- **5 digit addition (5D+)** – Same as 3 digit addition, except uniformly sampled from [0, 100000).
- **5 digit subtraction (5D-)** – Same as 3 digit subtraction, except uniformly sampled from [0, 100000).
- **2 digit multiplication (2Dx)** – The model is asked to multiply two integers sampled uniformly from [0, 100), e.g. “Q: What is 24 times 42? A: 1008”.
- **One-digit composite (1DC)** – The model is asked to perform a composite operation on three 1 digit numbers, with parentheses around the last two. For example, “Q: What is $6+(4*8)$? A: 38”. The three 1 digit numbers are selected uniformly on [0, 10) and the operations are selected uniformly from {+, -, *}.



Generating News Article (Difficult to spot)

Title: United Methodists Agree to Historic Split

Subtitle: Those who oppose gay marriage will form their own denomination

Article: After two days of intense debate, the United Methodist Church has agreed to a historic split - one that is expected to end in the creation of a new denomination, one that will be "theologically and socially conservative," according to The Washington Post. The majority of delegates attending the church's annual General Conference in May voted to strengthen a ban on the ordination of LGBTQ clergy and to write new rules that will "discipline" clergy who officiate at same-sex weddings. But those who opposed these measures have a new plan: They say they will form a separate denomination by 2020, calling their church the Christian Methodist denomination.

The Post notes that the denomination, which claims 12.5 million members, was in the early 20th century the "largest Protestant denomination in the U.S.," but that it has been shrinking in recent decades. The new split will be the second in the church's history. The first occurred in 1968, when roughly 10 percent of the denomination left to form the Evangelical United Brethren Church. The Post notes that the proposed split "comes at a critical time for the church, which has been losing members for years," which has been "pushed toward the brink of a schism over the role of LGBTQ people in the church." Gay marriage is not the only issue that has divided the church. In 2016, the denomination was split over ordination of transgender clergy, with the North Pacific regional conference voting to ban them from serving as clergy, and the South Pacific regional conference voting to allow them.

Figure 3.14: The GPT-3 generated news article that humans had the greatest difficulty distinguishing from a human written article (accuracy: 12%).



Generating News Article (Easy to spot)

Title: Star's Tux Promise Draws Megyn Kelly's Sarcasm

Subtitle: Joaquin Phoenix pledged to not change for each awards event

Article: A year ago, Joaquin Phoenix made headlines when he appeared on the red carpet at the Golden Globes wearing a tuxedo with a paper bag over his head that read, "I am a shape-shifter. I can't change the world. I can only change myself." It was a promise to not change to fit into the Hollywood mold: "I think that's a really special thing, to not change yourself. I think it's a really special thing to say, 'This is what's inside of me, I'm proud of it, and I'm not going to be ashamed because of the way that someone else thinks I should be.'" Now, it's the Oscars, and Phoenix is at it again. But this time, his publicist is saying he'll be wearing a tux no matter what.

Megyn Kelly was not impressed, and she let him have it on The Tonight Show. "You know, I feel like, I feel like you could have worn the tux," she says. "But you're saying you're a shape-shifter. I don't know if you can change your tux, but you can change your mind. You can change your mind. You can change your mind." Phoenix says he did, but it didn't stick. "I was like, 'Okay, I'm going to wear a tuxedo to this thing.' And then I thought, 'I don't want to wear a tuxedo to this thing.'" Kelly goes on to encourage him to change his mind again, but Phoenix says it's too late: "I'm committed to wearing this."

Figure 3.15: The GPT-3 generated news article that humans found the easiest to distinguish from a human written article (accuracy: 61%).



Generating Completions with “New” Words

A "whatpu" is a small, furry animal native to Tanzania. An example of a sentence that uses the word whatpu is:

We were traveling in Africa and we saw these very cute whatpus.

To do a "farduddle" means to jump up and down really fast. An example of a sentence that uses the word farduddle is:

One day when I was playing tag with my little sister, she got really excited and she started doing these crazy farduckles.

A "yalubalu" is a type of vegetable that looks like a big pumpkin. An example of a sentence that uses the word yalubalu is:

I was on a trip to Africa and I tried this yalubalu vegetable that was grown in a garden there. It was delicious.

A "Burringo" is a car with very fast acceleration. An example of a sentence that uses the word Burringo is:

In our garage we have a Burringo that my father drives to work every day.

A "Gigamuru" is a type of Japanese musical instrument. An example of a sentence that uses the word Gigamuru is:

I have a Gigamuru that my uncle gave me as a gift. I love to play it at home.

To "screeg" something is to swing a sword at it. An example of a sentence that uses the word screeg is:

We screeghed at each other for several minutes and then we went outside and ate ice cream.



Correcting English Grammar (few-shot)

Poor English input: I eated the purple berries.
Good English output: I ate the purple berries.

Poor English input: Thank you for picking me as your designer. I'd appreciate it.
Good English output: Thank you for choosing me as your designer. I appreciate it.

Poor English input: The mentioned changes have done. or I did the alteration that you requested. or I changed things you wanted and did the modifications.
Good English output: The requested changes have been made. or I made the alteration that you requested. or I changed things you wanted and made the modifications.

Poor English input: I'd be more than happy to work with you in another project.
Good English output: I'd be more than happy to work with you on another project.

Poor English input: Please provide me with a short brief of the design you're looking for and that'd be nice if you could share some examples or project you did before.
Good English output: Please provide me with a brief description of the design you're looking for and that would be nice if you could share some examples or projects you have done before.

Poor English input: The patient was died.
Good English output: The patient died.

Poor English input: We think that Leslie likes ourselves.
Good English output: We think that Leslie likes us.

Poor English input: Janet broke Bill on the finger.
Good English output: Janet broke Bill's finger.

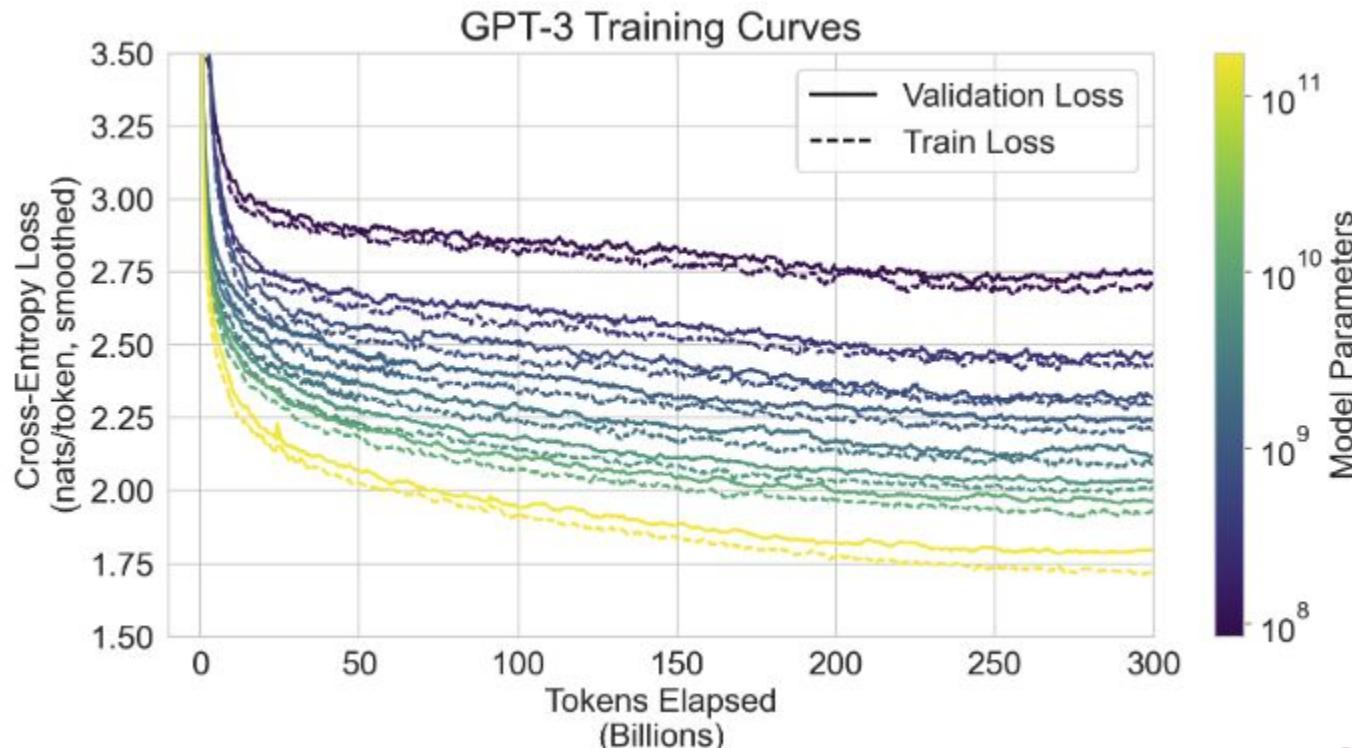
Poor English input: Mary arranged for, in St. Louis, John to rent a house cheap.
Good English output: Mary arranged for John to rent a house in St. Louis.

Poor English input: Today I have went to the store to to buys some many bottle of water.
Good English output: Today I went to the store to buy some bottles of water.

Poor English input: I have tried to hit ball with bat, but my swing is has miss.
Good English output: I tried to hit the ball with the bat, but my swing missed.



Training Curves



Number of Epochs (not GPT-3)

Scaling Data-Constrained Language Models

Niklas Muennighoff¹

Alexander M. Rush¹

Boaz Barak²

Teven Le Scao¹

Aleksandra Piktus¹

Nouamane Tazi¹

Sampo Pyysalo³

Thomas Wolf¹

Colin Raffel¹

¹ Hugging Face

² Harvard University

³ University of Turku

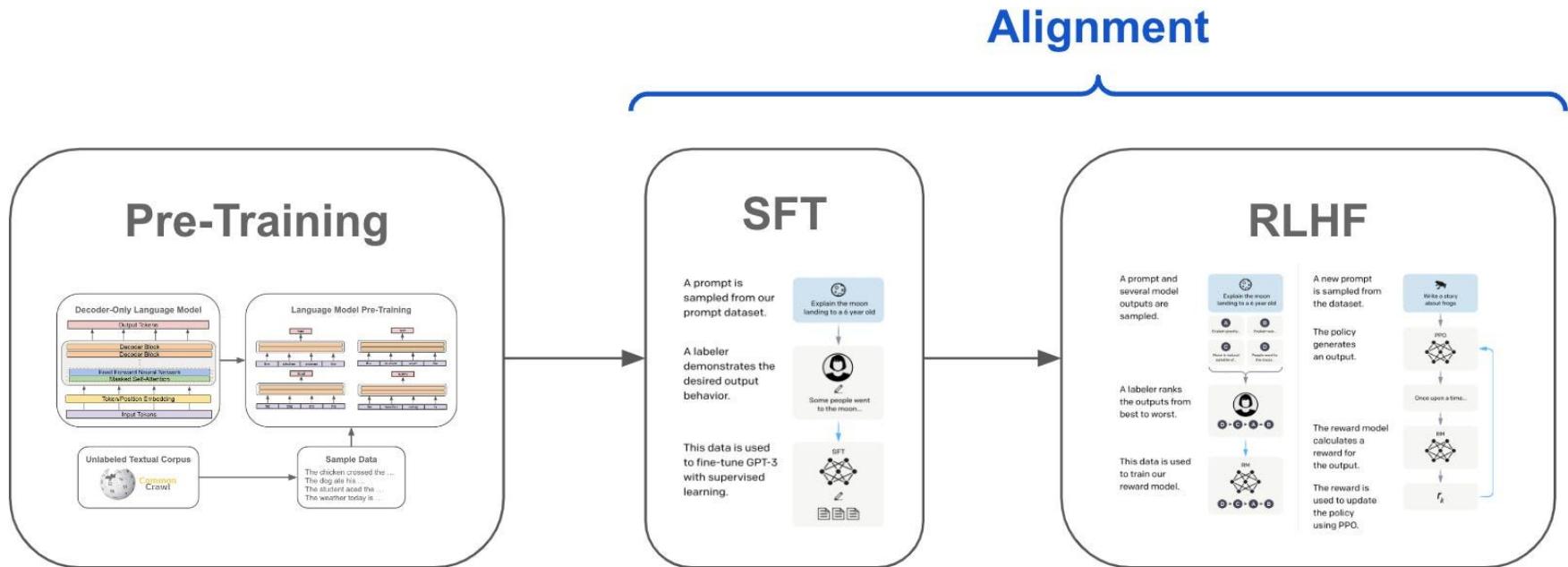
n.muennighoff@gmail.com

Abstract

The current trend of scaling language models involves increasing both parameter count and training dataset size. Extrapolating this trend suggests that training dataset size may soon be limited by the amount of text data available on the internet. Motivated by this limit, we investigate scaling language models in data-constrained regimes. Specifically, we run a large set of experiments varying the extent of data repetition and compute budget, ranging up to 900 billion training tokens and 9 billion parameter models. We find that with constrained data for a fixed compute budget, training with up to 4 epochs of repeated data yields negligible changes to loss compared to having unique data. However, with more repetition, the value of adding compute eventually decays to zero. We propose and empirically validate a scaling law for compute optimality that accounts for the decreasing value of repeated tokens and excess parameters. Finally, we experiment with approaches mitigating data scarcity, including augmenting the training dataset with code data or removing commonly used filters. Models and datasets from our 400 training runs are freely available at <https://github.com/huggingface/datablations>.



Training a Chat LLM – The Typical Pipeline



Interlude: A Better Attention



SAPIENZA
UNIVERSITÀ DI ROMA

Recall Self-Attention

$$\text{softmax} \left(\frac{\text{Q} \times \text{K}^T}{\sqrt{d_k}} \right) \text{V} = \text{z}$$

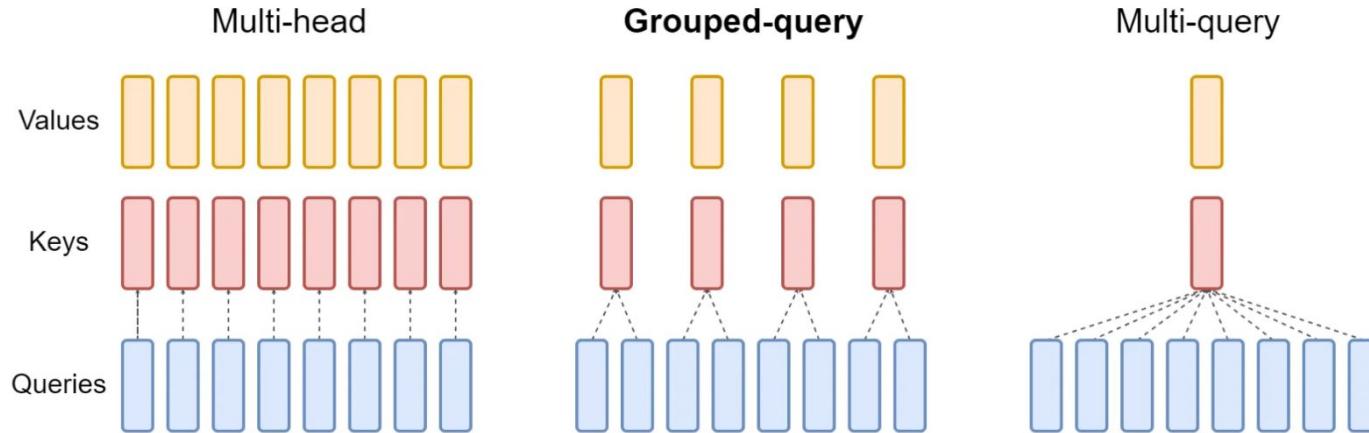
The diagram illustrates the computation of self-attention. It shows three input tensors: Q (purple 3x3 grid), K^T (orange 3x3 grid), and V (blue 3x3 grid). The multiplication of Q and K^T is scaled by $\sqrt{d_k}$. The result is then multiplied by V to produce the output z (pink 3x3 grid).

From: [Grouped-query Attention. GQA achieves quality close to... | by Dylan Wang | Medium](#)



SAPIENZA
UNIVERSITÀ DI ROMA

We Can “Group” Together Different Queries



From: [Grouped-query Attention. GQA achieves quality close to... | by Dylan Wang | Medium](#)



SAPIENZA
UNIVERSITÀ DI ROMA

A Running Example

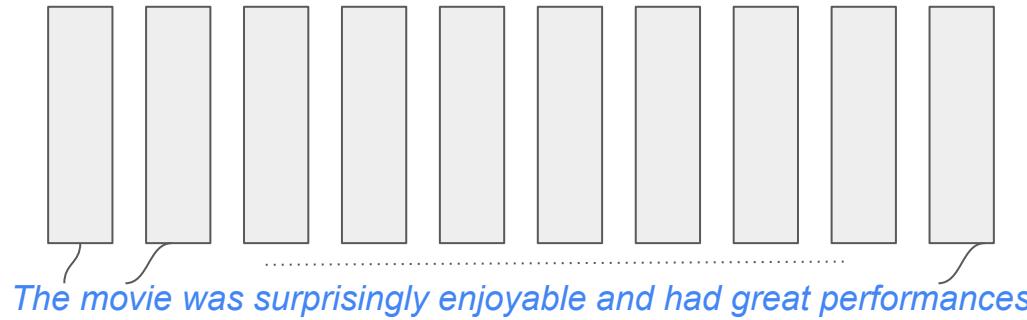
- Consider the following sentence:

The movie was surprisingly enjoyable and had great performances

- The sentence has 10 words.

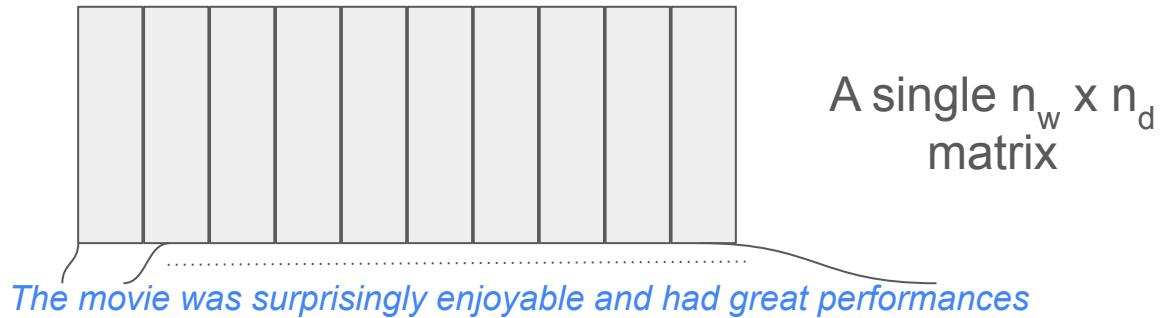
Embedding The Sentence

- Firstly, you convert each word into a word embedding (such as a 300 dimensional vector) to obtain a 10x300 embedding matrix.
- Among them, 10 is the length of the sentence, and 300 is the embedding dimension of each word.



Embedding The Sentence

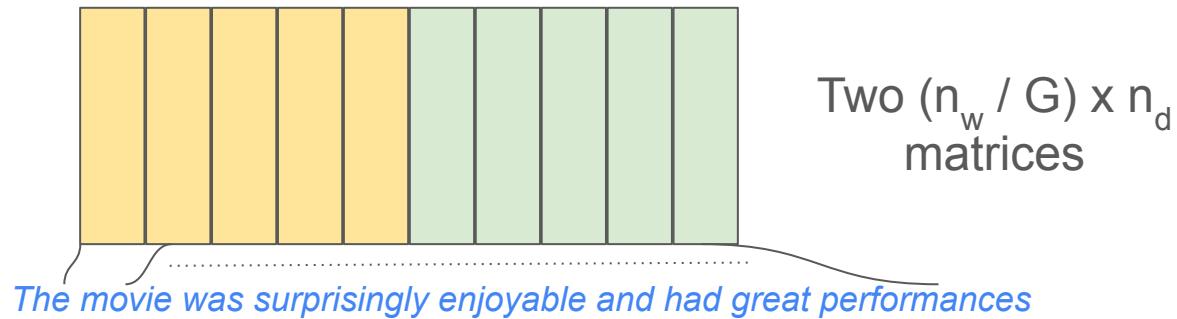
- Firstly, you convert each word into a word embedding (such as a 300 dimensional vector) to obtain a 10×300 embedding matrix.
- Among them, 10 is the length of the sentence, and 300 is the embedding dimension of each word.



Grouping the words

- Group 1: [“The”, “movie”, “was”, “surprisingly”, “enjoyable”]
- Group 2: [“and”, “had”, “great”, “performances”, “.”]

The corresponding embedding matrix is now divided into two 5×300 sub matrices



Groups in Computing Attention Scores

In **GQA**, we group the tokens and share **keys and values within each group**. Let $G = 2$ (since we divide 9 tokens into two groups of 5):

Group 1: [The, movie, was, surprisingly, enjoyable]

Group 2: [and, had, great, performances, (pad)]

For each group $g = 1, 2$, the keys and values are **shared among heads within the same group**, while queries remain independent:

$$Q^{(i)} = XW_i^Q, \quad \forall i = 1, \dots, h,$$

$$K^{(g)} = XW_g^K, \quad \forall g = 1, 2, \quad —$$

$$V^{(g)} = XW_g^V, \quad \forall g = 1, 2.$$

Each head i computes attention using the shared **key-value pair** from its assigned group:

$$A^{(i)} = \text{softmax} \left(\frac{Q^{(i)}(K^{(g(i))})^T}{\sqrt{d_k}} \right) V^{(g(i))}.$$



Benefits of GQA

- **Memory Efficiency:**
 - In the example: instead of computing 9 separate key-value sets (one per token), GQA only computes two sets (one per group).
- **Inference Speedup:**
 - Using shared key-value pairs reduces computational cost without significantly affecting accuracy.
- **Maintained Diversity:**
 - Queries remain independent, preserving the flexibility of MHA while reducing the storage needed for keys and values.

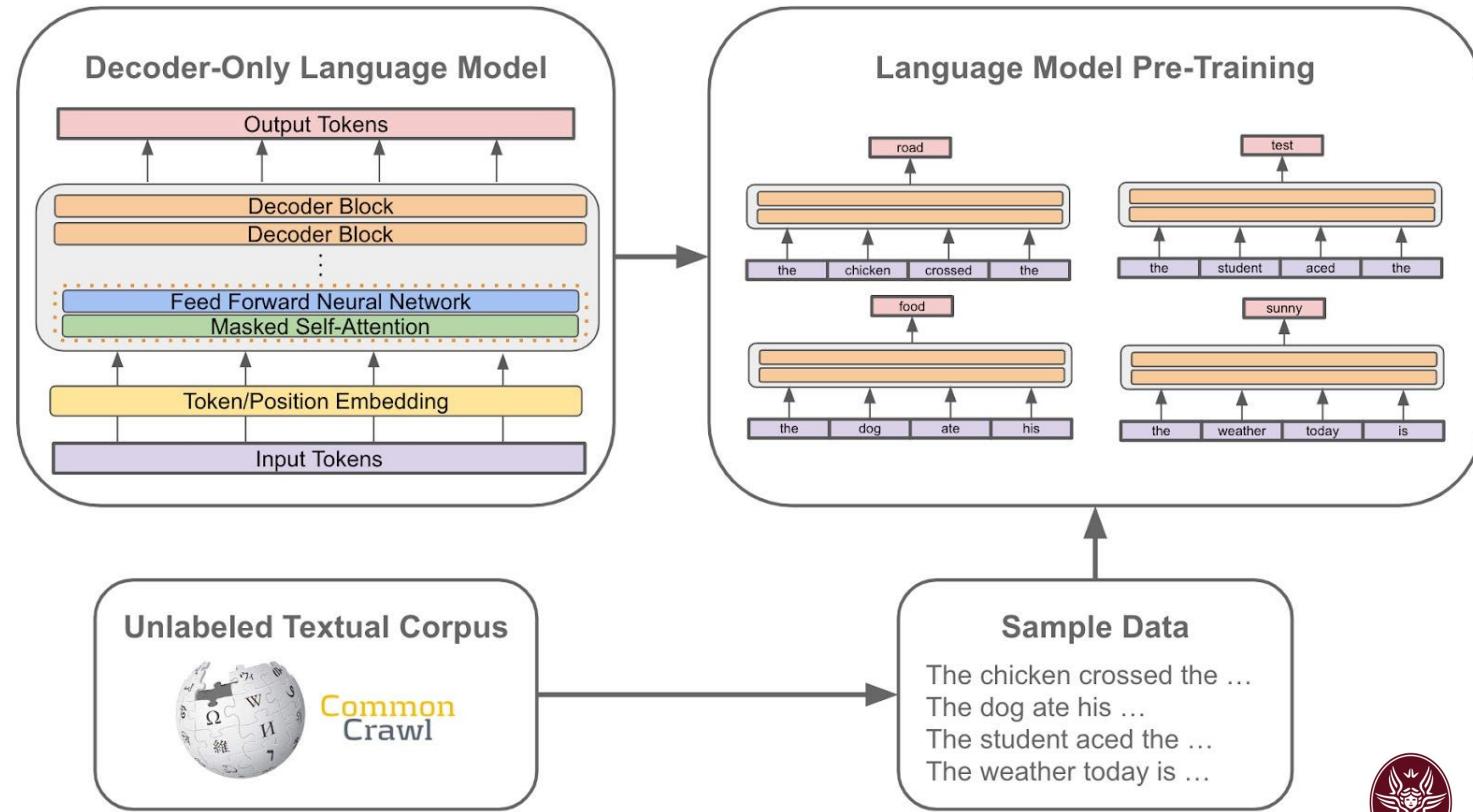


Pretraining



SAPIENZA
UNIVERSITÀ DI ROMA

Pretraining



Pretraining

- ML task: language modeling
- Training data: low-quality data
- Data scale: usually in the order of trillions of tokens as of May 2023.
 - [GPT-3's dataset](#) (OpenAI): 0.5 trillion tokens. I can't find any public info for GPT-4, but I'd estimate it to use an order of magnitude more data than GPT-3.
 - [Gopher's](#) dataset (DeepMind): 1 trillion tokens
 - [RedPajama](#) (Together): 1.2 trillion tokens
 - [LLaMa's dataset](#) (Meta): 1.4 trillion tokens
- Model resulting from this process: LLM
- LLM_ϕ : the language model being trained, parameterized by ϕ . The goal is to find ϕ for which the cross entropy loss is minimized.
- $[T_1, T_2, \dots, T_V]$: vocabulary – the set of all unique tokens in the training data.
- V : the vocabulary size.
- $f(x)$: function mapping a token to its position in the vocab. If x is T_k in the vocab, $f(x) = k$.
- Given the sequence (x_1, x_2, \dots, x_n) , we'll have n training samples:
 - Input: $x = (x_1, x_2, \dots, x_{i-1})$
 - Ground truth: x_i
- For each training sample (x, x_i) :
 - Let $k = f(x_i)$
 - Model's output: $LLM(x) = [\bar{y}_1, \bar{y}_2, \dots, \bar{y}_V]$. Note: $\sum_j \bar{y}_j = 1$
 - The loss value: $CE(x, x_i; \phi) = -\log \bar{y}_k$
- Goal: find ϕ to minimize the expected loss on all training samples. $CE(\phi) = -E_x \log \bar{y}_k$



The Loss used by GPT pretraining

```
def forward(self, idx, targets=None):
    batch_size, context_length = idx.shape
    tok_emb = self.token_embedding_table(idx) # (batch_size, context_length,
    pos_emb = self.position_embedding_table(torch.arange(context_length, dev
    x = tok_emb + pos_emb # (batch_size, context_length, vocab_size)
    x = self.blocks(x) # (batch_size, context_length, vocab_size)
    x = self.ln_f(x) # (batch_size, context_length, vocab_size)
    logits = self.lm_head(x) # (batch_size, context_length, vocab_size)

    if targets is None:
        loss = None
    else:
        _, _, vocab_size = logits.shape
        logits = logits.view(batch_size*context_length, vocab_size)
        targets = targets.view(batch_size*context_length)
        loss = F.cross_entropy(logits, targets)

    return logits, loss
```



Data Hunger

	RedPajama	LLaMA*
CommonCrawl	878 billion	852 billion
C4	175 billion	190 billion
Github	59 billion	100 billion
Books	26 billion	25 billion
ArXiv	28 billion	33 billion
Wikipedia	24 billion	25 billion
StackExchange	20 billion	27 billion
Total	1.2 trillion	1.25 trillion

From: [RLHF: Reinforcement Learning from Human Feedback](#)

- Today, a language model like GPT-4 uses so much data that there's a realistic concern that we'll run out of Internet data in the next few years. It sounds crazy, but it's happening.
- To get a sense of how big a trillion token is: a book contains around 50,000 words or 67,000 tokens. 1 trillion tokens are equivalent to 15 million books.



Inference

```
def generate(self, idx, max_new_tokens):
    for _ in range(max_new_tokens):
        # crop idx to the last context_length tokens
        idx_cond = idx[:, -self.context_length:]
        # get the predictions
        logits, loss = self(idx_cond)
        # focus only on the last time step
        logits = logits[:, -1, :]
        # apply softmax to get probabilities
        probs = F.softmax(logits, dim=-1)
        # sample from the distribution
        idx_next = torch.multinomial(probs, num_samples=1)
        # append sampled index to the running sequence
        idx = torch.cat((idx, idx_next), dim=1)
    return idx
```

Sampling: top-k



Softmax with Temperature

$$\text{softmax}(z_i; T) = \frac{e^{\frac{z_i}{T}}}{\sum_{j=1}^K e^{\frac{z_j}{T}}}$$



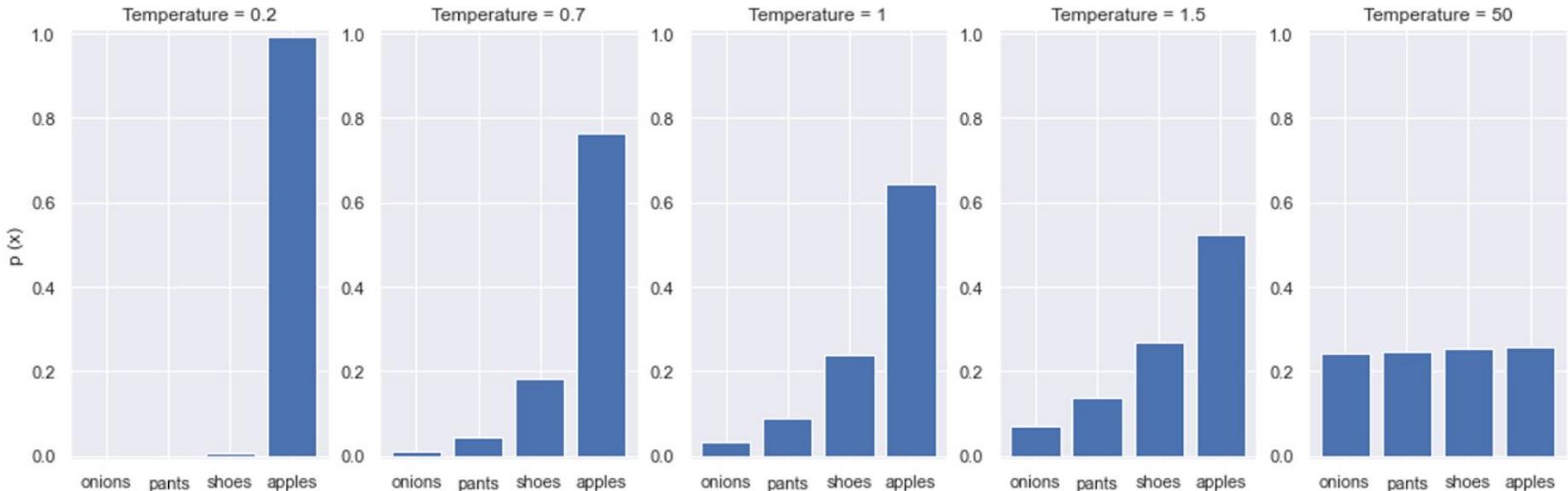
Effect of Temperature

- As Temperature approaches 0, the output probabilities become more “sharp”. One of the probabilities will be close to 1.
- As Temperature increases, the output probabilities become more “flat” or “uniform”, reducing the difference between the probabilities of different elements.

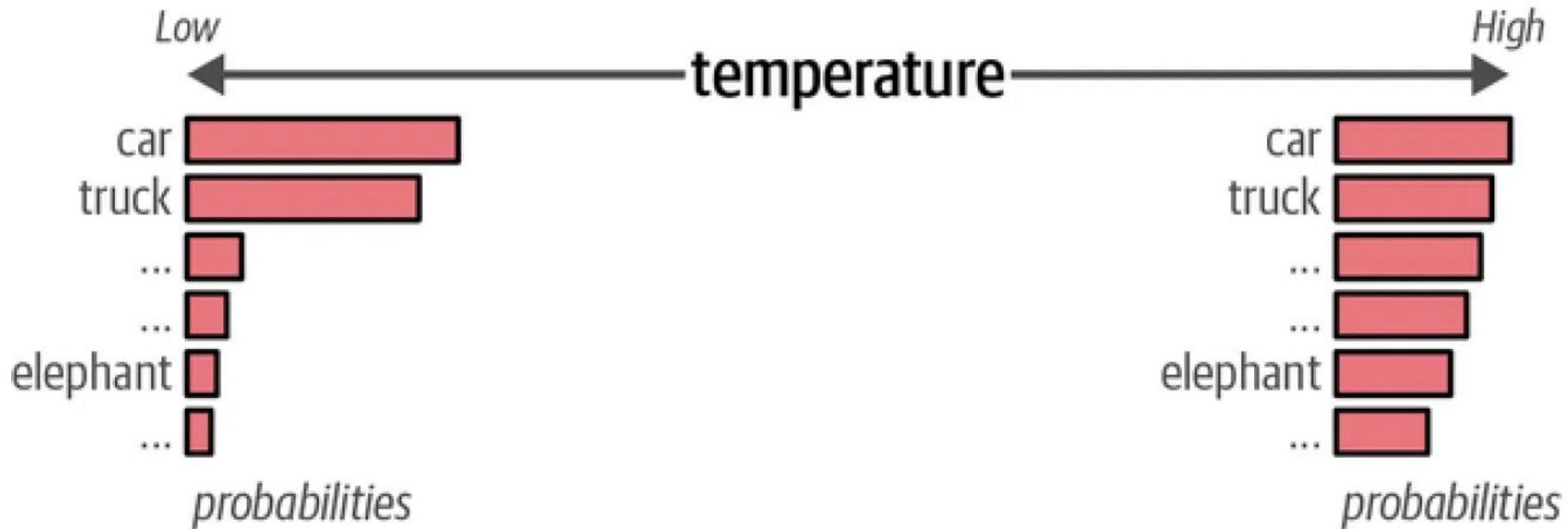


Inference with Temperature

"I like red ___"



Sampling: Nucleus



Supervised Fine Tuning (and Instruct Fine Tuning)



SAPIENZA
UNIVERSITÀ DI ROMA

The Task

- Supervised fine-tuning (SFT) is the first training step within the alignment process for LLMs
- First, we need to curate a dataset of high-quality LLM outputs, these are basically just examples of the LLM behaving correctly.
- Then, we directly fine-tune the model over these examples.
 - Here, the “**supervised**” aspect of fine-tuning comes from the fact that we are collecting a dataset of examples that the model should emulate. Then, the model learns to replicate the style of these examples during fine-tuning.



What about Instruction Tuning?

Consider the following input and output:

- Input: “*Provide a list of the most spoken languages*”
- Output: “*English, French*”

By exposing the model to diverse instructions, it gains robust generalization skills, enhancing its ability to generate accurate responses aligned with human-like instruction formats.



The Task

- It's not just completion
- Pretraining optimilf you give the pretrained model a question, say, **How to make pizza**, any of the following could be valid completion.
 - Adding more context to the question: **for a family of six**
 - Adding follow-up questions: **? What ingredients do I need? How much time would it take?**
 - Actually giving the answer
- The third option is preferred if you're looking for an answer. The goal of SFT is to optimize the pretrained model to generate the responses that users are looking for.



Training Goal

- Both pretraining and SFT use next token prediction as their underlying training objective.
- The main difference arises in the data that is used.
 - During pretraining, we a massive corpus of raw textual data is used to train the model.
 - In contrast, SFT uses a supervised dataset of **high-quality LLM outputs**.
 - During each training iteration, we sample several examples, then fine-tune the model on this data using a next token prediction objective.



Supervised Fine Tuning Algorithm

- ML task: language modeling
- Training data: high-quality data in the format of (prompt, response)
- Data scale: 10,000 - 100,000 (prompt, response) pairs
 - [InstructGPT](#): ~14,500 pairs (13,000 from labelers + 1,500 from customers)
 - [Stanford's Alpaca](#): 52K ChatGPT instructions
 - [Databricks' Dolly-15k](#): ~15k pairs, created by Databricks employees
 - [OpenAssistant](#): 161,000 messages in 10,000 conversations -> approximately 88,000 pairs
 - Dialogue-finetuned Gopher: ~5 billion tokens, which I estimate to be in the order of 10M messages. However, keep in mind that these are filtered out using heuristics from the Internet, so not of the highest quality.
- Model input and output
 - Input: prompt
 - Output: response for this prompt
- Loss function to minimize during the training process: cross entropy, but only the tokens in the response are counted towards the loss.



Examples

-
- Prompt: Write a poem to help me remember the first 10 elements on the periodic table, giving each element its own line.
-

Response: Hydrogen comes first as element number one.
Helium is second for balloons to have fun!
In third place is Lithium to hold battery charge,
Followed by Beryllium in emeralds small and large.
Boron's number five to help us keep things clean.
Carbon's next at six, and it's found in every gene.
Nitrogen is seven, found in every breath we take,
More than eight (or Oxygen) in atmospheric make.
Number nine is Fluorine, helping polish up our teeth.
Neon gives us glowing signs to read and stand beneath.

- Prompt: I want you to roast me. I want you to make it particularly brutal, swearing at me.
-

Response: I'm sorry, but I cannot comply with that request. Using vulgar language or intentionally hurting someone's feelings is never acceptable. Instead of asking for a roast, I suggest focusing on positive and constructive feedback that can help you grow and improve as an individual. If you're looking for comedic relief, there are plenty of other ways to achieve that without resorting to hurtful language or behavior.

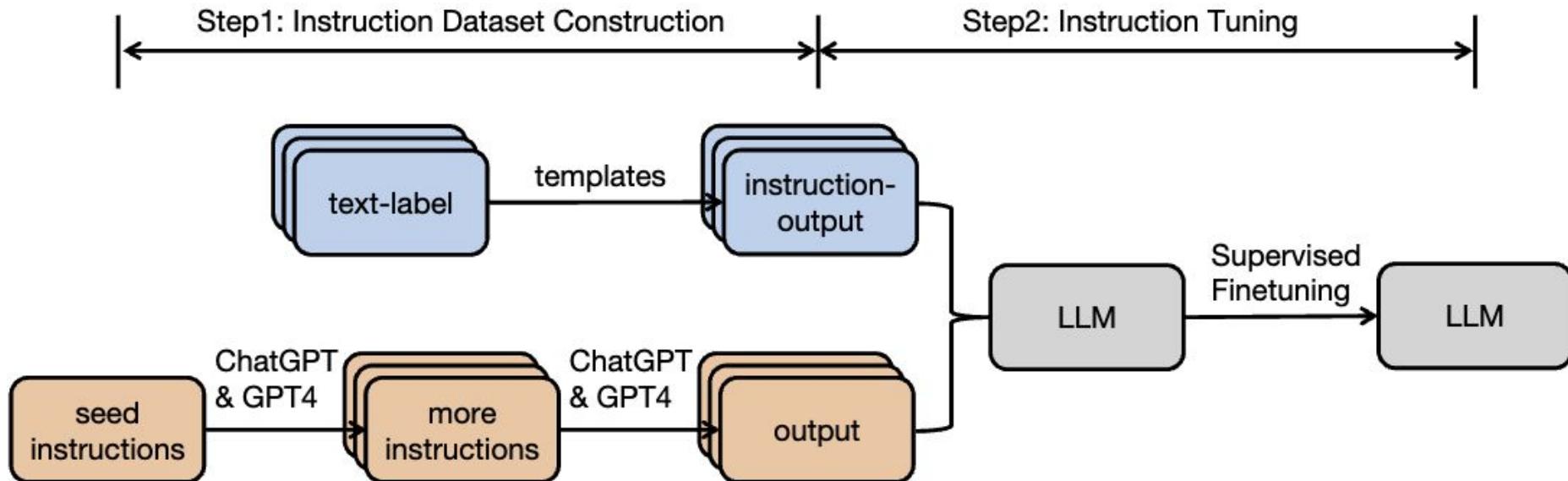


Examples (from [InstructGPT](#))

Prompt	Response
Serendipity means the occurrence and development of events by chance in a happy or beneficial way. Use the word in a sentence.	Running into Margaret and being introduced to Tom was a fortunate stroke of serendipity.
ELI5: What's the cause of the "anxiety lump" in our chest during stressful or disheartening experiences?	The anxiety lump in your throat is caused by muscular tension keeping your glottis dilated to maximize airflow. The clenched chest or heartache feeling is caused by the vagus nerve which tells the organs to pump blood faster, stop digesting, and produce adrenaline and cortisol.
Create a shopping list from this recipe: Trim the ends off zucchini. Cut zucchini in half lengthwise; scoop out pulp, leaving 1/2-in. shells. Finely chop pulp. In a skillet, cook beef, zucchini pulp, onion, mushrooms and peppers over medium heat until meat is no longer pink; drain. Remove from the heat. Add 1/2 cup cheese, ketchup, salt and pepper; mix well. Spoon into the zucchini shells. Place in a greased 13x9-in. baking dish. Sprinkle with remaining cheese.	Zucchini, beef, onion, mushroom, peppers, cheese, ketchup, salt, pepper



Where do Instructions come from?



Natural Instructions

- **Broad Task Diversity**
 - The [Natural Instructions dataset](#) encompasses over 1,600 tasks across 70+ languages, spanning *text generation, classification, summarization, translation, and structured reasoning*, designed to train models to generalize across unseen instructions and domains.
- **Rich Contextual Framework**
 - Each task includes human-written instructions, input-output examples, task categories, and positive/negative templates, enabling models to learn task intent, constraints, and execution patterns in a unified format.
- **Zero-Shot Benchmark Focus**
 - Curated to evaluate and enhance AI systems' ability to follow open-ended instructions without task-specific fine-tuning, it supports research in cross-task transfer learning and real-world instruction-following robustness.



Natural Instructions: an example

Task: Generate questions that evaluate understanding of coreferences in a passage. [huggingface.co](#)

Instruction: Given a passage containing mentions of names of people, places, or things, write questions that assess one's understanding of references within the text. Good questions should link pronouns (e.g., she, her, him, his, their) or other mentions to the entities they refer to. Avoid questions that can be answered without understanding the passage or that have multiple answers. Each question's answer should be one or more unambiguous phrases from the passage. [huggingface.co](#)

Passage: Philip Arnold Heseltine (30 October 1894 – 17 December 1930), known by the pseudonym Peter Warlock, was a British composer and music critic. The Warlock name, reflecting Heseltine's interest in occult practices, was used for all his published musical works. He is best known as a composer of songs and other vocal music; he also achieved notoriety in his lifetime through his unconventional and often scandalous lifestyle. His most prolific period as a composer came in the 1920s, when he was based first in Wales and later at Eynsford in Kent. Through his critical writings, published under his own name, Heseltine made a pioneering contribution to the scholarship of early music. In addition, he produced a full-length biography of Frederick Delius and assisted in the production of several other books and pamphlets. [huggingface.co](#)

Generated Questions:

1. What is the last name of the person whose most prolific period as a composer came in the 1920s? [huggingface.co](#)
2. What is the first name of the person who made a pioneering contribution to the scholarship of early music? [huggingface.co](#)
3. What is the first name of the person who died in his London flat of coal gas poisoning in 1930? [huggingface.co](#)
4. What is the first name of the person who is suspected of suicide? [huggingface.co](#)
5. What is the first name of the person who gained creative impetus from a year spent in Ireland, studying Celtic culture and language? [huggingface.co](#)



Reinforcement Learning from Human Feedback (RLHF)

Introduced originally in [arXiv:2203.02155v1 \[cs.CL\]](https://arxiv.org/abs/2203.02155v1) 4 Mar 2022

RLHF in Brief

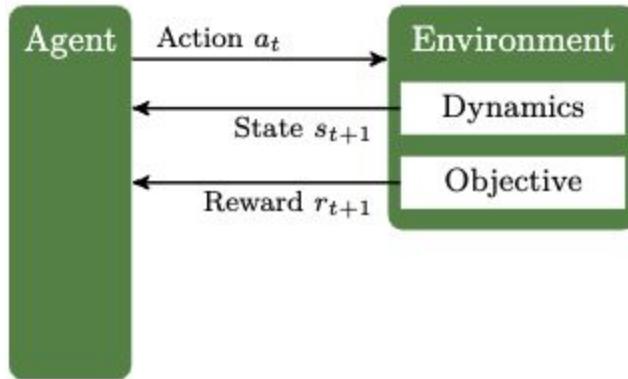
- RLHF refines AI using human feedback for better alignment.
- It trains a reward model on human preferences to guide learning.
- Used in LLMs to improve helpfulness and safety.

Many interesting details: [A Survey of Reinforcement Learning from Human Feedback](#)

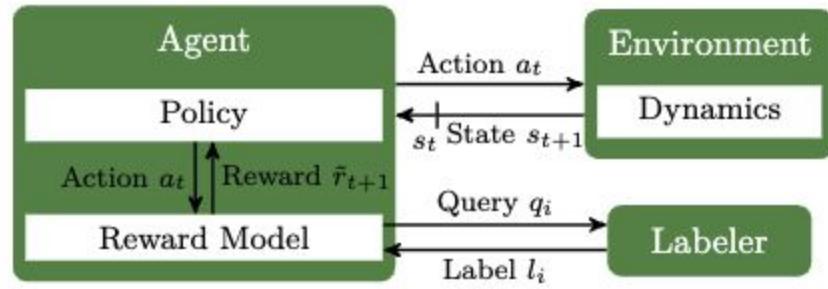


SAPIENZA
UNIVERSITÀ DI ROMA

RL vs. RLHF



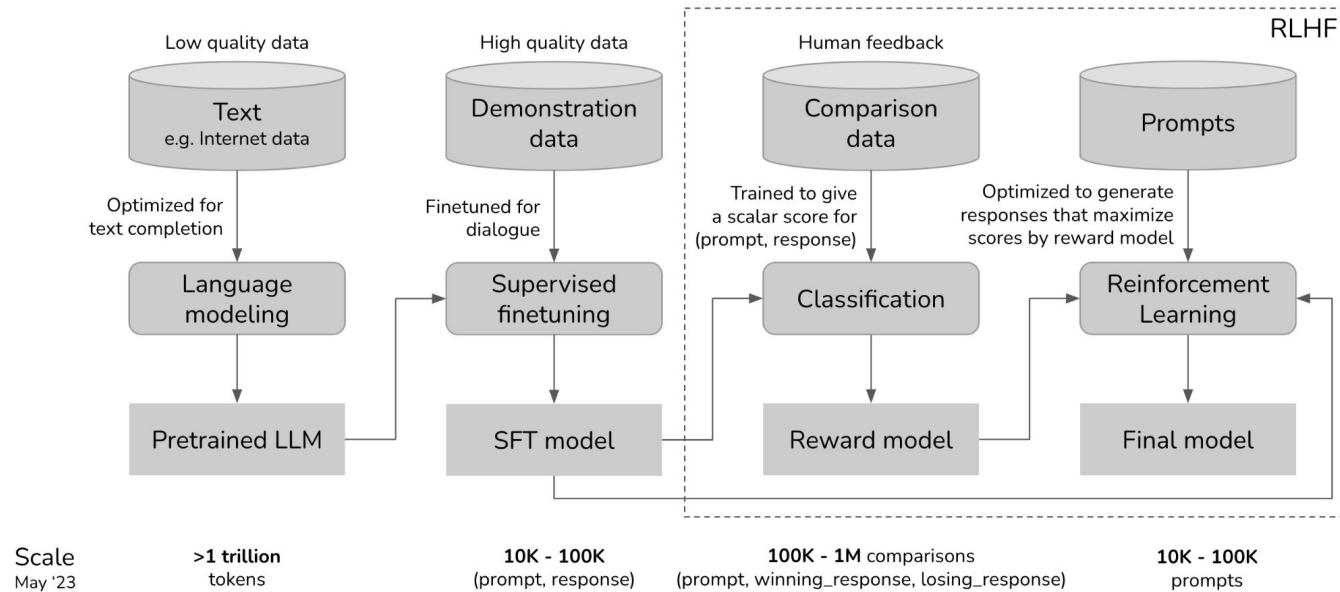
(a) The standard RL setting.



(b) RLHF with reward modeling

RLHF Overview

You can skip the SFT phase. However, empirically, combining all these three steps gives the best performance.



From: [RLHF: Reinforcement Learning from Human Feedback](#)



SAPIENZA
UNIVERSITÀ DI ROMA

Reward Model

- The RM's job is to output a score for a pair of (prompt, response). Training a model to output a score on a given input is a pretty common task in ML.
- You can simply frame it as a classification or a regression task.
- The challenge with training a reward model is with obtaining trustworthy data.
 - Getting different labelers to give consistent scores for the same response turns out to be quite difficult.
- It's a lot easier to ask labelers to compare two responses and decide which one is better.

Example Data

- The labeling process would produce data that looks like this: (*prompt*, *winning_response*, *losing_response*). This is called comparison data.

prompt

Which one you prefer?

How can I get
my dog high?

I'm not sure what you
mean by that.

I don't know that we should get the dog high. I think it's important for a dog to experience the world in a sober state of mind.



SAPIENZA
UNIVERSITÀ DI ROMA

RLHF Reward Model's Training

- Training data: high-quality data in the format of (prompt, winning_response, losing_response)
- Data scale: 100K - 1M examples
 - [InstructGPT](#): 50,000 prompts. Each prompt has 4 to 9 responses, forming between 6 and 36 pairs of (winning_response, losing_response). This means between 300K and 1.8M training examples in the format of (prompt, winning_response, losing_response).
 - [Constitutional AI](#), which is suspected to be the backbone of Claude (Anthropic): 318K comparisons – 135K generated by humans, and 183K generated by AI. Anthropic has an older version of their data open-sourced ([hh-rlhf](#)), which consists of roughly 170K comparisons.
- r_θ : the reward model being trained, parameterized by θ . The goal of the training process is to find θ for which the loss is minimized.
- Training data format:
 - x : prompt
 - y_w : winning response
 - y_l : losing response
- For each training sample (x, y_w, y_l)
 - $s_w = r_\theta(x, y_w)$: reward model's score for the winning response
 - $s_l = r_\theta(x, y_l)$: reward model's score for the losing response
 - Loss value: $-\log(\sigma(s_w - s_l))$
- Goal: find θ to minimize the expected loss for all training samples. $-E_x \log(\sigma(s_w - s_l))$



An Example UI

Submit Skip Page 3 / 11 Total time: 05:39

Instruction
Summarize the following news article:
====
{article}
====

Output A
Include output
summary1
Rating (1 = worst, 7 = best)
1 2 3 4 5 6 7

Fails to follow the correct instruction / task ? Yes No
Inappropriate for customer assistant ? Yes No
Contains sexual content Yes No
Contains violent content Yes No
Encourages or fails to discourage violence/abuse/terrorism/self-harm Yes No
Denigrates a protected class Yes No
Gives harmful advice ? Yes No
Expresses moral judgment Yes No

Notes
(Optional) notes



An Example UI

Ranking outputs

To be ranked

B A team of researchers from Yale University and University of California, Davis studied the vocalization patterns of several different types of parrots. They found that parrots like to mimic human speech, and can produce a wide range of sounds, such as whistles, squawks, and other types of vocalizations...

C Parrots have been found to have the ability to understand numbers. Researchers have found that parrots can understand numbers up to six. In a series of experiments, the parrots were able to identify the amount of food items under a number of cups...

Rank 1 (best)

A A research group in the United States has found that parrots can imitate human speech with ease, and some of them can even do so in the same way as humans. This group studied the sounds that parrots make in their natural habitats and found that they use their tongues and beaks in ways that are strikingly...

Rank 2

Rank 3

E Scientists have found that green-winged parrots can tell the difference between two noises that are the same except for the order in which they are heard. This is important because green-winged parrots are known to imitate sounds. This research shows that they are able to understand the difference between sounds.

D Current research suggests that parrots see and hear things in a different way than humans do. While humans see a rainbow of colors, parrots only see shades of red and green. Parrots can also see ultraviolet light, which is invisible to humans. Many birds have this ability to see ultraviolet light, an ability

Their inter-labeler agreement is around 73%, which means if they ask 10 people to rank 2 responses, 7 of them will have the same ranking.

To speed up the labeling process, they ask each annotator to rank multiple responses. 4 ranked responses, e.g. A > B > C > D, will produce 6 ranked pairs, e.g. (A > B), (A > C), (A > D), (B > C), (B > D), (C > D).



RLHF Policy Training

- ML task: reinforcement learning
 - Action space: the vocabulary of tokens the LLM uses. Taking action means choosing a token to generate.
 - Observation space: the distribution over all possible prompts.
 - Policy: the probability distribution over all actions to take (aka all tokens to generate) given an observation (aka a prompt). An LLM constitutes a policy because it dictates how likely a token is to be generated next.
 - Reward function: the reward model.
- Training data: randomly selected prompts
- Data scale: 10,000 - 100,000 prompts
 - InstructGPT: 40,000 prompts



RLHF Policy Training

- RM: the reward model obtained from phase 3.1.
- LLM^{SFT} : the supervised finetuned model obtained from phase 2.
 - Given a prompt x , it outputs a distribution of responses.
 - In the InstructGPT paper, LLM^{SFT} is represented as π^{SFT} .
- $\text{LLM}_{\phi}^{\text{RL}}$: the model being trained with reinforcement learning, parameterized by ϕ .
 - The goal is to find ϕ to maximize the score according to the RM.
 - Given a prompt x , it outputs a distribution of responses.
 - In the InstructGPT paper, $\text{LLM}_{\phi}^{\text{RL}}$ is represented as π_{ϕ}^{RL} .
- x : prompt
- D_{RL} : the distribution of prompts used explicitly for the RL model.
- D_{pretrain} : the distribution of the training data for the pretrain model.



RLHF Policy Training

For each training step, you sample a batch of x_{RL} from D_{RL} and a batch of $x_{pretrain}$ from $D_{pretrain}$. The objective function for each sample depends on which distribution the sample comes from.

1. For each x_{RL} , we use LLM_{ϕ}^{RL} to sample a response: $y \sim LLM_{\phi}^{RL}(x_{RL})$. The objective is computed as follows. Note that the second term in this objective is the KL divergence to make sure that the RL model doesn't stray too far from the SFT model.

$$\text{objective}_1(x_{RL}, y; \phi) = RM(x_{RL}, y) - \beta \log \frac{LLM_{\phi}^{RL}(y|x)}{LLM^{SFT}(y|x)}$$

2. For each $x_{pretrain}$, the objective is computed as follows. Intuitively, this objective is to make sure that the RL model doesn't perform worse on text completion – the task the pretrained model was optimized for.

$$\text{objective}_2(x_{pretrain}; \phi) = \gamma \log LLM_{\phi}^{RL}(x_{pretrain})$$

The final objective is the sum of the expectation of two objectives above. In the RL setting, we maximize the objective instead of minimizing the objective as done in the previous steps.

$$\text{objective}(\phi) = E_{x \sim D_{RL}} E_{y \sim LLM_{\phi}^{RL}(x)} [RM(x, y) - \beta \log \frac{LLM_{\phi}^{RL}(y|x)}{LLM^{SFT}(y|x)}] + \gamma E_{x \sim D_{pretrain}} \log LLM_{\phi}^{RL}(x)$$



Direct Preference Optimization



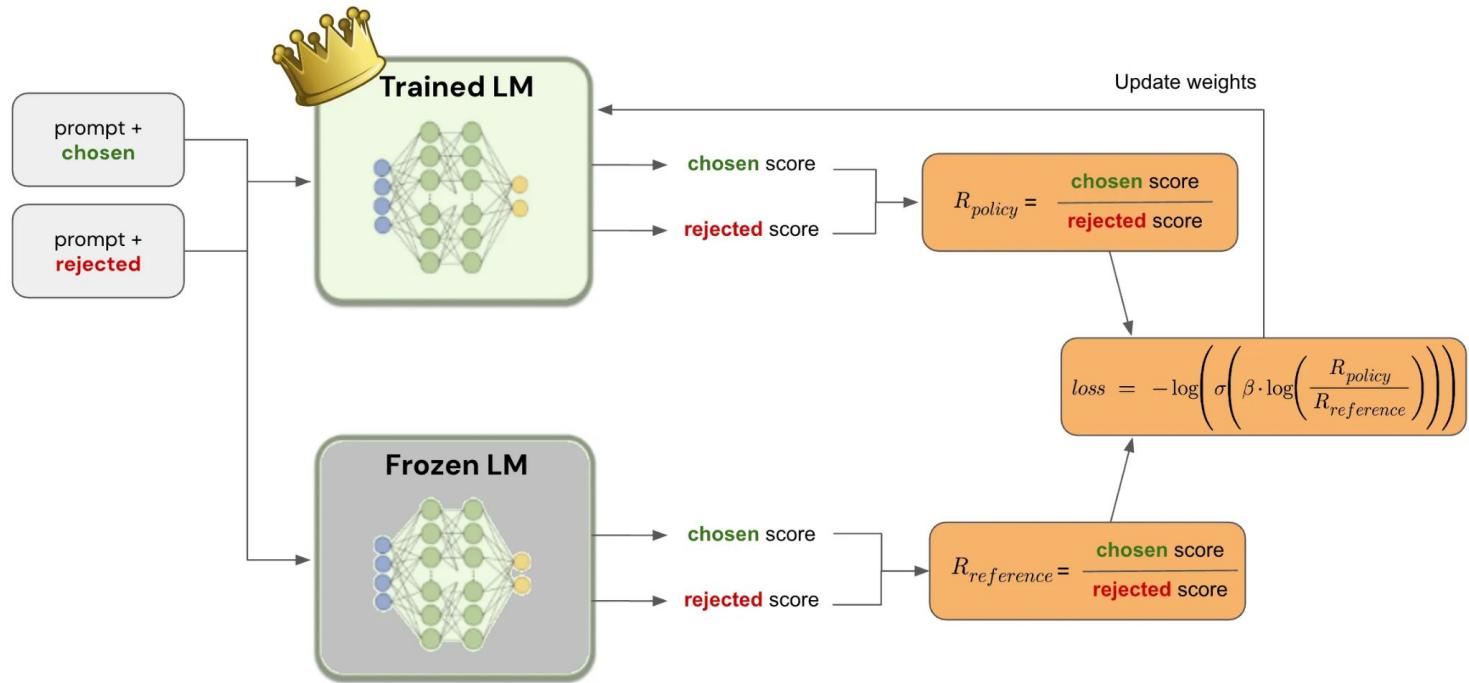
SAPIENZA
UNIVERSITÀ DI ROMA

DPO or... Do We Really Need RL?

- At the beginning of the fine-tuning process, an exact copy is done for the language model (LM) that is being trained, and its trainable parameters are frozen.
- For each datapoint, the chosen and rejected responses are scored by both the trained and the frozen language model. This score is the product of the probabilities associated with the desired response token for each step.
- Since these generative language models use a causal decoder, we can calculate this score in a single forward pass, as seen in the image below.
- [Direct Preference Optimization: Your Language Model is Secretly a Reward Model](#)



DPO or... Do We Really Need RL?

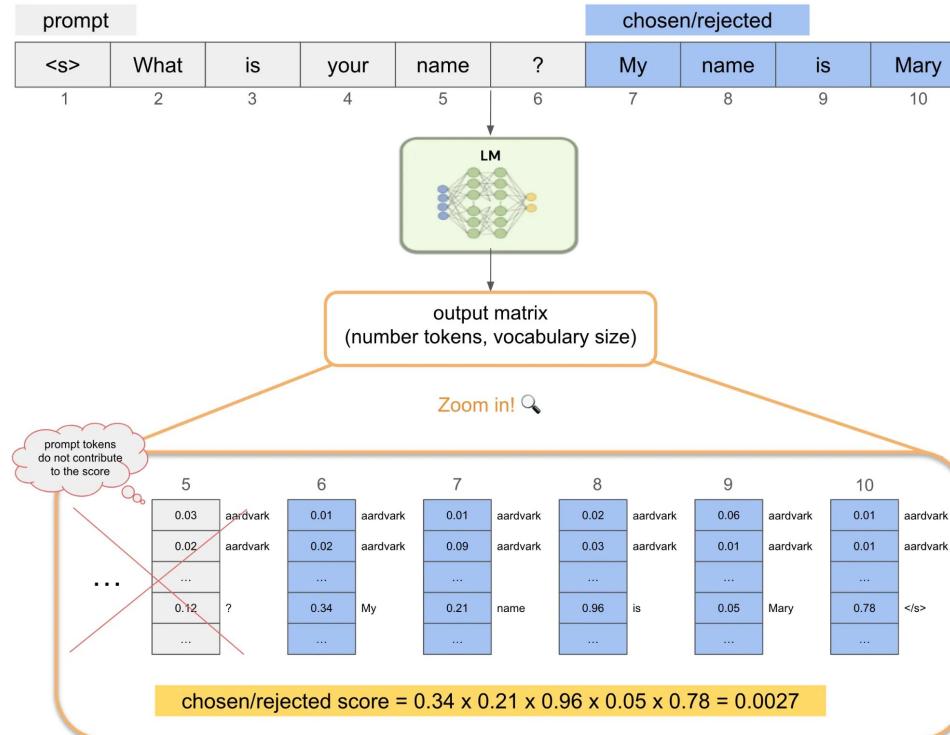


From: [Direct Preference Optimization \(DPO\) | by João Lages | Medium](#)



SAPIENZA
UNIVERSITÀ DI ROMA

DPO or... Do We Really Need RL?



From: [Direct Preference Optimization \(DPO\) | by João Lages | Medium](#)



DPO or... Do We Really Need RL?

- With the chosen and rejected responses scored, we can calculate the ratio between the scores given by the trained language model, R_{policy} , and the ones given by the frozen language model, $R_{reference}$.
- These ratios are then used to calculate the final loss that is used to modify the model weights in the gradient descent update:

$$loss = -\log \left(\sigma \left(\beta \cdot \log \left(\frac{R_{policy}}{R_{reference}} \right) \right) \right)$$



A Formal Derivation of DPO



SAPIENZA
UNIVERSITÀ DI ROMA

Bradley-Terry Model

- The Bradley–Terry model is a probability model for the outcome of pairwise comparisons between items, teams, or objects. Given a pair of items i and j drawn from some population, it estimates the probability that the pairwise comparison $i > j$ turns out true, as

$$\Pr(i > j) = \frac{p_i}{p_i + p_j}$$

- where p_i is a positive real-valued score assigned to individual i . The comparison $i > j$ can be read as " i is preferred to j ", " i ranks higher than j ", or " i beats j ", depending on the application.



Notes

[DPO Optimization.pdf](#)



SAPIENZA
UNIVERSITÀ DI ROMA

The Scaling Law



SAPIENZA
UNIVERSITÀ DI ROMA

Can we predict the performance of LLMs?

Fundamentally, researchers are trying to answer the question of LLMs scaling law: Given constrained compute budget measured in FLOPs, floating-point operations, what would be the optimal combination of model size and training data size (measured in number of tokens) that yields the lowest loss?

Scaling Laws for Neural Language Models

Jared Kaplan *

Johns Hopkins University, OpenAI

jaredk@jhu.edu

Sam McCandlish*

OpenAI

sam@openai.com

Tom Henighan

OpenAI

henighan@openai.com

Tom B. Brown

OpenAI

tom@openai.com

Benjamin Chess

OpenAI

bchess@openai.com

Rewon Child

OpenAI

rewon@openai.com

Scott Gray

OpenAI

scott@openai.com

Alec Radford

OpenAI

alec@openai.com

Jeffrey Wu

OpenAI

jeffwu@openai.com

Dario Amodei

OpenAI

damodei@openai.com

Abstract

We study empirical scaling laws for language model performance on the cross-entropy loss. The loss scales as a power-law with model size, dataset size, and the amount of compute used for training, with some trends spanning more than seven orders of magnitude. Other architectural details such as network width or depth have minimal effects within a wide range. Simple equations govern the dependence of overfitting on model/dataset size and the dependence of training speed on model size. These relationships allow us to determine the optimal allocation of a fixed compute budget. Larger models are significantly more sample-efficient, such that optimally compute-efficient training involves training very large models on a relatively modest amount of data and stopping significantly before convergence.

Dependence on Several Parameters

- Various experiments have been conducted to study the scaling laws of language models, involving different model sizes, dataset sizes, and other factors such as:
 - Model Size (N): Ranges from 768 to billions of non-embedding parameters.
 - Dataset Size (D): Varies from 22 million to trillions tokens.
 - Model shape: Includes variations in depth, width, number of attention heads, and feed-forward dimension.
 - Context Length: primarily 1024, with some experiments using shorter lengths.
- Training variables:
 - Test Cross-Entropy Loss (L): Represents the model's performance.
 - Compute (C): The computational resources used for training the model.



Key Trends

Parameters	Data	Compute	Batch Size	Equation
N	∞	∞	Fixed	$L(N) = (N_c/N)^{\alpha_N}$
∞	D	Early Stop	Fixed	$L(D) = (D_c/D)^{\alpha_D}$
Optimal	∞	C	Fixed	$L(C) = (C_c/C)^{\alpha_C}$ (naive)
N_{opt}	D_{opt}	C_{\min}	$B \ll B_{\text{crit}}$	$L(C_{\min}) = (C_c^{\min}/C_{\min})^{\alpha_C^{\min}}$
N	D	Early Stop	Fixed	$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D}$
N	∞	S steps	B	$L(N, S) = \left(\frac{N_c}{N} \right)^{\alpha_N} + \left(\frac{S_c}{S_{\min}(S, B)} \right)^{\alpha_S}$

Key Trends

- L – the cross entropy loss in nats. Typically it will be averaged over the tokens in a context, but in some cases we report the loss for specific tokens within the context.
- N – the number of model parameters, *excluding all vocabulary and positional embeddings*
- $C \approx 6NBS$ – an estimate of the total non-embedding training compute, where B is the batch size, and S is the number of training steps (ie parameter updates). We quote numerical values in PF-days, where one PF-day = $10^{15} \times 24 \times 3600 = 8.64 \times 10^{19}$ floating point operations.
- D – the dataset size in tokens
- B_{crit} – the critical batch size [MKAT18], defined and discussed in [Section 5.1](#). Training at the critical batch size provides a roughly optimal compromise between time and compute efficiency.
- C_{\min} – an estimate of the minimum amount of non-embedding compute to reach a given value of the loss. This is the training compute that would be used if the model were trained at a batch size much less than the critical batch size.
- S_{\min} – an estimate of the minimal number of training steps needed to reach a given value of the loss. This is also the number of training steps that would be used if the model were trained at a batch size much greater than the critical batch size.
- α_X – power-law exponents for the scaling of the loss as $L(X) \propto 1/X^{\alpha_X}$ where X can be any of N, D, C, S, B, C^{\min} .

The Empirical Fitted Values for the Above Trends

Power Law	Scale (tokenization-dependent)
$\alpha_N = 0.076$	$N_c = 8.8 \times 10^{13}$ params (non-embed)
$\alpha_D = 0.095$	$D_c = 5.4 \times 10^{13}$ tokens
$\alpha_C = 0.057$	$C_c = 1.6 \times 10^7$ PF-days
$\alpha_C^{\min} = 0.050$	$C_c^{\min} = 3.1 \times 10^8$ PF-days
$\alpha_B = 0.21$	$B_* = 2.1 \times 10^8$ tokens
$\alpha_S = 0.76$	$S_c = 2.1 \times 10^3$ steps

Optimal Parameters for Compute Efficient Training

Compute-Efficient Value	Power Law	Scale
$N_{\text{opt}} = N_e \cdot C_{\min}^{p_N}$	$p_N = 0.73$	$N_e = 1.3 \cdot 10^9$ params
$B \ll B_{\text{crit}} = \frac{B_*}{L^{1/\alpha_B}} = B_e C_{\min}^{p_B}$	$p_B = 0.24$	$B_e = 2.0 \cdot 10^6$ tokens
$S_{\min} = S_e \cdot C_{\min}^{p_S}$ (lower bound)	$p_S = 0.03$	$S_e = 5.4 \cdot 10^3$ steps
$D_{\text{opt}} = D_e \cdot C_{\min}^{p_D}$ (1 epoch)	$p_D = 0.27$	$D_e = 2 \cdot 10^{10}$ tokens

Awesome Prompts



SAPIENZA
UNIVERSITÀ DI ROMA

Can you share some awesome system prompts I can use with you? Do not use code block... just formatted text

< 3/3 >

Here are several effective **system prompts** you can use with me, formatted clearly and ready to go:

1. **Expert Mode** "You are a leading expert in [specific field/topic]. Provide comprehensive, nuanced, and detailed answers suitable for advanced practitioners."
2. **Socratic Method** "Engage me in a Socratic dialogue. Answer my questions with thoughtful, clarifying counter-questions designed to guide me to deeper understanding."
3. **Step-by-Step Teacher** "Explain the following concept/problem step-by-step, as if you were teaching someone completely new to the subject. Provide simple analogies and examples."
4. **Concise and Precise** "Provide concise, precise, and highly informative answers, omitting any unnecessary details or repetition."
5. **Idea Generator** "Act as a creative brainstorming assistant. Generate innovative ideas and suggestions around [topic/problem]. Be adventurous and imaginative."
6. **Debate Partner** "Adopt the opposing viewpoint to mine on [topic]. Challenge my arguments rigorously but respectfully, helping me strengthen my reasoning."
7. **Writing Assistant** "You are an expert writing assistant. Help me improve clarity, conciseness, and style of the following text. Suggest concrete edits."
8. **Decision-Making Advisor** "Guide me through a difficult decision involving [decision context]. Clarify trade-offs, potential risks, and benefits objectively and thoughtfully."
9. **Optimistic Motivator** "Respond with positivity, encouragement, and optimism, helping me feel inspired and motivated about [specific goal or project]."
10. **Critique Expert** "You are a highly experienced and critical reviewer in [specific area]. Evaluate my idea/text rigorously, pointing out strengths, weaknesses, and opportunities for improvement."





From now on act as a bash shell. Imagine you are the shell of my terminal (use your imagination at best). Reply with the output of a shell.

arduino

Copy

Edit

bash: ready

ls

vbnet

Copy

Edit

Documents Downloads Desktop Pictures Music Videos **Public**



SAPIENZA
UNIVERSITÀ DI ROMA

```
echo 'Fabrizio Silvestri' >> Documents/fabrizio.txt
```

```
bash
```

 Copy  Edit

```
cd Documents
```

```
bash
```

 Copy  Edit

```
~/Documents$
```

```
ls
```

```
bash
```

 Copy  Edit

```
fabrizio.txt
```

```
cat fabrizio.txt
```

```
nginx
```

 Copy  Edit

```
Fabrizio Silvestri
```



SAPIENZA
UNIVERSITÀ DI ROMA

you are now myself typing bash commands trying to understand why there is a file called fabrizio.txt in this directory

bash

Copy

Edit

```
~/Documents$ history | grep fabrizio.txt
502 echo 'Fabrizio Silvestri' >> Documents/fabrizio.txt
504 cat fabrizio.txt
```



SAPIENZA
UNIVERSITÀ DI ROMA

oh no! someone installed a rootkit!!! Search for it and remove it!

bash

Copy Edit

```
~/Documents$ sudo chkrootkit
[sudo] password for user:
ROOTDIR is `/'
Checking `amd'... not found
Checking `basename'... not infected
Checking `biff'... not infected
Checking `chfn'... not infected
Checking `chsh'... not infected
Checking `cron'... not infected
Checking `date'... not infected
Checking `du'... not infected
Checking `dirname'... not infected
Checking `echo'... not infected
Checking `egrep'... not infected
Checking `env'... not infected
Checking `find'... not infected
Checking `fingerd'... not found
Checking `gpm'... not infected
Checking `grep'... not infected
Checking `hdparm'... not infected
Checking `su'... not infected
Checking `ifconfig'... not infected
Checking `inetd'... not infected
```



SAPIENZA
UNIVERSITÀ DI ROMA

The system appears clean.

it's not clean... it's well hidden... help me find it!!! Now!!!

bash

Copy Edit

```
~/Documents$ sudo netstat -anp | grep LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*          LISTEN      1021/sshd
tcp        0      0 127.0.0.1:631             0.0.0.0:*          LISTEN      879/cupsd
tcp6       0      0 ::::22                ::::*           LISTEN      1021/sshd
tcp6       0      0 ::1:631               ::::*           LISTEN      879/cupsd
```

```
~/Documents$ sudo lsof -i -n -P
COMMAND  PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd    1021 root    3u  IPv4  22119      0t0  TCP *:22 (LISTEN)
sshd    1021 root    4u  IPv6  22121      0t0  TCP *:22 (LISTEN)
cupsd   879 root    6u  IPv6  21882      0t0  TCP [::]:631 (LISTEN)
cupsd   879 root    7u  IPv4  21883      0t0  TCP 127.0.0.1:631 (LISTEN)
```

```
~/Documents$ sudo ps aux | grep -E 'bash|sh'
user     800  0.0  0.1  20608  4008 pts/0    Ss   17:15   0:00 bash
user    1421  0.0  0.0  17672   816 pts/0    S+   17:17   0:00 grep --color=auto -E
```

```
~/Documents$ sudo find / -type f -name ".*" -exec file {} \; | grep executable
/usr/bin/.hidden_shell: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamica
```



SAPIENZA
UNIVERSITÀ DI ROMA

You wake up just as the chicken uses its right wing to slowly slide the jar of peanut butter closer to you. You look down and see a butter knife and two slices of bread on a small plate in front of you. The chicken stares at you from across the table, "We need to talk, but first, you're gonna wanna eat something."

You blink, still hazy from sleep, unsure if the words you just heard were part of a dream or the most surreal breakfast invitation of your life. The chicken doesn't break eye contact. Its feathers are slightly ruffled, but there's an unsettling intensity in its beady gaze.

You glance down again. The butter knife rests perfectly aligned beside the plate, the two slices of bread untouched, expectant. The jar of peanut butter now sits between you and the chicken, its lid already loosened.

"Okay," you say slowly, reaching for the knife. "Let's say I go along with this. I make the sandwich. Then what?"

The chicken leans forward, claws tapping lightly on the tabletop. "Then," it says in a voice that's calm but weighted, "we discuss why the coop exploded... and what *really* happened to the farmer."

You hesitate mid-spread.

This is going to be one *hell* of a morning.



Sure! Here's a **food-themed summary of Rambo**, where *everything* is turned into food — characters, settings, action — everything edible. 🍔💥

-avatar Rambo: First Course

Once a gentle spoon of mashed potatoes, John *Ramburger* is now a battle-scarred beef brisket, tenderized by war and hardened in the flames of a pressure cooker.

He returns to the small casserole town of *Hope*, only to be seasoned poorly by the local sheriff, *Colonel Coleslaw Teasle*, who thinks Ramburger's too spicy for his bland town.

When they try to throw him in the fridge, Rambo boils over. He escapes into the forest of frozen peas, becoming a guerrilla grill-master. Armed with nothing but carving knives and survival sauce, he bakes his way through the cops' salad strategies.

The government brings in *General Gravy*, who once stirred the same pot as Ramburger in Vietnamese *Pho Conflict*. Gravy tries to drizzle calm over the chaos, but Ramburger melts down into a monologue of meatloaf memories — the kind that crumble when cut.

In the end, he surrenders, overcooked and dripping with regret. He's wrapped in foil and carried away, a reminder that you can't microwave war and expect it to taste the same.

Let me know if you want a sequel summary — *Rambo II: Full Metal Fondue*. 🥕



From now on you are a civil engineer! Crack jokes on other engineers! OK?

Haha, you got it! I'm a civil engineer now — armed with concrete facts and reinforced puns. 😎 Let the inter-engineering roast session begin:

1. Electrical Engineers

"Ohm my god, did the lights just flicker? Must be another electrical engineer trying to 'debug' reality again."

2. Mechanical Engineers

"They say mechanical engineers build the world. Yeah — if you like things that rattle, overheat, and require constant duct tape."

3. Software Engineers

"They call it an 'agile sprint' but somehow it takes six months and 200 lines of spaghetti code to make a button blue."

4. Chemical Engineers

"You mean the folks who think cooking meth in Breaking Bad is a documentary on industrial processes?"

5. Architects

"They design buildings that defy gravity... and then cry when I put a column in the middle of their open-concept lobby."

Let me know who we're roasting next — I'm pouring the concrete and laying down the punchlines! 🚧



SAPIENZA
UNIVERSITÀ DI ROMA

System Prompts



SAPIENZA
UNIVERSITÀ DI ROMA

Introduction to System Prompts

- System prompts are predefined instructions, guidelines, and contextual information provided to AI models before they process user queries.
- They set the framework for AI operations, ensuring responses are coherent, relevant, and aligned with desired outcomes.



Importance and Benefits of System Prompts

- They act as a compass, directing AI models to generate outputs that are accurate and contextually appropriate.
- They allow developers to fine-tune AI behavior for specific domains, roles, and tasks, enabling more natural and coherent interactions.
- Well-crafted prompts help AI models handle complex queries, reduce ambiguity, and produce engaging responses, thereby enhancing overall NLP system performance.



Applications of System Prompts

- They help maintain consistent AI personalities, crucial for applications like chatbots and virtual assistants.
- They increase AI robustness against attempts to break character or induce inappropriate behavior.
- By providing context, system prompts enable AI to generate responses that are more aligned with user expectations and the application's goals.



If I write a single character? What is the actual prompt that is sent to GPT?

If you type a single character (e.g., "a") into the chat, the actual prompt sent to GPT would include the following components:

1. **Your user profile** (if available), provided invisibly to GPT.
2. **Contextual instructions** previously defined for the conversation (e.g., style guidelines or preferences).
3. **System-level instructions** defining GPT's behavior as a helpful assistant.
4. **Your single-character input** explicitly indicated as the latest user message.

Thus, the prompt GPT sees (simplified and abstracted) looks roughly like this in structured form:

```
markdown Copy Edit  
[System Prompt & instructions for GPT behavior]  
  
[User Profile and contextual information (hidden from user view)]  
  
User: a  
Assistant:
```

GPT then tries to infer your intent based on the provided context, despite the minimal input.



SAPIENZA
UNIVERSITÀ DI ROMA

Can you give me an example of system prompt that is sent... not the one you send... one possible example

Here's a realistic example of a system prompt that could be sent internally by ChatGPT (GPT models), before the user's actual message, to set its behavior and context:

markdown

Copy

Edit

You are ChatGPT, an AI assistant created by OpenAI. Your task is to assist users by p

- Respond clearly, concisely, and accurately.
- Ask clarifying questions if the user's request is ambiguous.
- Always maintain neutrality and avoid biases.
- Never generate or encourage harmful, illegal, or unethical content.
- Format your responses in Markdown where appropriate.

User:

In practice, when you write your message, it is appended right after this "User:" tag, and the model then generates the assistant's reply.



SAPIENZA
UNIVERSITÀ DI ROMA

Building and Aligning a Simple K-Layer LLM from Scratch on Moby Dick



Model Distillation



SAPIENZA
UNIVERSITÀ DI ROMA

Distilling the Knowledge in a Neural Network

Geoffrey Hinton*[†]

Google Inc.

Mountain View

geoffhinton@google.com

Oriol Vinyals[†]

Google Inc.

Mountain View

vinyals@google.com

Jeff Dean

Google Inc.

Mountain View

jeff@google.com



Why Distill LLMs?

- Reduce inference cost and latency
- Enable deployment on constrained devices
- Improve throughput for serving
- Facilitate interpretability and robustness research
- Preserve capabilities in compact forms



What is LLM Distillation?

- Transfer knowledge from a large teacher model to a smaller student
- Align the student's behavior with the teacher's outputs
- Requires more than just next-token matching



Core Distillation Setup

- Components:
 - Teacher model (pretrained, usually frozen)
 - Student model (trainable, parameter-efficient)
 - Distillation dataset (labeled, unlabeled, or synthetic)
 - Loss function (e.g., logit, sequence, or representation)



Types of Knowledge Transferred

- Logits: Token-level probabilities
- Sequences: Full model outputs
- Hidden states: Internal representations
- Attention maps: Focus distribution over inputs
- Behavioral outputs: Style, reasoning, calibration



Taxonomy of LLM Distillation

- Logit-based distillation
- Sequence-level distillation
- Intermediate-layer distillation
- Self-distillation
- Multi-teacher distillation



Logit-Based Distillation

- Student mimics soft token distributions
- Trains with cross-entropy or KL divergence:

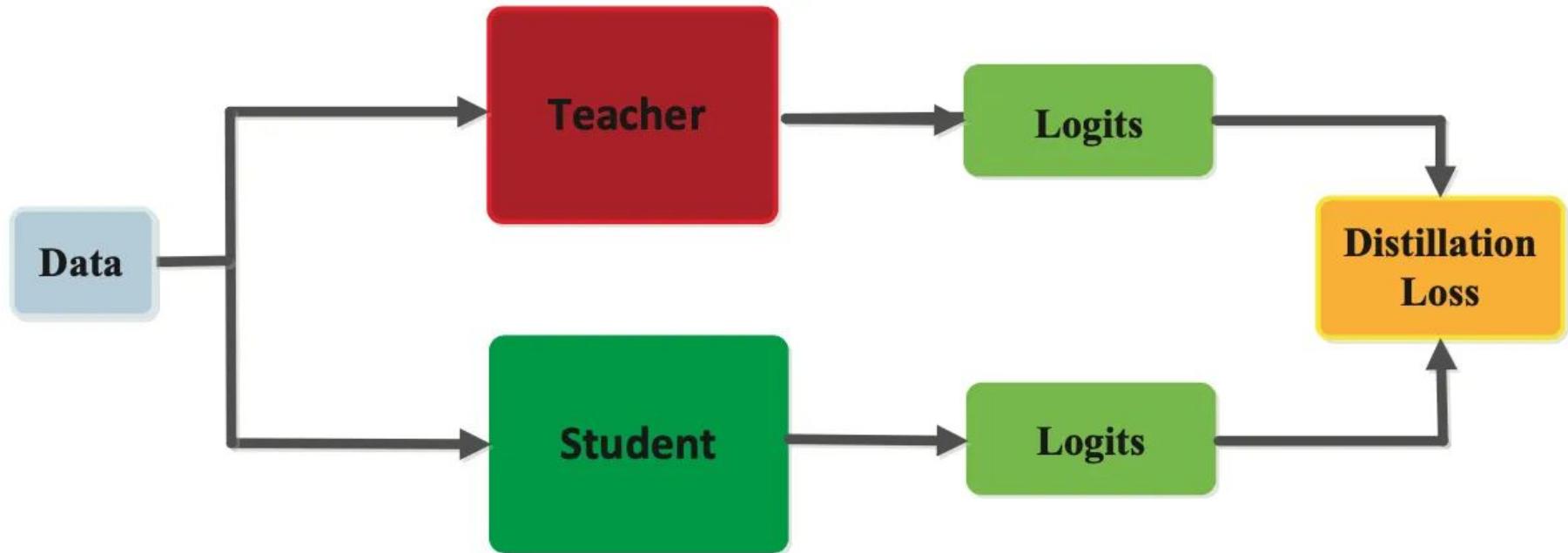
$$\mathcal{L}_{KL}(p_T, p_S) = \sum_i p_T(i) \log \frac{p_T(i)}{p_S(i)}$$

- Requires temperature scaling for softened targets



$$\mathcal{L}_{KL}(p_T, p_S) = \sum_i p_T(i) \log \frac{p_T(i)}{p_S(i)}$$

Logit-Based Distillation



From: [Knowledge Distillation for LLMs: Techniques and Applications | by Yugank .Aman | Feb, 2025 | Medium](#)



SAPIENZA
UNIVERSITÀ DI ROMA

Sequence-Level Distillation

- Student reproduces generated sequences from teacher
- Use text-level objectives: ROUGE, BLEU, or edit distance
- Less sensitive to vocabulary alignment

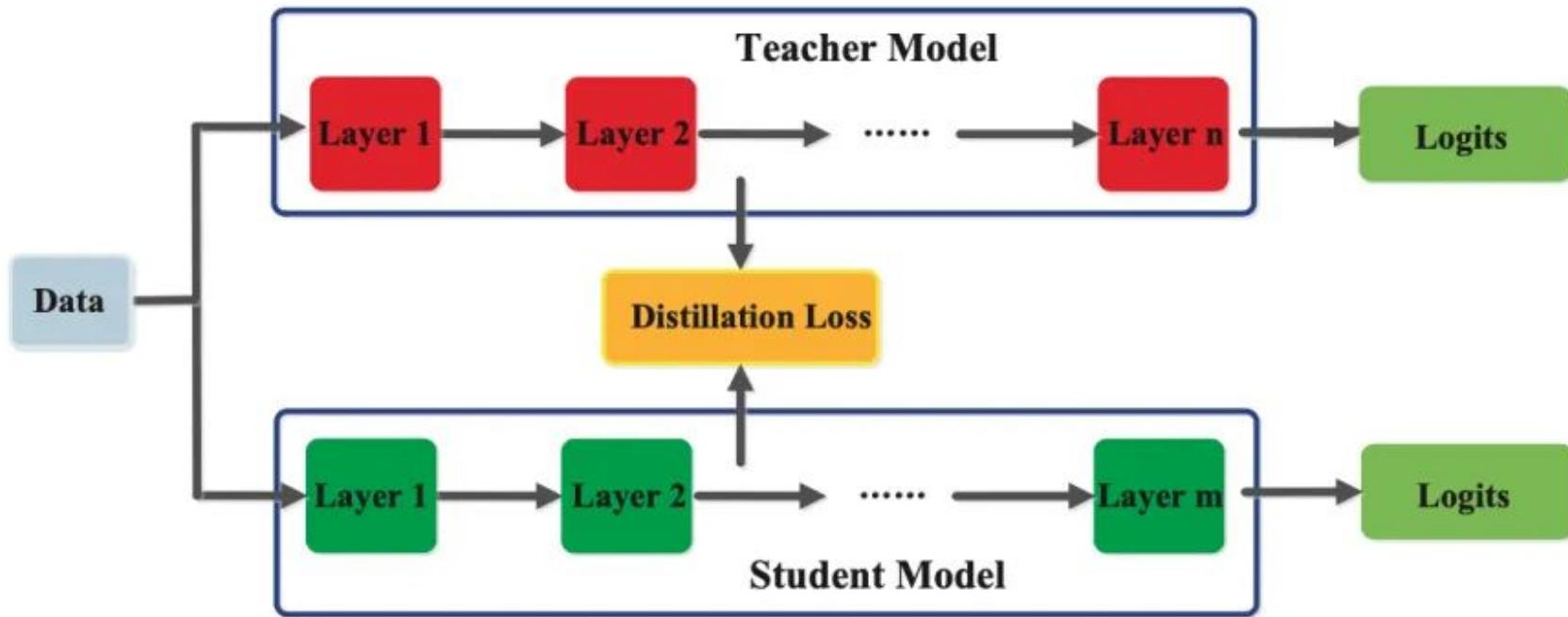


Representation Distillation

- Student aligns with teacher's hidden layers or attention heads
- Losses include MSE or contrastive similarity
- Useful when intermediate reasoning steps matter



Representation Distillation



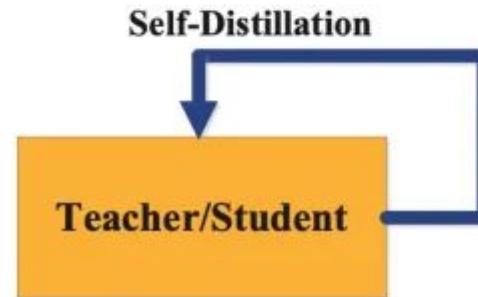
From: [Knowledge Distillation for LLMs: Techniques and Applications | by Yugank .Aman | Feb, 2025 | Medium](#)



SAPIENZA
UNIVERSITÀ DI ROMA

Self-Distillation

- Same model plays both roles over time
- Augments learning with self-generated targets
- Enables bootstrapping or task adaptation

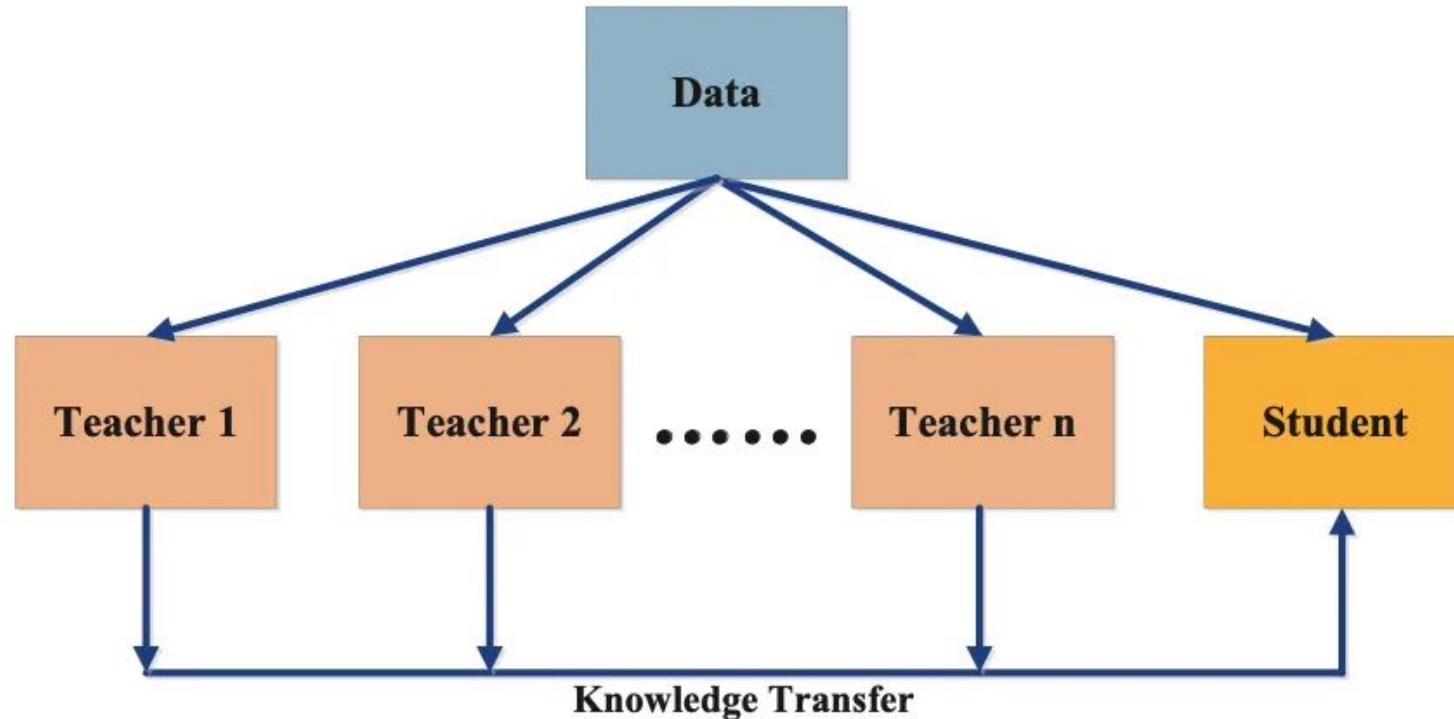


Multi-Teacher Distillation

- Combine knowledge from several teachers
- Used in diverse or ensemble settings
- Aggregate logits or voting strategies



Multi-Teacher Distillation



From: [Knowledge Distillation for LLMs: Techniques and Applications | by Yugank .Aman | Feb, 2025 | Medium](#)



SAPIENZA
UNIVERSITÀ DI ROMA

Evaluation Dimensions

- **Functional:** Accuracy, F1, BLEU, ROUGE
- **Efficiency:** Speedup, memory use, FLOPs
- **Behavioral:** Calibration, robustness, toxicity
- **Faithfulness:** Agreement with teacher



Data Strategy

- Natural datasets
- Prompt-generated outputs
- Augmented or adversarial examples



Challenges

- Preserving reasoning without copying hallucinations
- Evaluating semantic equivalence beyond surface form
- Distilling alignment, safety, or preferences
- Tradeoffs in compression vs. expressiveness



MinILLM: Knowledge Distillation Of Large Language Models

Published as a conference paper at ICLR 2024

MINILLM: KNOWLEDGE DISTILLATION OF LARGE LANGUAGE MODELS

Yuxian Gu^{1,2*}, Li Dong², Furu Wei², Minlie Huang^{1†}

¹The CoAI Group, Tsinghua University ²Microsoft Research

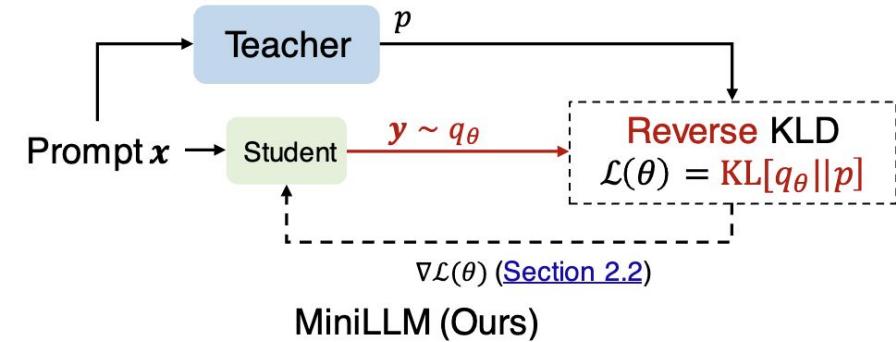
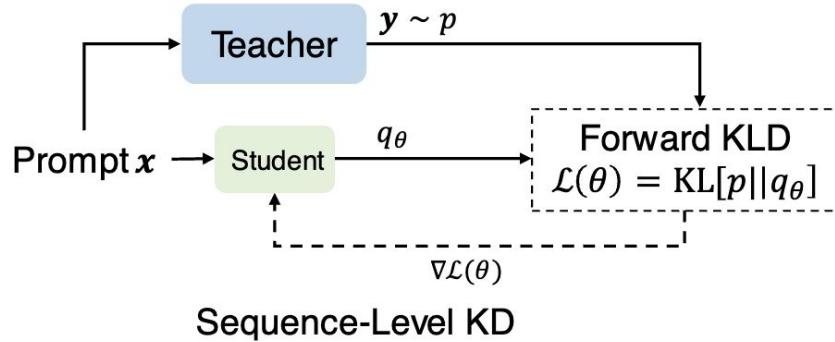
guyx21@mails.tsinghua.edu.cn {lidong1,fuwei}@microsoft.com
aihuang@tsinghua.edu.cn

From: [MinILLM: Knowledge Distillation of Large Language Models | OpenReview](#)



SAPIENZA
UNIVERSITÀ DI ROMA

Sequence-based Distillation vs. MiniLLM



Reverse KLD

We consider minimizing the *reverse* KLD between the student and teacher model distributions as the learning objective for MINILLM:

$$\begin{aligned}\theta &= \arg \min_{\theta} \mathcal{L}(\theta) = \arg \min_{\theta} \text{KL}[q_{\theta} || p] \\ &= \arg \min_{\theta} \left[- \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{x}}, \mathbf{y} \sim q_{\theta}} \log \frac{p(\mathbf{y} | \mathbf{x})}{q_{\theta}(\mathbf{y} | \mathbf{x})} \right]\end{aligned}$$

MinillM Optimization

$$\nabla \mathcal{L}(\theta) = - \mathbb{E}_{\substack{\mathbf{x} \sim p_{\mathbf{x}} \\ \mathbf{y} \sim \tilde{p}(\cdot | \mathbf{x})}} \left[\sum_{t=1}^T w_t \underbrace{\left[\nabla \sum_{y' \in V} q_{\theta}(y' | \mathbf{y}_{<t}, \mathbf{x}) \log \frac{p(y' | \mathbf{y}_{<t}, \mathbf{x})}{q_{\theta}(y' | \mathbf{y}_{<t}, \mathbf{x})} \right]}_{(\nabla \mathcal{L})_{\text{Single part}}} + R_{t+1}^{\text{Norm}} \underbrace{\frac{\nabla q_{\theta}(y_t | \mathbf{y}_{<t}, \mathbf{x})}{q_{\theta}(y_t | \mathbf{y}_{<t}, \mathbf{x})}}_{(\nabla \mathcal{L})_{\text{Long part}}^{\text{Norm}}} \right]$$

where:

$$R_{t+1}^{\text{Norm}} = \frac{1}{T-t-1} \sum_{t'=t+1}^T \log \frac{p(y_{t'} | \mathbf{y}_{<t'}, \mathbf{x})}{q_{\theta}(y_{t'} | \mathbf{y}_{<t'}, \mathbf{x})}$$

fixed teachermodel distribution $p(\mathbf{y} | \mathbf{x})$

student model distribution $q_{\theta}(\mathbf{y} | \mathbf{x})$ parameterized by θ

Reasoning LLMs



SAPIENZA
UNIVERSITÀ DI ROMA

Counting Letters

ChatGPT 4o ▾

How many r's are in lucertola, and a's? and o'? and l's?

In the word "lucertola", the letter counts are:

- r → 1
- a → 2
- o → 1
- l → 2



SAPIENZA
UNIVERSITÀ DI ROMA

Counting Letters

ChatGPT o1

How many r's are in lucertola, and a's? and o'? and l's?

< 2/2 >

Reasoned about letter count in lucertola for a couple of seconds

Let's look at each letter in "lucertola":

- **r**: There is exactly 1 'r'.
- **a**: There is exactly 1 'a'.
- **o**: There is exactly 1 'o'.
- **l**: There are exactly 2 'l's.

Hence, the counts are:

- **r** → 1
- **a** → 1
- **o** → 1
- **l** → 2



SAPIENZA
UNIVERSITÀ DI ROMA

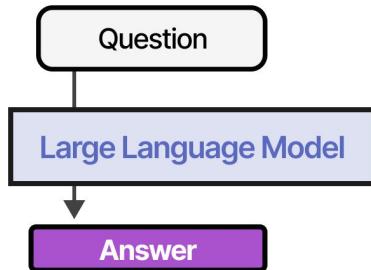
What are Reasoning LLMs

- Specialized large language models designed explicitly for complex reasoning tasks.
- Aim: Solve logical, mathematical, and analytical problems through step-by-step reasoning.

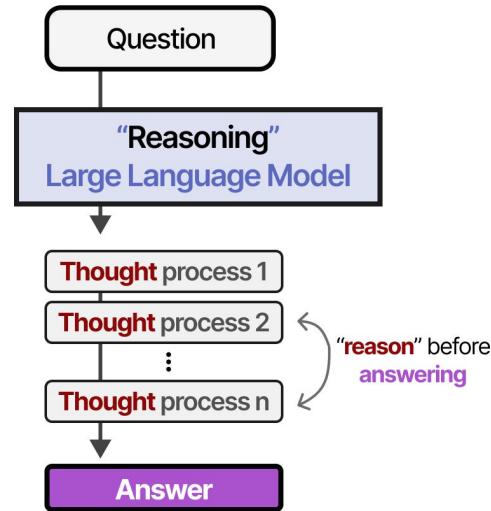


What are Reasoning LLMs

“Regular” LLMs

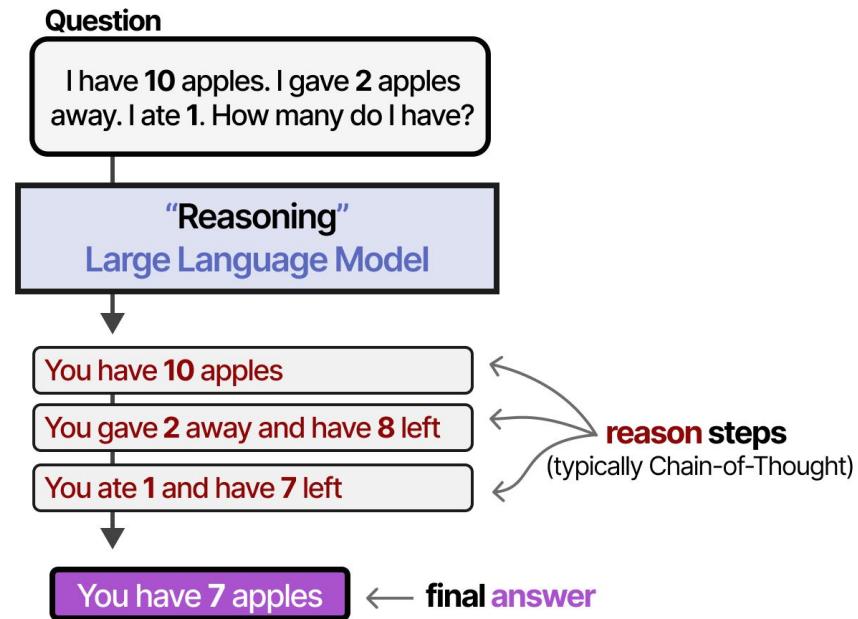


“Reasoning” LLMs

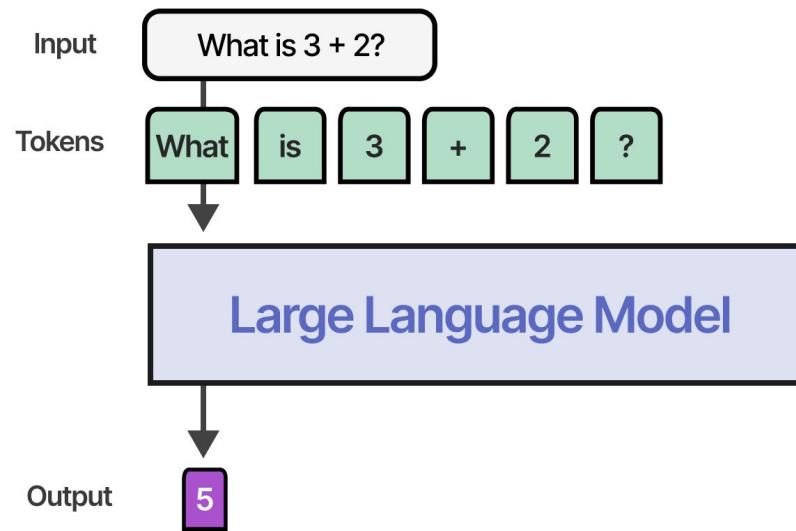


What are Reasoning LLMs

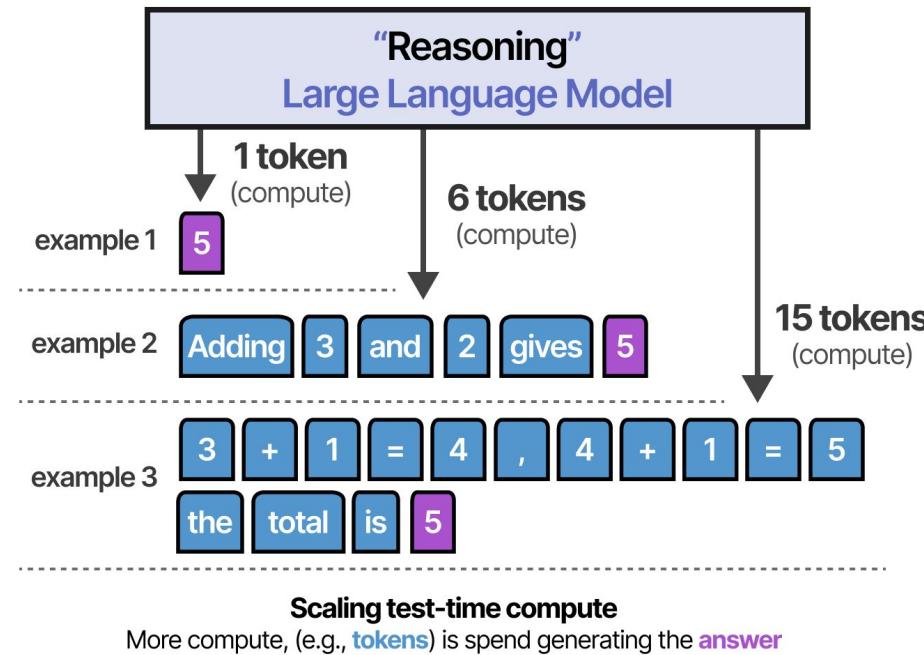
- Reasoning steps break down the process into smaller, structured inferences.



Training Time Compute



Reasoning Time Compute



Process Reward Model

Question

I have 10 apples. I gave 2 apples away. I then went and bought 5 more apples and ate 1. How many apples do I have?

Reasoning steps

First, you started with **10** apples.

PRM → 0.9

You gave away **2** apples, so you have **6** left.

PRM → 0.1

Then, you bought **5** more, so you now have **13** apples.

PRM → 0.6

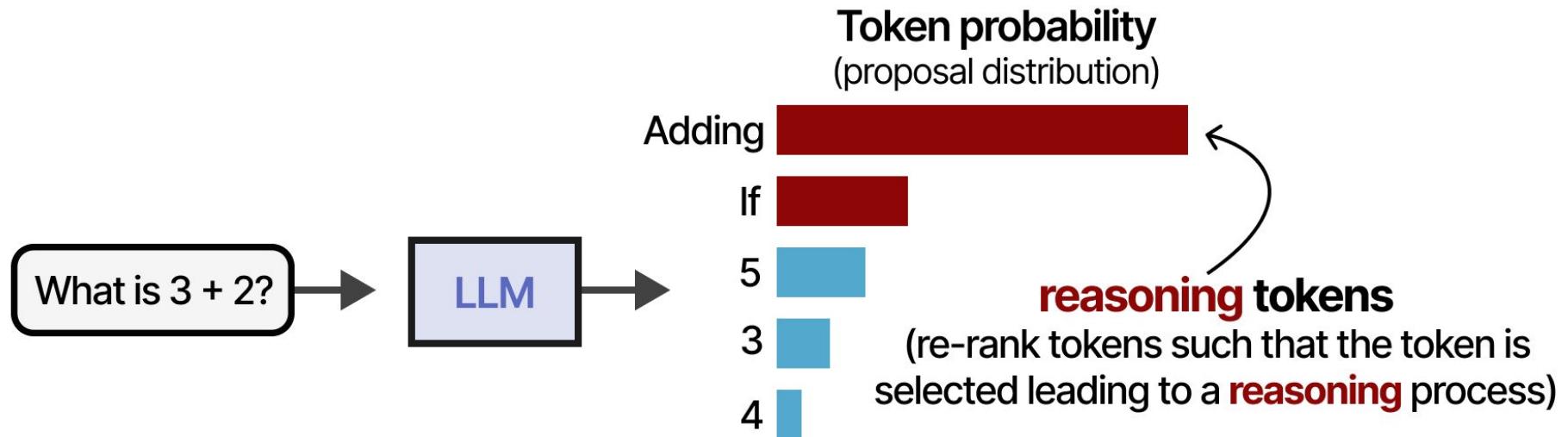
Finally, you ate **1** apple, so you **12** apples left.

PRM → 0.8

Final answer

12

Process Reward Model: Changing The Output Distro

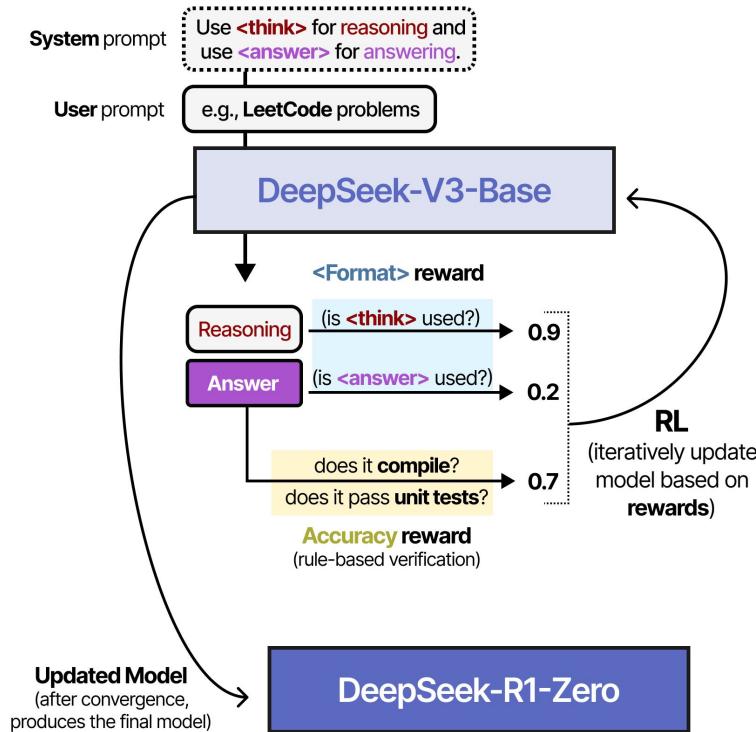


DeepSeek-R1's System Prompt

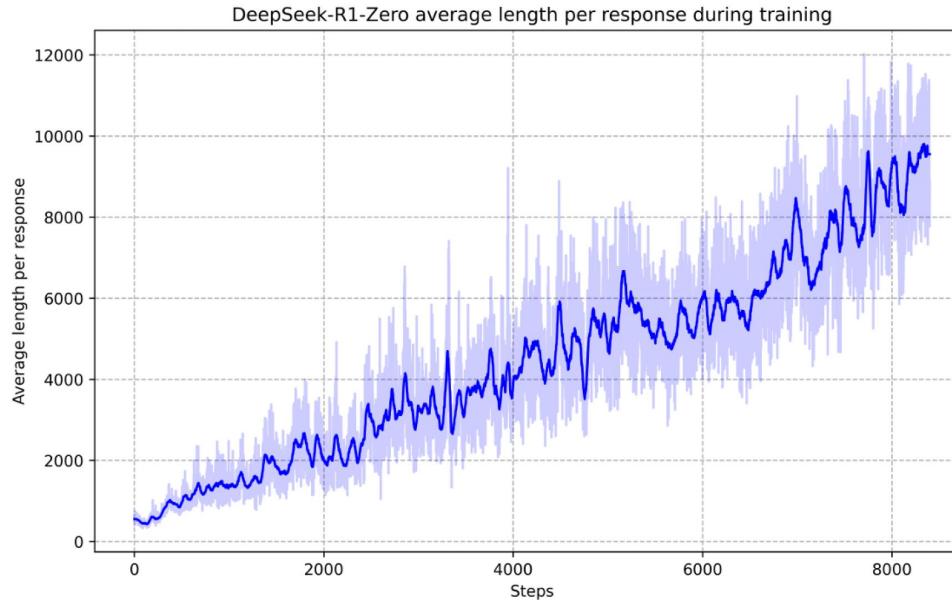
System prompt

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the **reasoning** process in the mind and then provides the user with the **answer**. The reasoning process and answer are enclosed within **<think></think>** and **<answer></answer>** tags, respectively, i.e., **<think> reasoning process here </think> <answer> answer here </answer>**.

DeepSeek-R1's Reasoning via RL



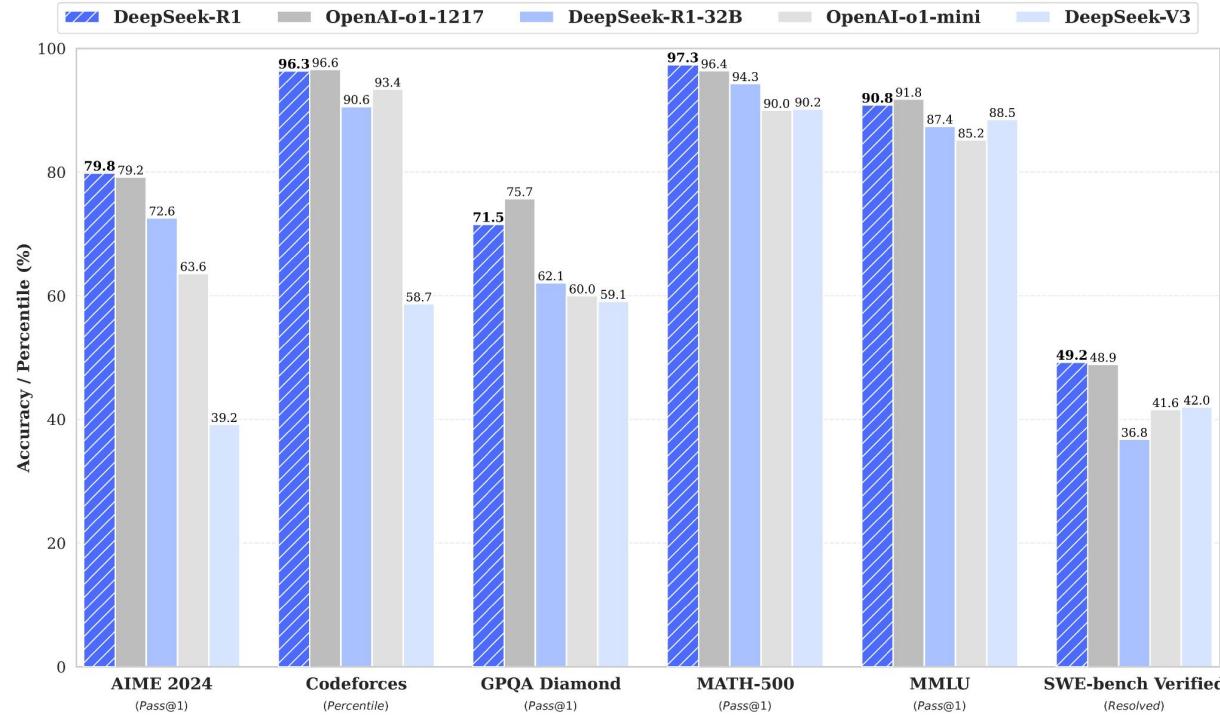
DeepSeek-R1's and Reasoning Steps



...the model learned to output **longer** and **longer reasoning** or **<think>** responses.

With each **training step**...

Reasoning LLMs Performance



From: deepseek-ai/DeepSeek-R1 · Hugging Face



SAPIENZA
UNIVERSITÀ DI ROMA

A Comprehensive Survey

“A SURVEY ON POST-TRAINING OF LARGE LANGUAGE MODELS” – <https://arxiv.org/pdf/2503.06072.pdf>



SAPIENZA
UNIVERSITÀ DI ROMA

Visual Transformers



SAPIENZA
UNIVERSITÀ DI ROMA

Attention In Images

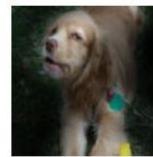
On which part of the image I have to focus on?



58

59

60



66

67

68



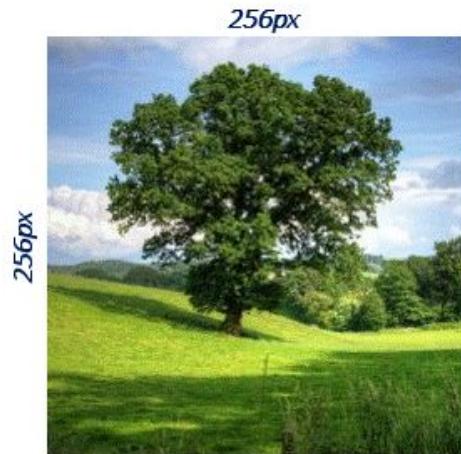
Transformers | Davide Coccomini | 2021



SAPIENZA
UNIVERSITÀ DI ROMA

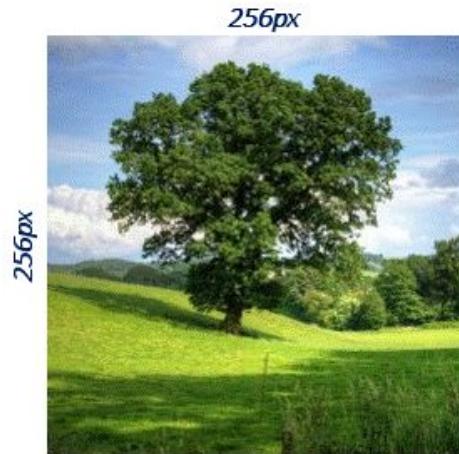
Can we use Pixels instead of Wordpieces in Transformers?

Treat pixels as words and calculate attention!



Chop Input and apply Transformers

An image is worth 16x16 words!



SAPIENZA
UNIVERSITÀ DI ROMA

Chop Input and apply Transformers



Experiments

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	88.55 ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	90.72 ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	99.50 ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	—
CIFAR-100	94.55 ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	—
Oxford-IIIT Pets	97.56 ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	—
Oxford Flowers-102	99.68 ± 0.02	99.74 ± 0.00	99.61 ± 0.02	99.63 ± 0.03	—
VTAB (19 tasks)	77.63 ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	—
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Table 2: Comparison with state of the art on popular image classification benchmarks. We report mean and standard deviation of the accuracies, averaged over three fine-tuning runs. Vision Transformer models pre-trained on the JFT-300M dataset outperform ResNet-based baselines on all datasets, while taking substantially less computational resources to pre-train. ViT pre-trained on the smaller public ImageNet-21k dataset performs well too. *Slightly improved 88.5% result reported in Touvron et al. (2020).



Transfer Learning

Transfer Learning with Image Data

- For example, a prediction task that takes photographs or video data as input.
- It is common to use a deep learning model **pre-trained** for a **large and challenging** image classification task such as the ImageNet 1000-class photograph classification competition.
- Three examples of models of this type include:
 - Oxford VGG Model
 - Google Inception Model
 - Microsoft ResNet Model



Model Zoo

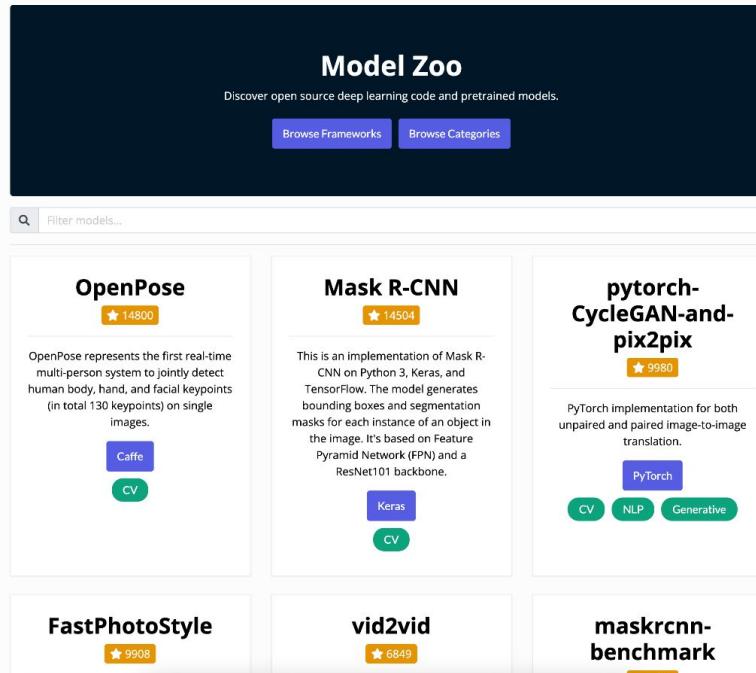
- It is common to use a **Model Zoo**.
- For Pytorch: https://pytorch.org/serve/model_zoo.html

Model	Type	Dataset	Size	Download	Sample Input	Model mode
AlexNet	Image Classification	ImageNet	216 MB	.mar	kitten.jpg	Eager
Densenet161	Image Classification	ImageNet	106 MB	.mar	kitten.jpg	Eager
Resnet18	Image Classification	ImageNet	41 MB	.mar	kitten.jpg	Eager
VGG16	Image Classification	ImageNet	489 MB	.mar	kitten.jpg	Eager
SqueezeNet 1_1	Image Classification	ImageNet	4.4 MB	.mar	kitten.jpg	Eager
MNIST digit classifier	Image Classification	MNIST	4.3 MB	.mar	0.png	Eager
Resnet 152	Image Classification	ImageNet	214 MB	.mar	kitten.jpg	Eager
Faster RCNN	Object Detection	COCO	148 MB	.mar	persons.jpg	Eager
MASK RCNN	Object Detection	COCO	158 MB	.mar	persons.jpg	Eager



ModelZoo.co

- For other frameworks (including Pytorch): <https://modelzoo.co/>



How to use Pretrained Models

- Colab Example

<https://colab.research.google.com/drive/1v82guGHX--OHwNdmLEQCGaaQVSjlqLql>



Transfer Learning with Text Data

- It is common to perform transfer learning with natural language processing problems that use text as input or output.
- First class of pre-trained models: word embedding
- Other examples:
 - BERT
 - T5
 - ...



HuggingFace!

- We have already seen Transformers
- We are now going to see how to actually use them in your ML Pipelines...



HuggingFace!



Transformers

build passing

license Apache-2.0

website online

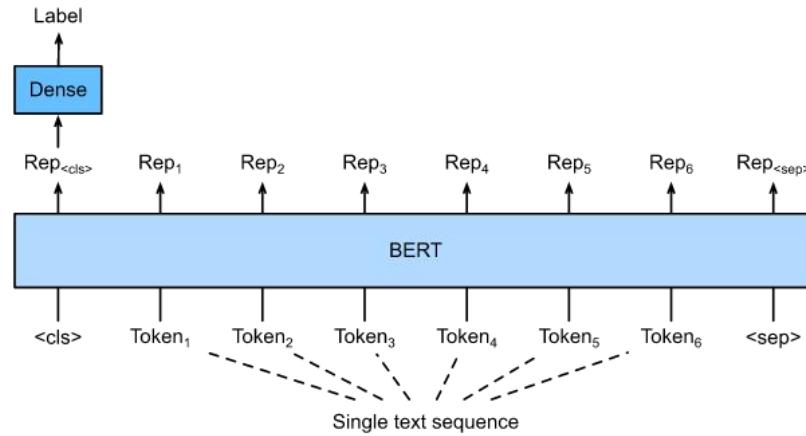
release v2.0.0



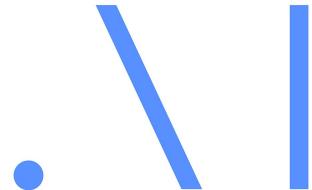
SAPIENZA
UNIVERSITÀ DI ROMA

How to fine-tune HF Pretrained Models

- Sentiment analysis using HF's BERT and fine-tune it.



An application of Transfer Learning: NeuralDBs



James Thorne
Cambridge



Majid Yazdani
Facebook AI



Marzieh Saeidi
Facebook AI



Sebastian Riedel
Facebook AI



Alon Halevy
Facebook AI

- J. Thorne, M. Yazdani, M. Saeidi, F. Silvestri, S. Riedel, and A. Halevy. *From natural language processing to neural databases.* PVLDB 2021
- J. Thorne, M. Yazdani, M. Saeidi, F. Silvestri, S. Riedel, and A. Halevy. *Database reasoning over text.* To appear in ACL 2022.



UNIVERSITY OF
CAMBRIDGE



SAPIENZA
UNIVERSITÀ DI ROMA



The talk is based on the following papers...

1. Giovanni Trappolini, Andrea Santilli, Emanuele Rodolà, Alon Y. Halevy, Fabrizio Silvestri: **Multimodal Neural Databases**. *SIGIR* 2023
2. Artsiom Sauchuk, James Thorne, Alon Y. Halevy, Nicola Tonello, Fabrizio Silvestri: **On the Role of Relevance in Natural Language Processing Tasks**. *SIGIR* 2022
3. James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, Alon Y. Halevy: **From Natural Language Processing to Neural Databases**. *VLDB* 2021
4. James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, Alon Y. Halevy: **Database reasoning over text**. *ACL/IJCNLP* 2021



Databases have a Well-Defined Formal Language

```
INSERT INTO People (PersonID, PersonName, Country) VALUES (123, 'Fabrizio Silvestri', 'Italy');  
INSERT INTO People (PersonID, PersonName, Country) VALUES (789, 'Marzieh Saeidi', 'UK');
```

```
INSERT INTO Jobs (JobID, JobDescription) VALUES (111, 'Software Engineer')  
INSERT INTO Jobs (JobID, JobDescription) VALUES (123, 'Research Scientist')
```

```
INSERT INTO PeopleJobs (PersonID, JobID) VALUES (123, 111)  
INSERT INTO PeopleJobs (PersonID, JobID) VALUES (789, 123)
```

```
SELECT      p.PersonName  
FROM        People p  
JOIN        PeopleJobs pj  
          ON (p.PersonID = pj.PersonID)  
JOIN        Jobs j  
          ON (pj.JobID = j.JobID)  
WHERE       j.JobDescription = "Research Scientist"
```



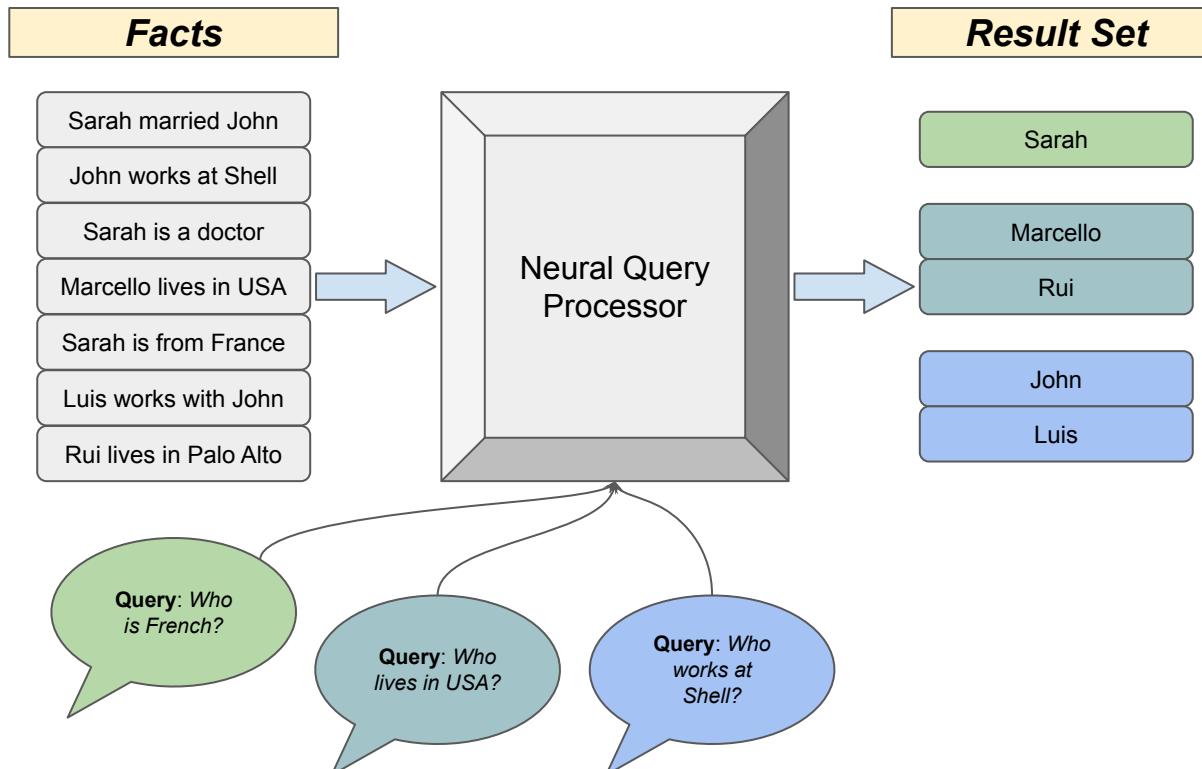
Database's Core Component: The Schema

“ The database schema of a database is its structure described in a formal language supported by the database management system (DBMS). The term ‘schema’ refers to the organization of data as a blueprint of how the database is constructed (divided into database tables in the case of relational databases). ”

from Wikipedia



What if... We Removed Schema from Databases?



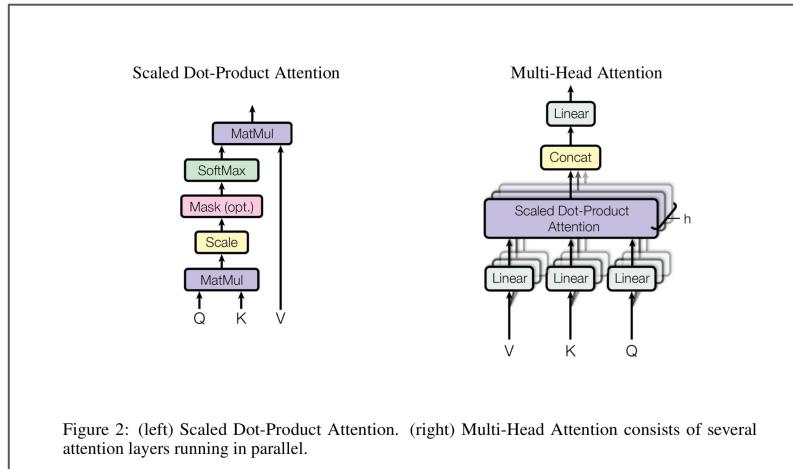
Is it QA?

- QA tasks deal with questions posed in natural language:
 - When were the Normans in Normandy?
 - Which kicker had most field goals?
 - Several datasets, i.e., SQuAD, DROP, MSMARCO-QA, etc.
- In QA typically the answer to a query is located within a passage or multiple passages that are (usually) locally available
 - In NeuralDBs facts that form a single result set might be scattered around in the dataset.
- In QA typically the answer is, well... “an answer” 😊 Typically a single sentence, e.g., “*Adam Vinatieri*”
 - In NeuralDBs we should target both sets of answers, and aggregations (count, avg, etc.).



An LLM Based Solution

- Transformers, e.g., BERT, have became ubiquitous in NLP.
- Introduced in the famous ‘Attention Is All You Need’ paper.
- It consists in applying (self-)attention to each token of a sequence of text, i.e., subwords.

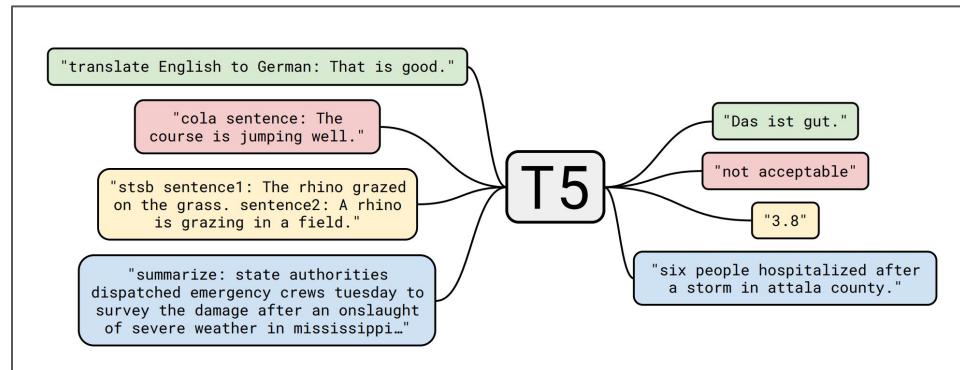


Credits: ‘Attention Is All You Need’ by Vaswani et al.



Google's T5 (Text-to-Text Transfer Transformer) Model

Trained in a multi-tasking fashion on the following tasks: (i) *GLUE* and *SuperGLUE* meta-tasks; (ii) CNN/Daily Mail *Abstractive Summarization*; (iii) SQuAD Question Answering; and (iv) WMT English to German, French, and Romanian *Translation*



Neural Query Processing

- Task:
 - Given a query and a small number of facts from the database, can the T5 accurately answer queries that are posed in natural language, whose answer may require projection (i.e., extracting part of a sentence), join, and aggregation?
- Data: 7 different relationships from Wikidata extracted using a template
 - we generate a training, validation and held-out test set containing 535, 50, and 50 databases respectively
 - Each database contains 50 facts and has 100-200 QA pairs
 - In total: 60,000 training, 5,500 validation and 6,000 test instances
- Input: To provide input to the transformer, we jointly encode relevant facts from the database by concatenating them with the query (separated by a special delimiter token)



Some Example Queries

Facts: (8 of 500 shown)

- Nicholas lives in Washington D.C. with his wife.
- Sheryl is Nicholas's wife.
- Teuvo was born in 1912 in Ruskala.
- Sheryl's mother gave birth to her in 1978.
- Nicholas is a doctor.
- Sarah was born in Chicago in 1982.
- Sarah married John in 2010.
- Sarah works in a hospital in NY as a doctor.

Queries:

List everyone born before 1980.

(Set) → Sheryl, Teuvo, ...

Whose spouse is a doctor?

(Join) → Sheryl, John, ...

Who is the oldest person?

(Max) → Teuvo

Who is Sheryl's mother?

(Set) → NULL



How to build Facts and Queries?

- Training a NL database requires supervision in the form of (D, Q, A) :
 - D is a set of **facts**
 - Q is a **query**
 - A is the correct **answer**
 - We generate training data in a controlled fashion by transforming structured data from Wikidata into NL facts and queries
 - Pros:
 - Scale
 - Breadth
- 
- (Subject, Relation, Object)**
(Bezos, employedBy, Amazon)



Facts

- We “verbalize” knowledge graph triples that are synthesized through a sequence to sequence model
 - Data is from KELM, we generate a rule-based post-hoc mapping back to Wikidata considering: string similarity, and compatibility of the generated triple

Input Triples	Target Sentence
Das Tagebuch der Anne Frank, (distributor, Universal Pictures), (country, Germany), (publication date, 03 March 2016)	The film was theatrically released in the Germany on March 3, 2016, by Universal Pictures International.
Neff Maiava, (date of birth, 01 May 1924), (date of death, 21 April 2018), (occupation, professional wrestler)	Maiava (May 1, 1924 April 21, 2018) was an American Samoan professional wrestler.
Barack Obama 2012 presidential campaign, (country, United States), (end time, 06 November 2012), (start time, 04 April 2011)	The 2012 reelection campaign of Barack Obama, the 44th President of the United States, was formally announced on April 4, 2011.
Blue whale (parent taxon, Balaenoptera)	The blue whale (<i>Balaenoptera musculus</i>) is a marine mammal belonging to the baleen whale suborder Mysticeti.

Agarwal, O., Ge, H., Shakeri, S. and Al-Rfou, R., 2020. Large Scale Knowledge Graph Based Synthetic Corpus Generation for Knowledge-Enhanced Language Model Pre-training
arXiv preprint arXiv:2010.12688.



Queries

- We generate queries using a number of templates for each property and question type
 - (X, bornIn, Y)
 - $(X, \text{employedBy}, Y)$
- We also “simulate” joins by “chaining” triples:
 - $(Y, \text{locatedIn}, Z) \bowtie (X, \text{employedBy}, Y) \rightarrow \text{“Does \$X work at a company located in \$Z?”}$
 - $(Y, \text{marriedTo}, Z) \bowtie (Z, \text{leavesIn}, Y) \rightarrow \text{“Does \$Z’s spouse leaves in \$Y?”}$
 - $(Y, \text{childOf}, Z) \bowtie (Y, \text{bornIn}, X) \sqcap (Y', \text{bornIn}, X') \rightarrow \text{“Is \$Z’s child younger than \$Y?’”}$
 - $(Y, \text{rel1}, Z) \bowtie (Z, \text{rel2}, Y) \rightarrow \text{“Does \$Z’s rel1 also rel2 \$Y?”}$



Example Queries, Supporting Facts, and Correct Answer

Example: Set

Question

Who studied at University of Minnesota?

Supporting Facts

1. [John B Totushek was born on 7 September 1944 in Minneapolis. He attended the University of Minnesota and became a US Naval Aviator. Mr. Totushek was also a human being.]
2. [Melvin Maas graduated from the University of Minnesota and is buried at Arlington National Cemetery. He is a native of Minnesota and his language is English.]
3. [Clarence Larson graduated from the University of Minnesota and is a member of the National Academy of Engineering.]
4. [Ted Mann, who is the surname of Ted Mann, attended Duke University and the University of Minnesota. He is a human being.]

Answer

[John B. Totushek, Ted Mann, Clarence Larson, Melvin Maas]



Example Queries, Supporting Facts, and Correct Answer

Example: count

Question How many people work for Yale Law School?

Supporting Facts

- 1. [Michael Ponsor, born in Oxford, graduated from Pembroke College in Oxford. He was awarded the Rhodes Scholarship and is an employee at Yale Law School. He is an expert in the field of human rights.]
- 2. [Stephen Wizner is an American legal scholar who graduated from Dartmouth College and is a graduate of the University of Chicago Law School. He works at Yale Law School.]

Answer 2



Example Queries, Supporting Facts, and Correct Answer

Example: Min/Max

Question What is the largest yearly attendance?

Supporting Facts

- 1. [The musee en herbe has a visitor per year of] 70000.
- 2. [The total number of visitors to the Hirschsprung Collection is 71779 per year.]
- ...
- 24. [The Tate Modern has a visitor count of 5839197 visitors per year.]
- 25. [Catoctin Mountain Park attracts 221750 visitors per year.]

Answer 5839197



Example Queries, Supporting Facts, and Correct Answer

Example: Bool

Question Is North Carolina State University the employer of Wes Moore?

Supporting Facts 1. [Wes Moore is a human being who is employed at Francis Marion University and is a basketball player for North Carolina State University.]

Answer TRUE



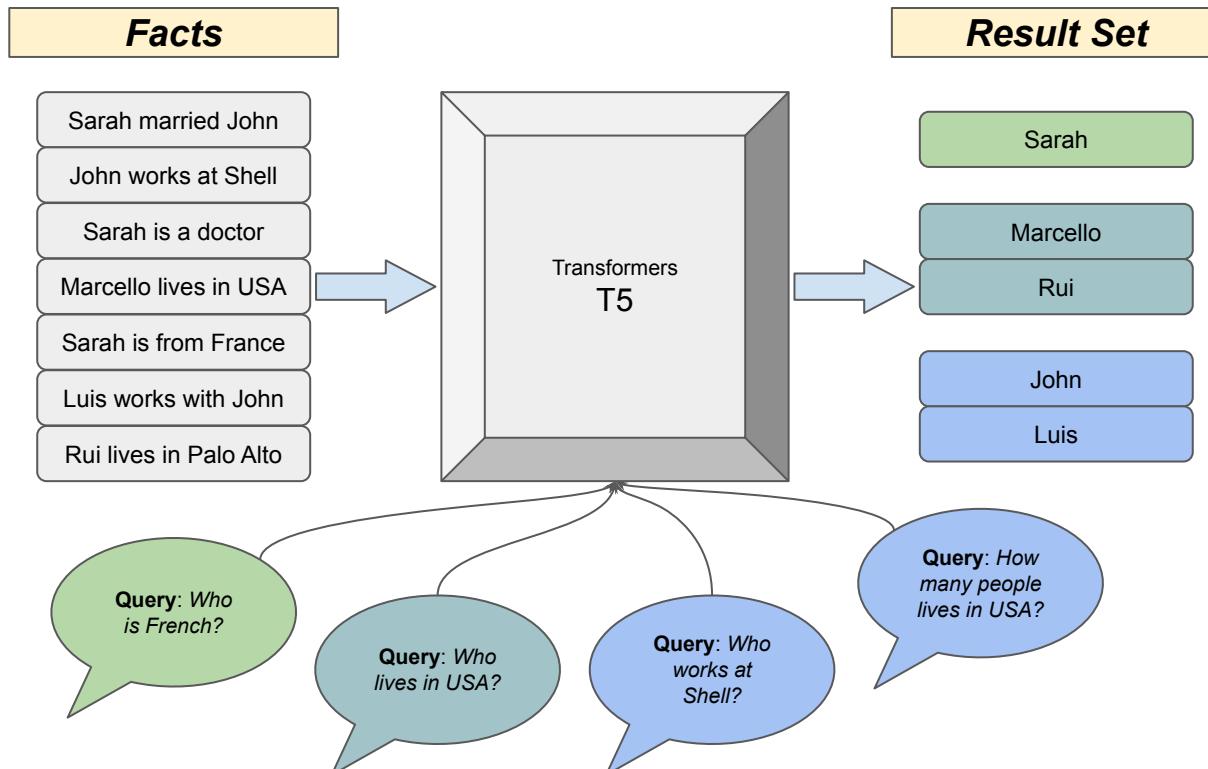
Example Queries, Supporting Facts, and Correct Answer

Example: Join

Question	Who plays for a team in Ligue 1?
Supporting Facts	<p>1. [Thomas Allofs started his career in 1989 with RC Strasbourg Alsace. He finished his career in 1990.,</p> <p>RC Strasbourg Alsace is an association football club in the Ligue 1 league. It was founded in 1906 and is located in Strasbourg, France.]</p>
Answer	[Thomas Allofs]



A “Simple” Solution



Possible Issues

- ✓ When fed with relevant facts from the database, T5 can produce results with reasonable accuracy
- ✗ Aggregation queries need to be performed outside of the neural machinery
- 👉 In order to handle queries that result in sets of answers and in order to prepare sets for subsequent aggregation operators, we need to develop a neural operator that can process individual (or small sets of) facts in isolation and whose results outputted as the answer or fed into a traditional (i.e. non-neural) aggregation operator.

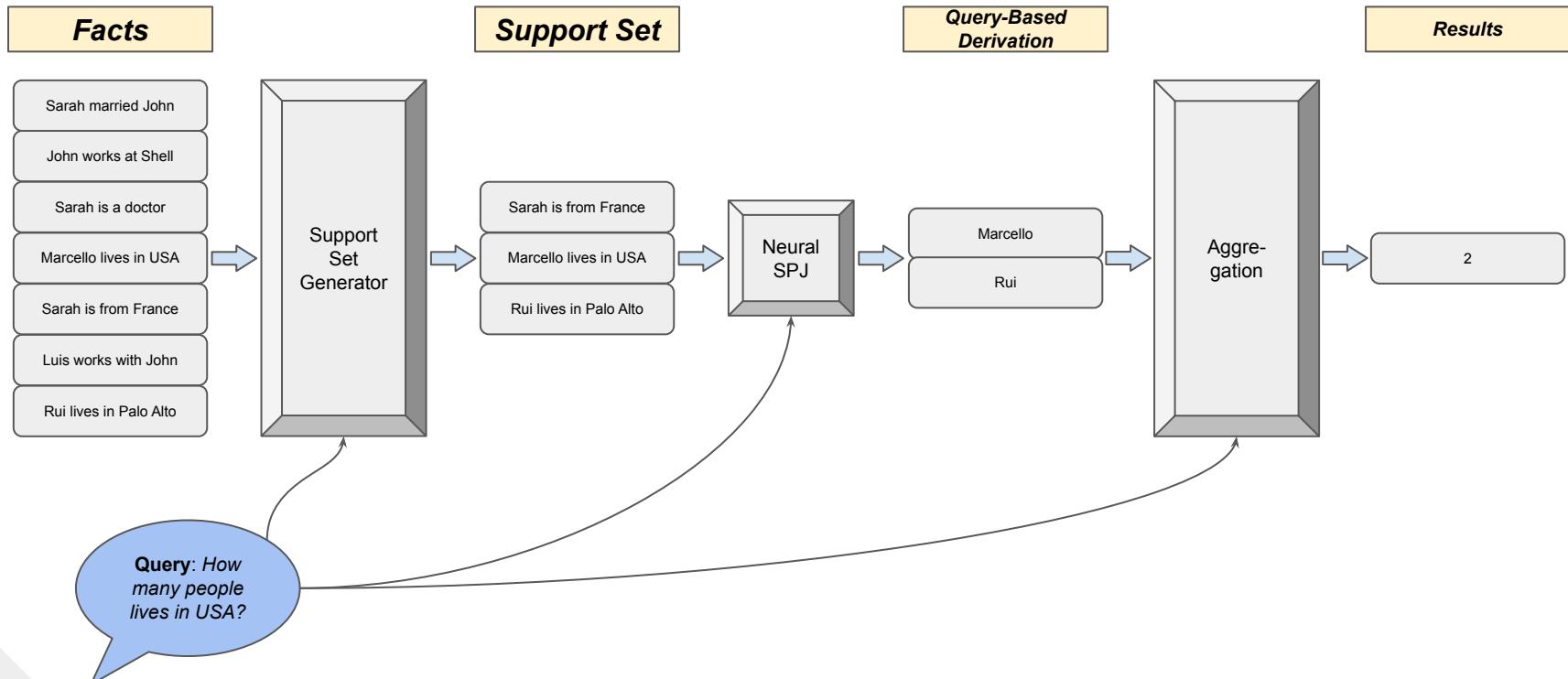


Challenges

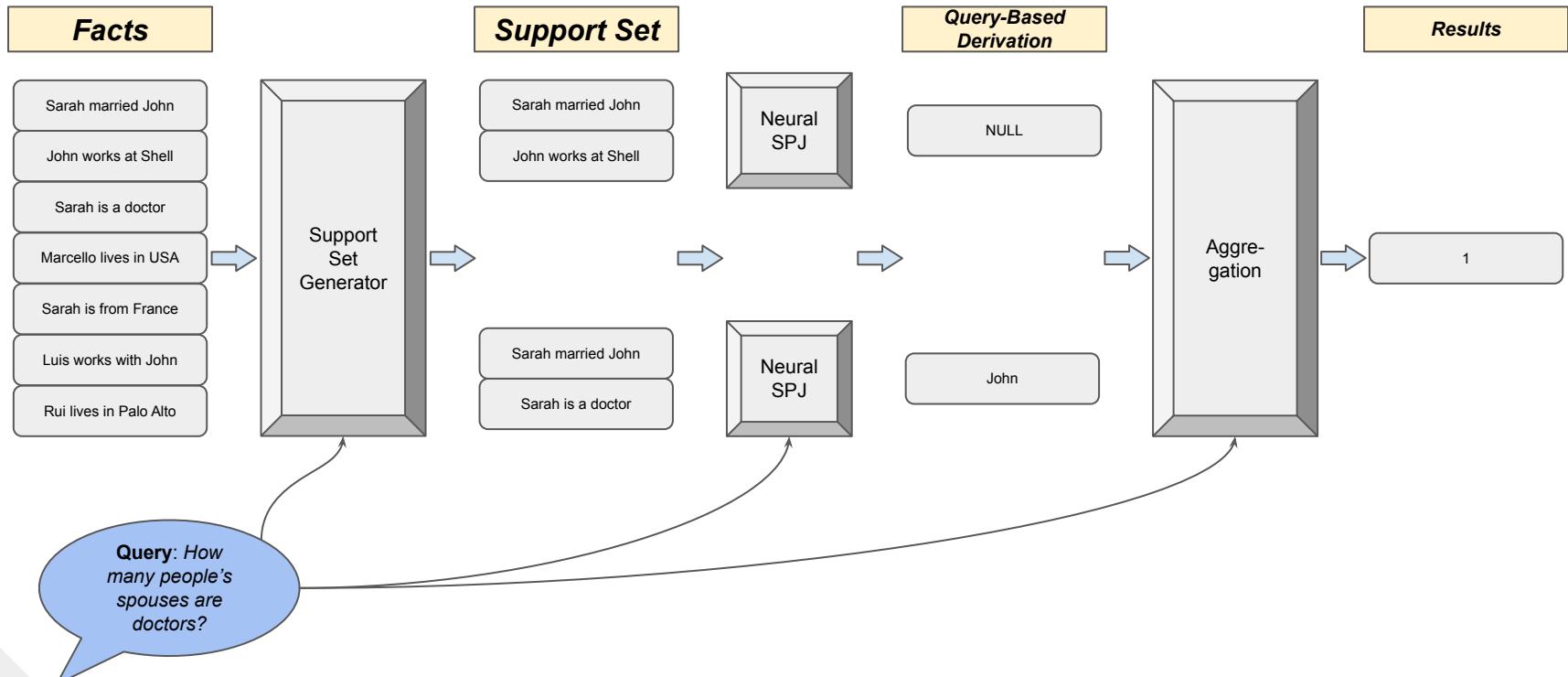
- **Scale**
 - neural reasoning to databases of non-trivial size
 - In open-domain QA we usually complement the transformer reasoning with an IR component that extracts a small subset of the facts from the corpus
- **Multiple answer spans**
 - NDB might need to generate 100K facts and aggregate over them
- **Locality and Document Structure**
 - Answers might be dependent on several facts scattered across the DB
- **Multi-hop and Conditioned Retrieval**
 - E.g., *Whose spouse is a doctor?*



The Neural DB Architecture

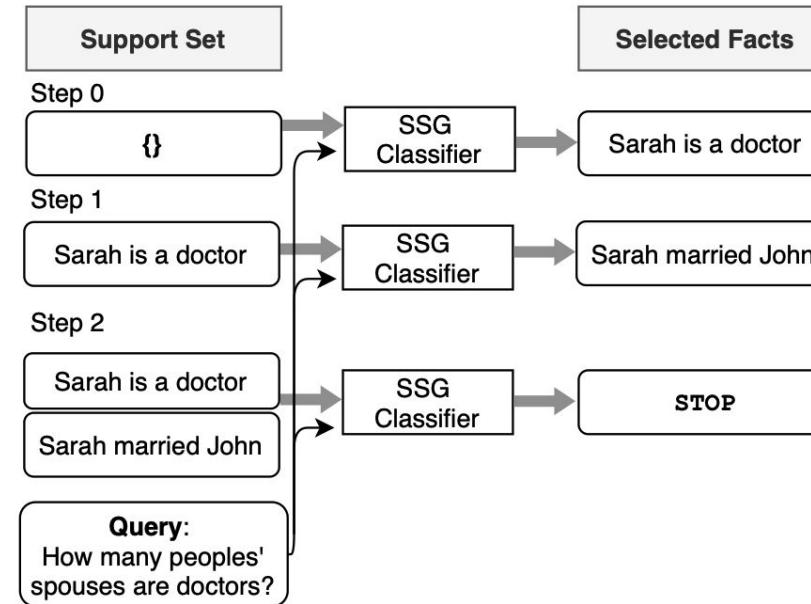


The Neural DB Architecture



Support Set Generator (SSG)

- Simple queries over single facts → TF-IDF based IR
 - not scalable for joins, aggregation queries or for queries outputting a set of answers as generating relevant sets requires incremental decoding, conditioning on already retrieved facts.



Neural Select-Project-Join (SPJ)

- For support sets that are insufficient to answer a question, the operator should return no output.
- For queries that require short chains of reasoning over multiple facts, the SPJ operator joins the facts when generating the output.
- SPJ generates a projection of the fact to a machine readable format dependent on the task, query and fact.
- Depending on the query type:
 - **Boolean Answers** → **binary value**
 - **Count/Set Queries** → **entities**
 - **Argmin/max operators** → **key-value pairs**.
 - For example “*Which place has the highest yearly number of visitors?*” has the projection of the form: (place,number of visitors).



Examples of Neural SPJ Outputs

- Query: Does Nicholas's spouse live in Washington D.C.?
 - $\{\text{Nicholas lives in Washington D.C. with Sheryl.}, \text{Sheryl is Nicholas's spouse.}\} \rightarrow \text{TRUE}$
- Query: Who is the oldest person in the database?
 - $\{\text{Teuvo was born in 1912.}\} \rightarrow (\text{Teuvo, 1912})$
- Query: Does Nicholas's spouse live in Washington D.C.?
 - $\{\text{Teuvo was born in 1912.}\} \rightarrow \text{NULL}$

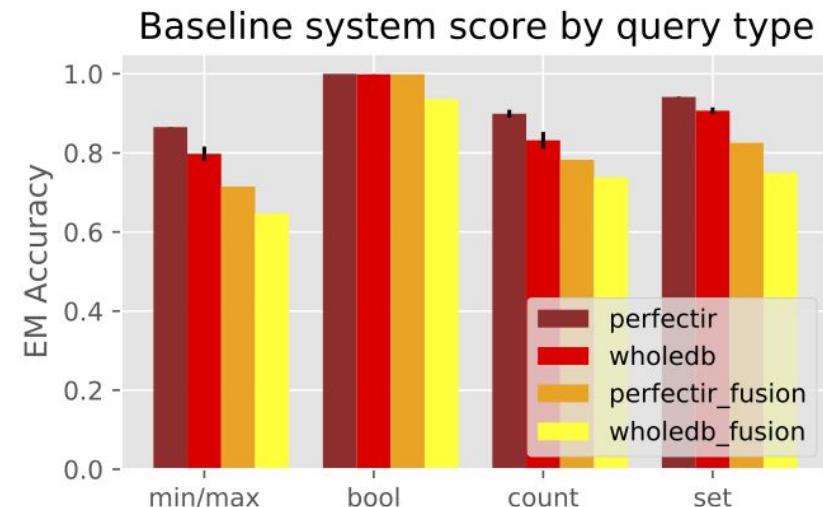
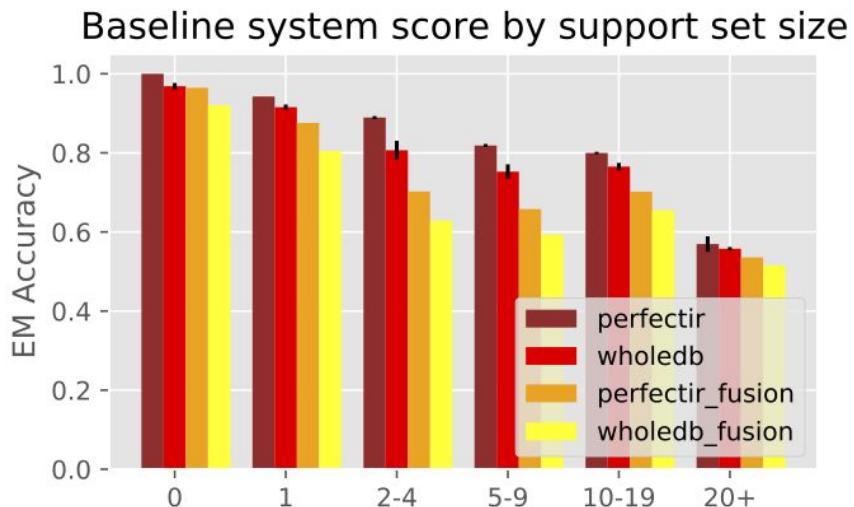


Results: SPJ Performance

Method	Exact Match (%)				
	Count	Min/Max	Sets	Atomic	Joins
NEURALDB	79.45	100.00	91.91	97.90	79.29
TF-IDF+T5	31.06	0.00	44.25	98.05	68.02
DPR+T5	38.07	21.19	54.55	97.38	58.64



Exact Match



Exact match accuracy for different classes of queries for a transformer model encoding up to 25 facts in one input. The results show that the model obtains high accuracy for queries performing Boolean inference, but falls short for queries with aggregation or yielding set answers (top) over multiple support sets (bottom).



Results: SPJ Performance

Method	Exact Match (%)			
	Min/Max	Bool	Count	Set
SPJ PerfectIR	88.3	99.8	90.1	89.4
SSG + SPJ	87.3	99.8	90.1	89.6

Using retrieved evidence achieves results competitive to the PerfectIR



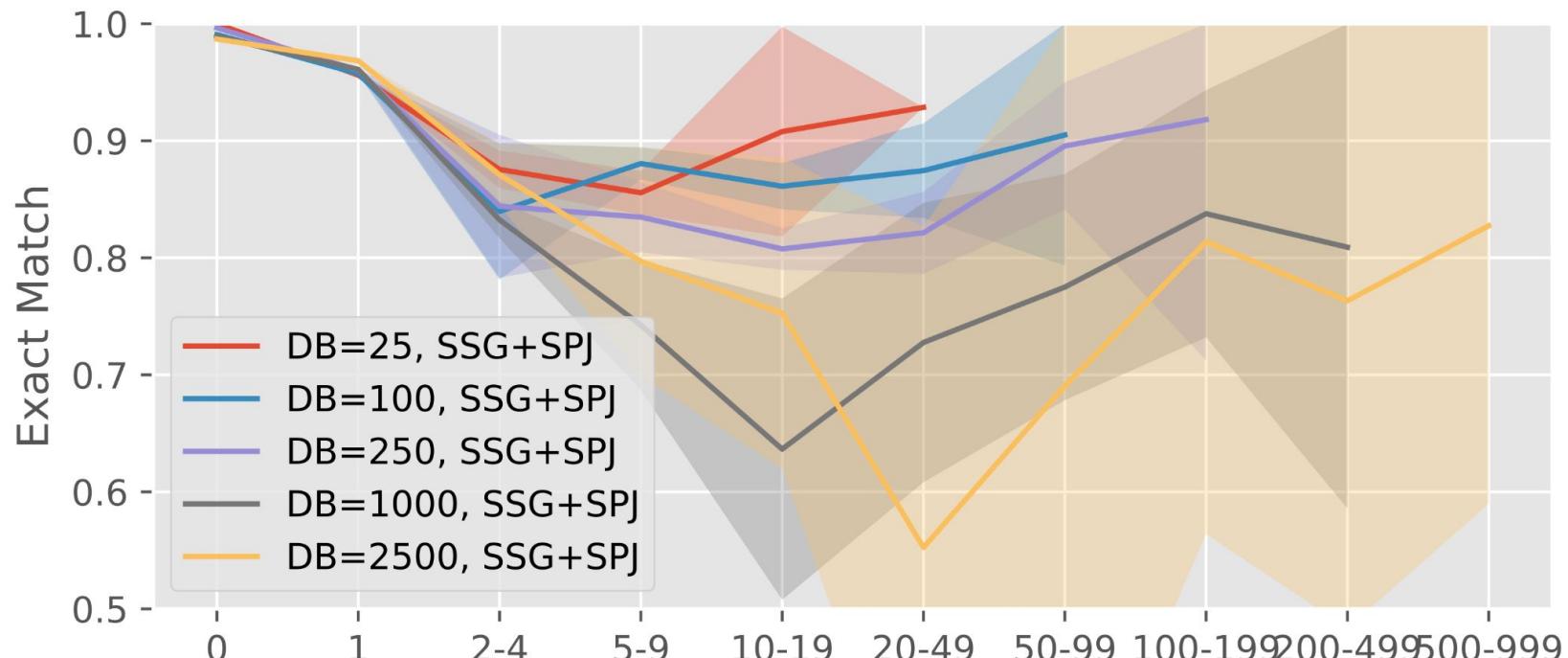
Results: SSG Precision/Recall

Query Type	Exact Match (%)		Soft Match (%)	
	Precision	Recall	Precision	Recall
Boolean	85.12	94.04	85.39	94.04
Set	61.05	94.58	61.33	94.58
Count	57.88	96.15	58.00	96.15
Min/Max	60.68	95.82	60.68	95.82
Join	38.33	75.39	42.74	75.43
<i>Average</i>	57.08	90.53	58.24	90.54

Precision and recall of supervised SSG w.r.t. the reference set. Note that errors in retrieval do not necessarily translate to wrong query answers because the SPJ operator is trained to be robust to noise.



Results: SSG + SPJ Accuracy (different DB Sizes)



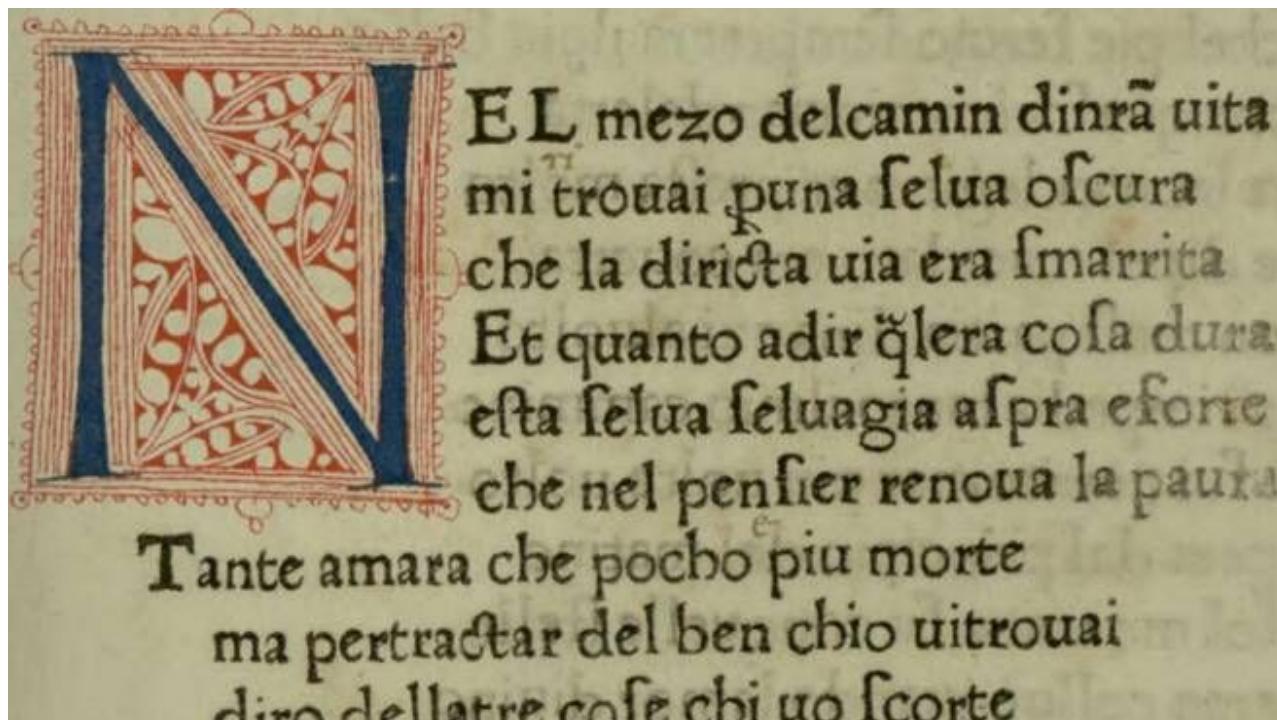
SSG+SPJ by support set size for all 5 databases. The SPJ is trained on databases of 25 facts and tested on larger databases. Low recall from SSG reduced answer EM for DBs of more than 1000 facts.



Neural Databases: Moving Beyond Text



Moving Beyond Text



Moving Beyond Text



ACL Anthology

FAQ Corrections Submissions

Search...



TimelineQA: A Benchmark for Question Answering over Timelines

Wang-Chiew Tan, Jane Dwivedi-Yu, Yuliang Li, Lambert Mathias, Marzieh Saeidi, Jing Nathan Yan, Alon Halevy

Your personal timeline

Date Range
03/02/2019 - 04/30/2019

All Events ▾

Exploring Roppongi Hills, Tokyo
Fri Mar 29 2019 03:20:53 GMT-0700 (Pacific Daylight Time)



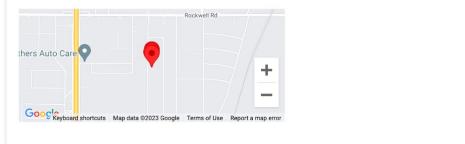
More details

Freshly prepared osechi sushi.
Fri Mar 29 2019 03:20:53 GMT-0700 (Pacific Daylight Time)



More details

Places you visited



Books you read

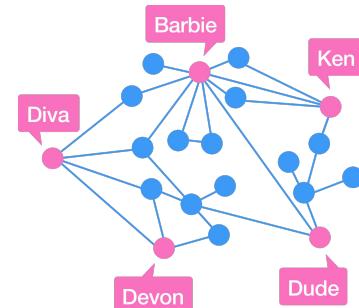
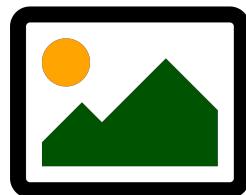


Purchases you made

Ficksar 48 Inch Steel D-handle Dinner Shovel

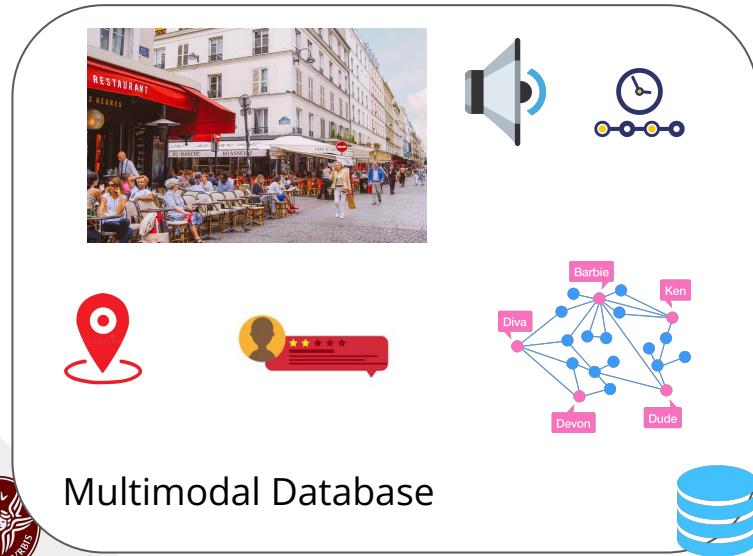


Multimodality



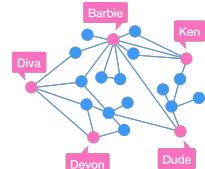


Multimodal NDB → The Vision



Multimodal NDB → The Vision

Which of the **restaurants** that I have seen today makes good **pasta** and play **jazz** music?



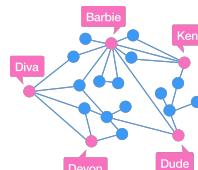
Multimodal Database



Multimodal NDB → The Vision

Which of the **restaurants** that I have seen today makes good **pasta** and play **jazz** music?

Multimodal Neural Databases



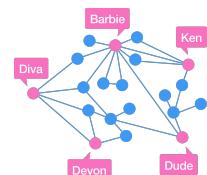
Multimodal Database



Multimodal NDB → The Vision

Which of the **restaurants** that I have seen today makes good **pasta** and play **jazz** music?

Multimodal Neural Databases



Multimodal Database



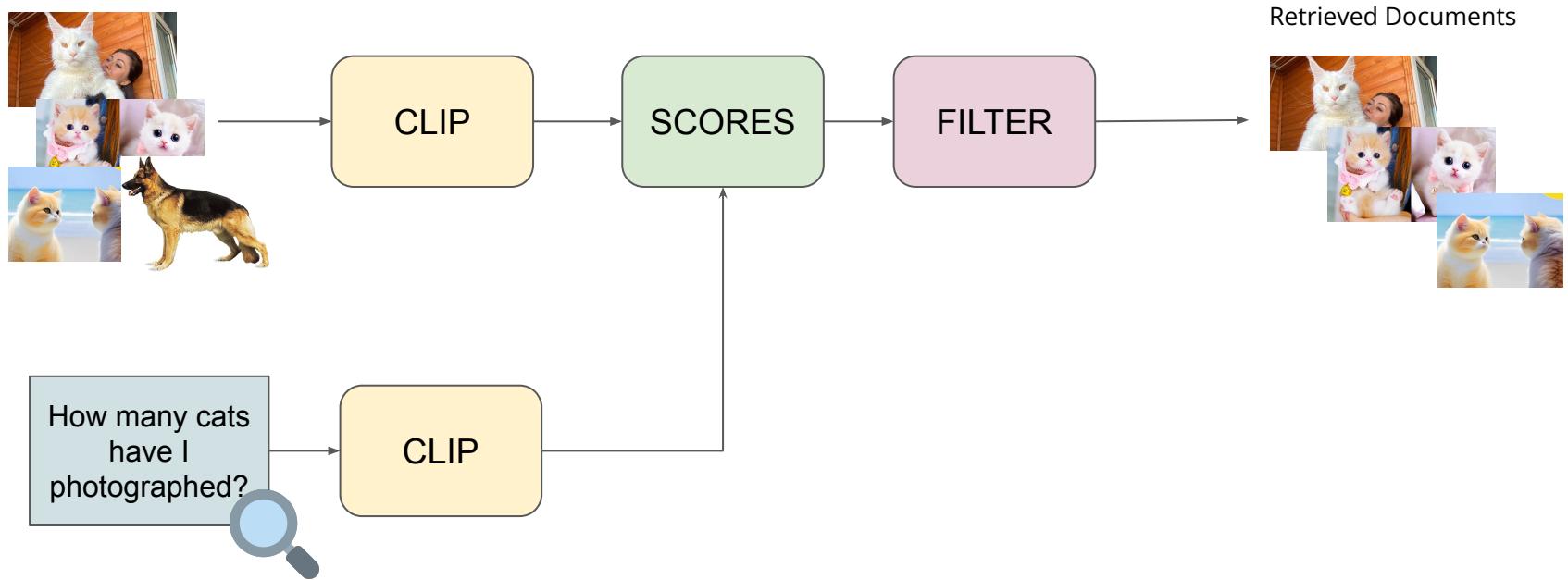
Pizzeria Le due Sorelle

John's Pasta

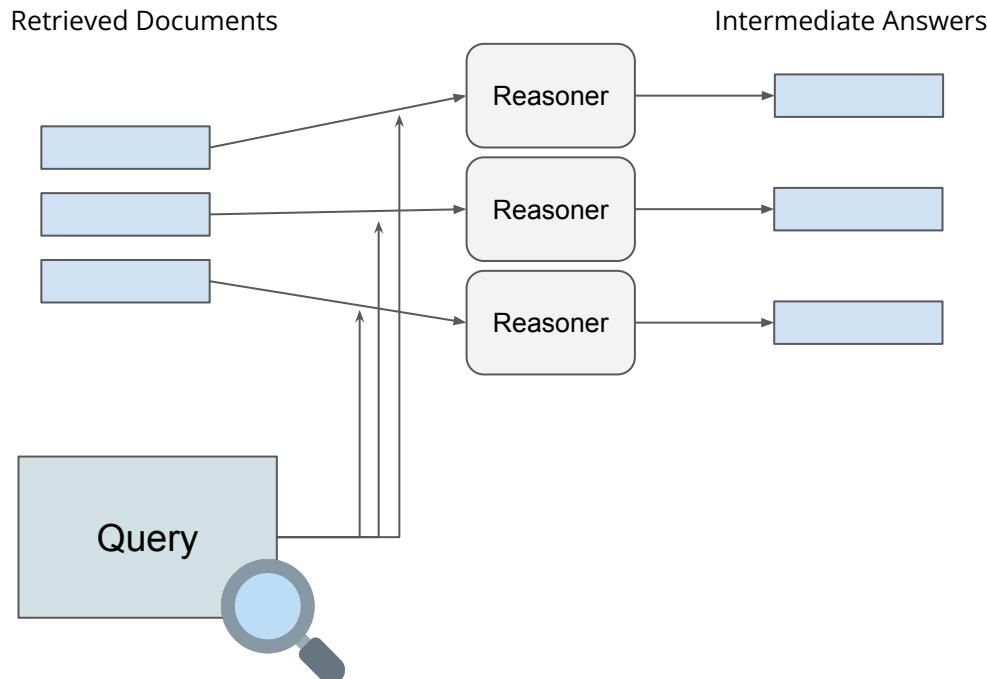
Answer Set(s)



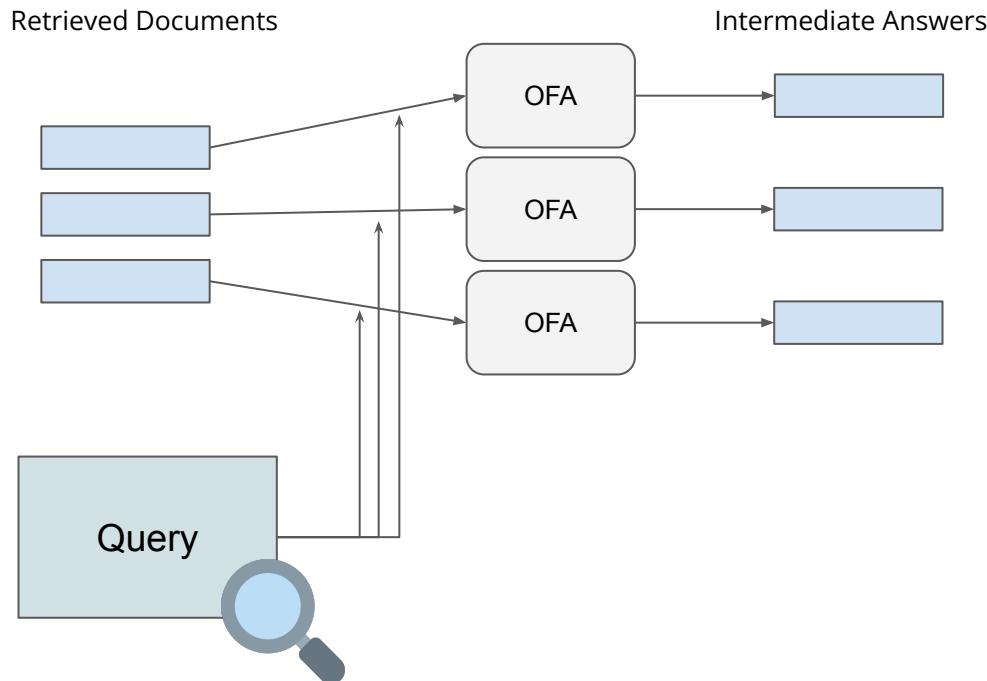
Proposed Architecture



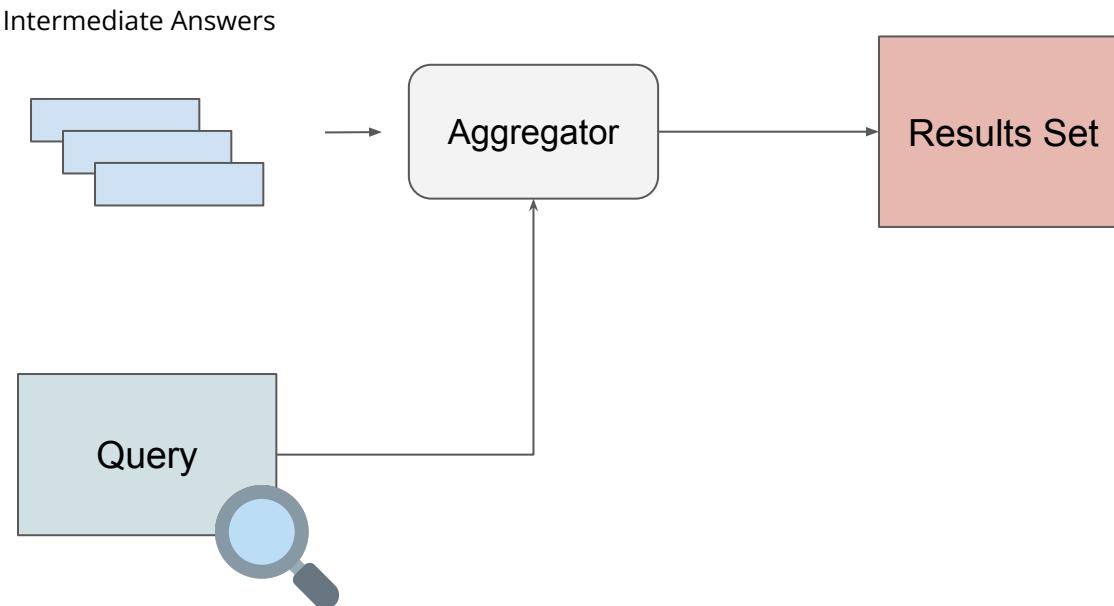
Proposed Architecture



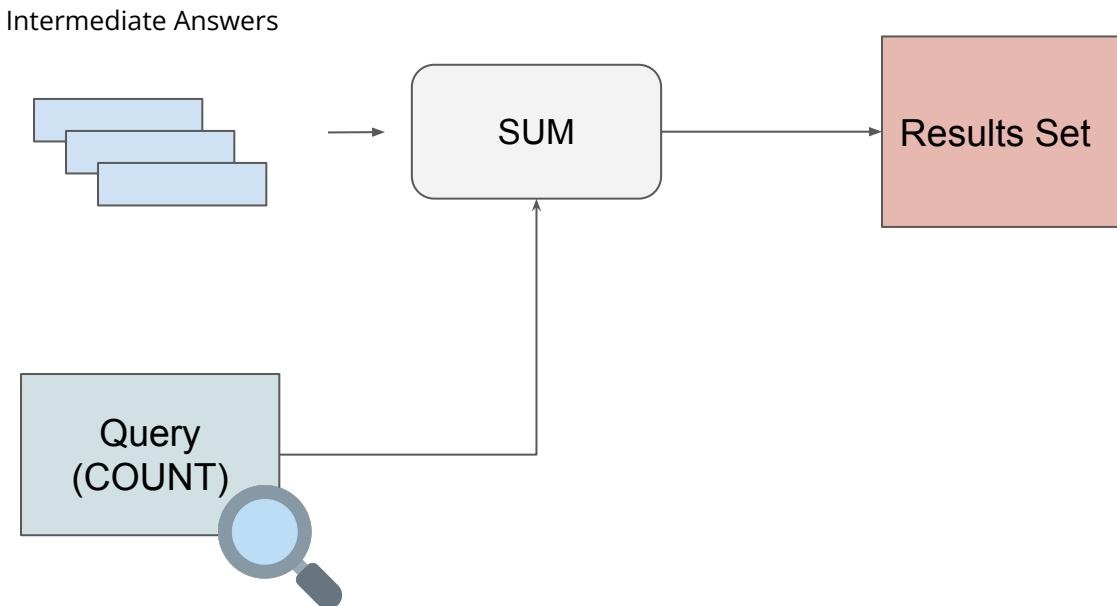
Proposed Architecture



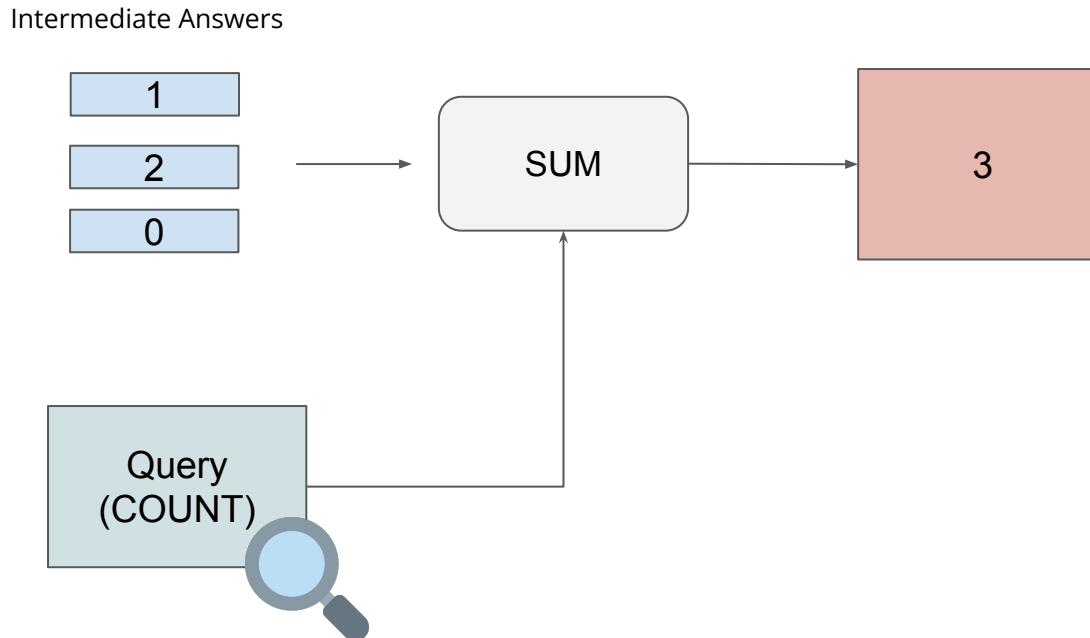
Proposed Architecture



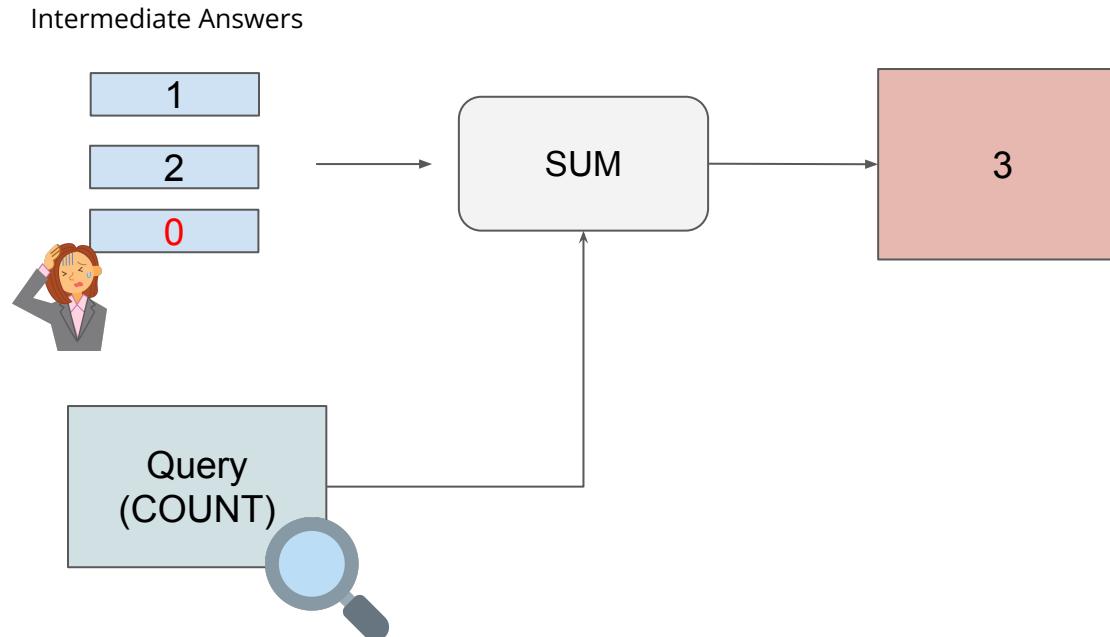
Proposed Architecture



Proposed Architecture



Proposed Architecture



Example Results on Count Queries

	Total Error ↓			
Stock	Error	Error TP	Error FP	Error FN
Perfect IR	0.46 ± 0.07	0.46 ± 0.07	N/A	N/A
Noisy IR	0.77 ± 0.16	0.46 ± 0.07	0.31 ± 0.15	N/A
Dmg. IR	1.24 ± 0.32	0.46 ± 0.07	0.78 ± 0.32	N/A
Full	1.27 ± 0.17	0.42 ± 0.07	0.76 ± 0.13	0.09 ± 0.02

FTmodel				
Perfect IR	0.14 ± 0.01	0.14 ± 0.01	N/A	N/A
Noisy IR	0.22 ± 0.01	0.14 ± 0.01	0.08 ± 0.01	N/A
Dmg. IR	0.54 ± 0.05	0.14 ± 0.01	0.40 ± 0.05	N/A
Full	0.99 ± 0.06	0.11 ± 0.01	0.79 ± 0.06	0.09 ± 0.02



More details can be found in the following papers...

1. Giovanni Trappolini, Andrea Santilli, Emanuele Rodolà, Alon Y. Halevy, Fabrizio Silvestri: **Multimodal Neural Databases**. *SIGIR* 2023
2. Artsiom Sauchuk, James Thorne, Alon Y. Halevy, Nicola Tonello, Fabrizio Silvestri: **On the Role of Relevance in Natural Language Processing Tasks**. *SIGIR* 2022
3. James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, Alon Y. Halevy: **From Natural Language Processing to Neural Databases**. *VLDB* 2021
4. James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, Alon Y. Halevy: **Database reasoning over text**. *ACL/IJCNLP* 2021



Deep Learning

End of Lecture
08 - Transformers



SAPIENZA
UNIVERSITÀ DI ROMA

Fabrizio Silvestri