Modeling **structure**, learning **representations**

The two sides of language modeling

François Yvon

LISN — CNRS and Université Paris-Saclay

ALPS winter school

january 2023

Modeling syntax, probabilistically

A ubiquitous problem: comparing sentences

A: *I do not like to teach early in the morning*
B: *I do not like teaching early in the morning*

A better than B or B better than A ?

Optimal probabilistic decision rule

$$\text{A better than B} \Leftrightarrow \mathrm{P}(A) \geq \mathrm{P}(B)$$

$\mathrm{P}(\text{sentence})$ is a **language model**

Modeling syntax, probabilistically

A ubiquitous problem: comparing sentences

A: *I do not like to teach early in the morning*
B: *I do not like teaching early in the morning*

A better than B or B better than A ?

Ranking sentences in natural language generation tasks

	danser	
les enfants ont	*dansés*	*sur la plage*
	dansé	
	dansée	

Useful for speech transcription, OCR correction, spelling / grammar checking, , text generation, machine translation …

Optimal probabilistic decision rule

Modeling syntax, probabilistically

A ubiquitous problem: comparing sentences

A: *I do not like to teach early in the morning*
B: *I do not like teaching early in the morning*

A better than B or B better than A ?

Optimal probabilistic decision rule

$$\text{A better than B} \Leftrightarrow \mathrm{P}(A) \geq \mathrm{P}(B)$$

$\mathrm{P}(\text{sentence})$ is a **language model**

Modeling syntax, probabilistically

A ubiquitous problem: comparing sentences

A: *I do not like to teach early in the morning*
B: *I do not like teaching early in the morning*

A better than B or B better than A ?

Optimal probabilistic decision rule

$$\text{A better than B} \Leftrightarrow \mathrm{P}(A) \geq \mathrm{P}(B)$$

$\mathrm{P}(\text{sentence})$ is a **language model**

A Markov model for natural languages

The simplest sequence model: n-gram

$$P(w_1 \ldots w_L) = \prod_{i=1}^{L} P(w_i \mid w_1 \ldots w_{i-1}) \quad (1)$$

$$= \prod_{i=1}^{L} P(w_i \mid w_{i-n+1} \ldots w_{i-1}) \quad (2)$$

- (2): given recent history $h = w_{i-n+1} \ldots w_{i-1}$, remote words are unimportant.
- linguistically naïve, computationally efficient.
- very old model (Markov, Shannon)

A Markov model for natural languages

The simplest sequence model: *n*-gram

$$\mathrm{P}(w_1 \dots w_L) = \prod_{i=1}^{L} \mathrm{P}(w_i \mid w_1 \dots w_{i-1}) \quad (1)$$

$$= \prod_{i=1}^{L} \mathrm{P}(w_i \mid w_{i-n+1} \dots w_{i-1}) \quad (2)$$

n-gram text generation with ancestral sampling

$$w_1 \sim \mathrm{Unif}(W_1); w_2 \sim \mathrm{P}(W_2 \mid w_1); w_3 \sim \mathrm{P}(W_3 \mid w_2 w_1) \dots$$

Length L is not modelled: make sure to know when to stop sampling

A Markov model for natural languages

The simplest sequence model: *n*-gram

$$\mathrm{P}(w_1 \dots w_L) = \prod_{i=1}^{L} \mathrm{P}(w_i \mid w_1 \dots w_{i-1}) \tag{1}$$

$$= \prod_{i=1}^{L} \mathrm{P}(w_i \mid w_{i-n+1} \dots w_{i-1}) \tag{2}$$

Process more than words: letters, phones, morphs, clauses, speech turns, etc

An effective model of sequences without hierarchical structures

n-gram models are so simple, yet so tricky to estimate

Zipf's curse strickes back

Parameter estimation from running texts

Maximum likelihood estimates (with 2-word histories: trigrams)

$$P(w \mid uv) = \frac{c(uvw)}{\sum_{w'} c(uvw')} \tag{3}$$

$c()$ is the count function, $h = uv$ is the history

The art of language modeling

- with $100,000$ words, $100,000^3$ 3-gram counts, most of them 0
- build history classes ($uv \rightarrow h(uw)$) to keep models small
- building history classes ? the science of count smoothing [1992-2012]

n-gram models are so simple, yet so tricky to estimate

Zipf's curse strickes back

Parameter estimation from running texts

Maximum likelihood estimates (with 2-word histories: trigrams)

$$P(w \mid uv) = \frac{c(uvw)}{\sum_{w'} c(uvw')} \tag{3}$$

$c()$ is the count function, $h = uv$ is the history

The art of language modeling

- with $100,000$ words, $100,000^3$ 3-gram counts, most of them 0
- build history classes ($uv \rightarrow h(uw)$) to keep models small
- building history classes ? the science of count smoothing [1992-2012]

The art of smoothing

Smoothing is some black magic: estimate the probability of unseen events

"+1"-smoothing (Laplace):

$$P_{+1}(w \mid h) = \frac{c(hw) + 1}{|\mathcal{V}| + c(h)}$$

Variant (add $+\alpha$, with α a parameter). Alternatives

- Linear interpolation
- Clustering: building word / history classes
- Discounting techniques (Katz back-off, Good-Turing back-off, Knesser-Ney smoothing)
- Non-parametric Bayesian models

A side effect of smoothing

Before smoothing, many word sequences have a null probability;
After smoothing, all word sequences have probability > 0.

The art of smoothing

Smoothing is some black magic: estimate the probability of unseen events

"+1"-smoothing (Laplace):

$$P_{+1}(w \mid h) = \frac{c(hw) + 1}{|\mathcal{V}| + c(h)}$$

Variant (add $+\alpha$, with α a parameter). Alternatives

- Linear interpolation
- Clustering: building word / history classes
- Discounting techniques (Katz back-off, Good-Turing back-off, Knesser-Ney smoothing)
- Non-parametric Bayesian models

A side effect of smoothing

Before smoothing, many word sequences have a null probability;
After smoothing, all word sequences have probability > 0.

Implementations

- n-grams, with all bells and whistles: SRILM ☞ www.speech.sri.com/projects/srilm/
- n-grams, scalable & extremely efficient: KenLM ☞ https://github.com/kpu/kenlm
- n-grams, integrate with finite-state tools ☞ http://www.opengrm.org/
- n-grams NLTK - basic implementation ☞ http://nltk.org

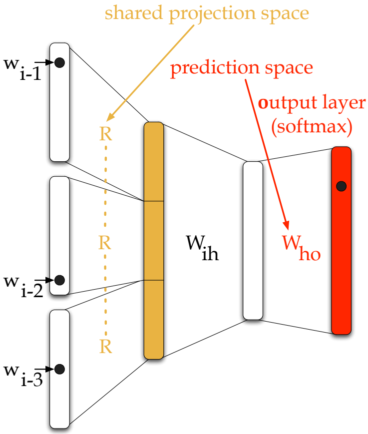
ngram LMs: a summary

- n-grams LM are computationally efficient models of word sequences
- scales well to very large corpus
- discounting techniques are key to performance
- unknown words a big issue

Still useful ? Finite-state distillation of neural LMs

Feed-forward language models [Bengio et al., 2003]

Continuous space *n*-gram models



$$\boldsymbol{i} = [w_{i-1}^T R, w_{i-2}^T R, w_{i-3}^T R]$$

$$\boldsymbol{h} = \boldsymbol{i}^T W_{ih} + \boldsymbol{b}_{ih}$$

$$\boldsymbol{o} = \tanh(\boldsymbol{h})^T W_{ho} + \boldsymbol{b}_{ho}$$

$$\mathrm{P}(w_i \mid w_{i-3}, w_{i-2}, w_{i-1}) = \frac{\exp \boldsymbol{o}[w_i]}{\sum_w \exp \boldsymbol{o}[w]}$$

What neural language modeling does:

1. encodes history as $\phi(w_{i-3}, w_{i-2}, w_{i-1})$
2. compares $\phi(w_{i-3}, w_{i-2}, w_{i-1})$ and $R'(w_i)$

Feed-forward language models [Bengio et al., 2003]

Continuous space *n*-gram models

Training FFLMs - maximize log-likelihood / cross-entropy

$$\begin{aligned}\boldsymbol{\theta}^* = [\mathbf{R}, \mathbf{W}_{ih}, \boldsymbol{b}_i, \mathbf{W}_{ho}, \boldsymbol{b}_o] &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_t \log \frac{\exp \boldsymbol{o}[w_t]}{\sum_w \exp \boldsymbol{o}[w]} \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_t \boldsymbol{o}[w_t] - \log(\sum_w \exp \boldsymbol{o}[w])\end{aligned}$$

- $\mathrm{softmax}(\boldsymbol{x}) = [\frac{\exp \boldsymbol{x}[i]}{\sum_k exp \boldsymbol{x}[k]}]$ computes dense distributions
- embeddings $\mathbf{R}$ are learned
- computationally demanding (softmax layer)
- superior to discrete (n-gram) LMs across the board [Schwenk, 2007, Le et al., 2012]

Feed-forward language models [Bengio et al., 2003]

Continuous space *n*-gram models

word (freq.)	*model*	*5 nearest neighbors*
is	standard	was are were been remains
900, 350	1 vector init. (*)	was are be were been
conducted	standard	undertaken launched $270,900 Mufamadi 6.44-km-long
18, 388	1 vector init. (*)	pursued conducts commissioned initiated executed
Cambodian	standard	Shyorongi $3,192,700 Zairian depreciations teachers'
2, 381	1 vector init. (*)	Danish Latvian Estonian Belarussian Bangladeshi
automatically	standard	MSSD Sarvodaya $676,603,059 Kissana 2,652,627
1, 528	1 vector init. (*)	routinely occasionally invariably inadvertently seldom
Tosevski	standard	$12.3 Action,3 Kassouma 3536 Applique
34	1 vector init. (*)	Shafei Garvalov Dostiev Bourloyannis-Vrailas Grandi
October-12	standard	39,572 anti-Hutu $12,852,200 non-contracting Party's
8	1 vector init. (*)	March-26 April-11 October-1 June-30 August4
3727th	standard	Raqu Tatsei Ayatallah Mesyats Langlois
1	1 vector init. (*)	4160th 3651st 3487th 3378th 3558th

(*) *1 vector init: share parameters R and W_{ho} during init.*

[Examples from [Le et al., 2010]]

FFLMs induce similarities between histories and between words

The infamous < unk >nown word

"Closed world" assumptions

- The support of LM: a fixed vocab $\mathcal{V}$. Sentences with unknowns have 0 probability.
- The support of LM: a fixed vocab $\mathcal{V} \cup \{ <\text{unk}> \}$.
  Estimation: all words $\notin \mathcal{V}$ are unked [makes < unk > very likely].
- Variant: consider classes of < unk > (proper names, numbers, etc).

"Open" world models with subword units: morphemes, char ngrams, bytes, pixels

- morph-based LM: require morphogical analysis, < unk > still possible
- letters: no more unknown words - unknown symbols instead ?
- a mixture of words and letters

Caveats

- Shorter units require longer histories [estimation problems]
- And imply longer sentences [computational problems]

The infamous $<$ unk $>$nown word

"Closed world" assumptions

- The support of LM: a fixed vocab $\mathcal{V}$. Sentences with unknowns have 0 probability.
- The support of LM: a fixed vocab $\mathcal{V} \cup \{ < \text{unk} > \}$.
  Estimation: all words $\notin \mathcal{V}$ are unked [makes $<$ unk $>$ very likely].
- Variant: consider classes of $<$ unk $>$ (proper names, numbers, etc).

"Open" world models with subword units: morphemes, char ngrams, bytes, pixels

- morph-based LM: require morphogical analysis, $<$ unk $>$ still possible
- letters: no more unknown words - unknown symbols instead ?
- a mixture of words and letters

Caveats

- Shorter units require longer histories [estimation problems]
- And imply longer sentences [computational problems]

The infamous < unk >nown word

"Closed world" assumptions

- The support of *LM*: a fixed vocab $\mathcal{V}$. Sentences with unknowns have 0 probability.
- The support of *LM*: a fixed vocab $\mathcal{V} \cup \{ < \text{unk} > \}$.
  Estimation: all words $\notin \mathcal{V}$ are unked [makes < unk > very likely].
- Variant: consider classes of < unk > (proper names, numbers, etc).

"Open" world models with subword units: morphemes, char ngrams, bytes, pixels

- morph-based LM: require morphogical analysis, < unk > still possible
- letters: no more unknown words - unknown symbols instead ?
- a mixture of words and letters

Caveats

- Shorter units require longer histories [estimation problems]
- And imply longer sentences [computational problems]

Subword units in language models: BPEs, wordpieces, etc

Byte pair encoding: *N* deterministic merge operations

1. Make symbol map (greedy)
   `Repeat till done: merge most frequent bigram into new compound symbol`
2. Encode (greedy)
   `split each word into compound symbols`

Subword units in language models: BPEs, wordpieces, etc

Byte pair encoding: *N* deterministic merge operations

1. Make symbol map (greedy)
   `Repeat till done: merge most frequent bigram into new compound symbol`
2. Encode (greedy)
   `split each word into compound symbols`

Example from [Sennrich et al., 2016]

$L = \{$ *lower*, *lowest*, *newer*, *wider*, *wide* $\}$

e	r#	3	[er#]	[lo]	w	2	[low]
l	o	2	[lo]	w	i	2	[wi]

Segmentations: [low]+ [er#], [low]+ e+ s+ [t#], n+ e+ w+ [er#], [wid]+ [er#], [wid]+ [e#]

☞ https://github.com/rsennrich/subword-nmt

Subword units in language models: BPEs, wordpieces, etc

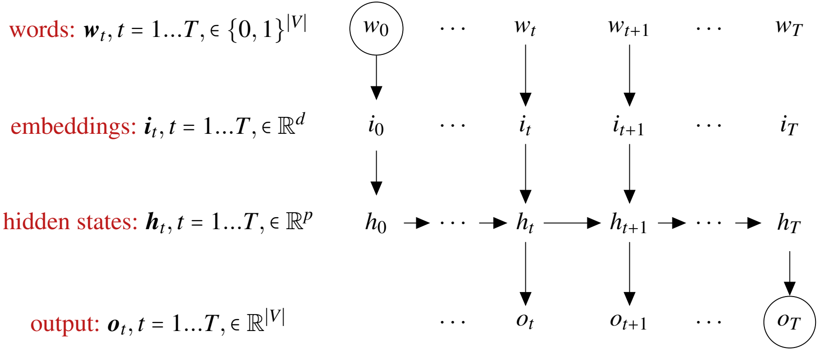
Byte pair encoding: *N* deterministic merge operations

1. Make symbol map (greedy)
   `Repeat till done:  merge most frequent bigram into new compound symbol`
2. Encode (greedy)
   `split each word into compound symbols`

- better solutions: SentencePiece (unigram model), wordPiece
- also regularization through sampling
- handles multilingual vocabularies [use with care]
- segmentation is a latent variable
  $\mathrm{P}(w \mid h)$ requires summing over segmentations [Cao and Rimell, 2021]

Recurrent Neural Networks as LMs [Mikolov et al., 2010]

From finite to infinite contexts



$$
\begin{aligned}
\boldsymbol{i}_t &= [w_t^T R] \\
\boldsymbol{h}_t &= \tanh(\boldsymbol{i}_t^T W_{ih} + \boldsymbol{h}_{t-1}^T W_{hh} + \boldsymbol{b}_{ih}) \\
\boldsymbol{o} &= \boldsymbol{h}^T W_{ho} + \boldsymbol{b}_{ho}
\end{aligned}
$$

Recurrent Neural Networks as LMs [Mikolov et al., 2010]

From finite to infinite contexts

words: $\boldsymbol{w}_t, t = 1...T, \in \{0,1\}^{|V|}$

embeddings: $\boldsymbol{i}_t, t = 1...T, \in \mathbb{R}^d$

hidden states: $\boldsymbol{h}_t, t = 1...T, \in \mathbb{R}^p$

output: $\boldsymbol{o}_t, t = 1...T, \in \mathbb{R}^{|V|}$

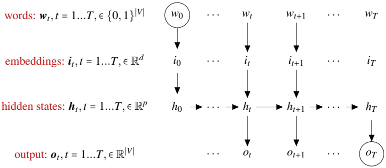
$$\begin{tikzcd}
w_0 \arrow[d] & \cdots & w_t \arrow[d] & w_{t+1} \arrow[d] & \cdots & w_T \\
i_0 \arrow[d] & \cdots & i_t \arrow[d] & i_{t+1} \arrow[d] & \cdots & i_T \\
h_0 \arrow[r] & \cdots \arrow[r] & h_t \arrow[r] \arrow[d] & h_{t+1} \arrow[r] \arrow[d] & \cdots \arrow[r] & h_T \arrow[d] \\
 & \cdots & o_t & o_{t+1} & \cdots & o_T
\end{tikzcd}$$

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_t \log \boldsymbol{o}[w_t] - \log(\sum_w \exp \boldsymbol{o}[w])$$

$$\mathrm{P}(w_{t+1} = k \mid w_{\leq t}; \theta_{LM}) = \operatorname{softmax}(\boldsymbol{o}_t = W^{ho}\boldsymbol{h}_t + \boldsymbol{b}^o)[k]$$

Recurrent Neural Networks as LMs [Mikolov et al., 2010]

From finite to infinite contexts



- train with word prediction objective and cross-entropy loss
- more complex cells ($(\boldsymbol{w}_t, \boldsymbol{h}_t) \rightarrow \boldsymbol{h}_{t+1}$): GRUs, LSTMs
- same issues as FFLMs with $\mathrm{softmax}$, same solutions apply
- stack several hidden layers $\boldsymbol{h}_t^k = f(\boldsymbol{h}_{t-1}^k, \boldsymbol{h}_t^{k-1})$: biRNNs, etc.
- $\boldsymbol{h}_T$: compact, effective sentence representation for $w_1 \ldots w_T$
- backwards processing computes $\bar{\boldsymbol{h}}_{-1}$
- $[\boldsymbol{h}_T, \bar{\boldsymbol{h}}_{-1}]$ a better representation: text classification, etc.
- $[\boldsymbol{h}_t, \bar{\boldsymbol{h}}_t]$ represents word w_t and its context

RNNs as "pure" encoders

An approach to sentence classification

$\boldsymbol{h}_T = \mathrm{RNN}(w_1 \dots w_T)$ encodes a variable-length sentence in a fixed-length vector.

Decision rule for TC tasks, mapping sentences to classes (sentiment / opinion mining, stance detection, entailment, etc): $w_1 \dots w_T \rightarrow y$

$$\mathrm{P}(y = 1 \,|\, w_1 \dots w_T) = \sigma(\boldsymbol{W}^T \boldsymbol{h}_T + b)$$
$$\theta^* = \operatorname{argmax} \sum_i \log \mathrm{P}(y^{(i)} \,|\, w_1^{(i)} \dots w_{T^{(i)}}^{(i)})$$

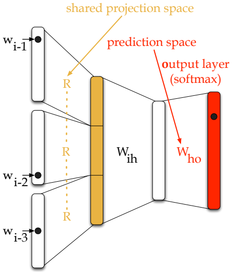
Improved classification results with backward encoding, multiple layers etc.

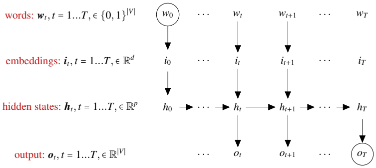
Also works with pairs of sentences (textual entailment, discourse modeling):

$$\boldsymbol{h}_T^w = \mathrm{RNN}(w_1 \dots w_T), \boldsymbol{h}_S^v = \mathrm{RNN}(v_1 \dots v_S)$$
$$h^{v,w} = \boldsymbol{h}_T^w \oplus \boldsymbol{h}_S^v \text{ or } h^{v,w} = [\boldsymbol{h}_T^w, \boldsymbol{h}_S^v] \text{ or } \dots$$

Self-attention: cooking complex histories

FFLMs / RNNs predict next word based on a continuous representation of the history with trained similarity





- FFLMs use a fixed-size history
- RNNs use a pre-defined structure: all words matter, recent words matter more than distant words.

New architecture: SAN-LM (aka Transformers) [Vaswani et al., 2017] trained to learn which history words matter.

Self-attention: cooking complex histories

Heads: parameterized computing modules

$$\text{Query parameters: } \mathbf{H}_q \in \mathbb{R}^l \times \mathbb{R}^d$$
$$\text{Key parameters: } \mathbf{H}_k \in \mathbb{R}^l \times \mathbb{R}^d$$
$$\text{Value parameters: } \mathbf{H}_v \in \mathbb{R}^o \times \mathbb{R}^d$$

Computing attention with heads

Heads linearly transform matrices $\boldsymbol{I} \in \mathbb{R}^d \times \mathbb{R}^T$ into matrices $\boldsymbol{O}$ in $\mathbb{R}^o \times \mathbb{R}^T$.

$$\text{transform input matrix for words: } \mathbf{J} = \boldsymbol{H}_q \times \mathbf{I} \in \mathbb{R}^l \times \mathbb{R}^T$$
$$\text{transform input matrix for contexts: } \mathbf{K} = \boldsymbol{H}_k \times \mathbf{I} \in \mathbb{R}^l \times \mathbb{R}^T$$
$$\text{transform input matrix for outputs: } \mathbf{L} = \boldsymbol{H}_v \times \mathbf{I} \in \mathbb{R}^o \times \mathbb{R}^T$$
$$\text{compute similarities words/context: } \mathbf{D} = \mathbf{J} \times \boldsymbol{K}^T \in \mathbb{R}^T \times \mathbb{R}^T$$
$$\text{compute linear weights: } \tilde{\mathbf{D}} = \text{softmax}(\frac{\mathbf{D}}{\sqrt{d}}) \in [0,1]^T \times [0,1]^T \text{ columnwise}$$
$$\text{linear combination of cols: } \boldsymbol{O} = \tilde{\boldsymbol{D}} \times \mathbf{L} \in \mathbb{R}^T \times \mathbb{R}^o$$

Self-attention: cooking complex histories

Using several Heads [in parallel]: MultiHeads

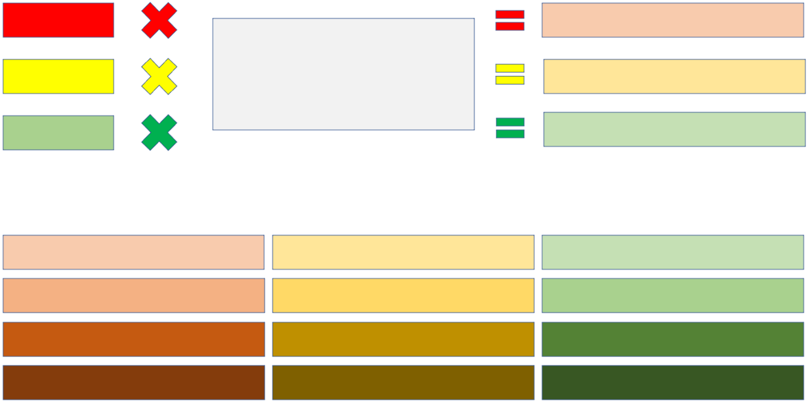
- one Head : $\mathbf{I}(d \times T) \rightarrow \mathbf{O}(o \times T)$
- k Heads : $\mathbf{I}(d \times T) \rightarrow [\mathbf{O}_1, \ldots, \mathbf{O}_k](ko \times T)$

Using multiple layers of MultiHeads

- compute with k Heads : $\mathbf{I}(d \times T) \rightarrow \mathbf{O} = [\mathbf{O}_1 \ldots \mathbf{O}_k](ko \times T)$
- enable residual (direct) connections $\mathbf{O'} = \mathbf{O} + \mathbf{I}$
- pass $\mathbf{O'}$ through a "linear" layer $\mathbf{O''} = \mathbf{O'} + \mathbf{W}' \times \mathrm{RELU}(\mathbf{WO})$, with $\mathbf{O''} \in \mathbb{R}^{(d \times T)}$
- make layers and sublayers comparable through layer normalization (substract mean, divide by stddev)
- stack multiple layers $\mathbf{I}_1 \rightarrow \mathbf{I}_2 \rightarrow \mathbf{I}_3 \rightarrow \mathbf{I}_4 \ldots$

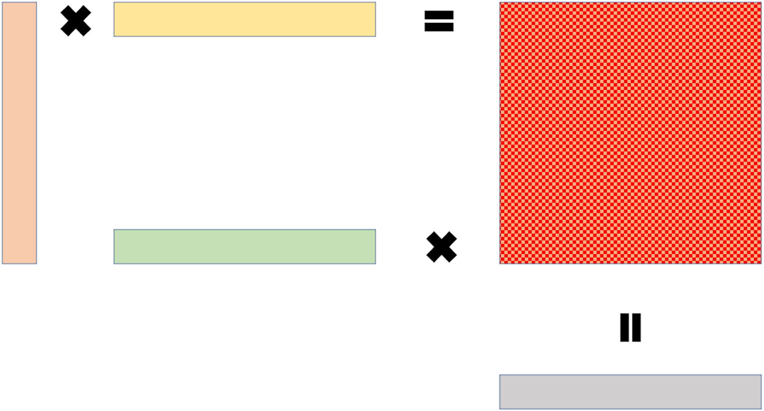
Typical values: 8 Heads of output dimension $o = 64$, $6 - 12$ layers of heads of dimension 512.

Self-attention: cooking complex histories



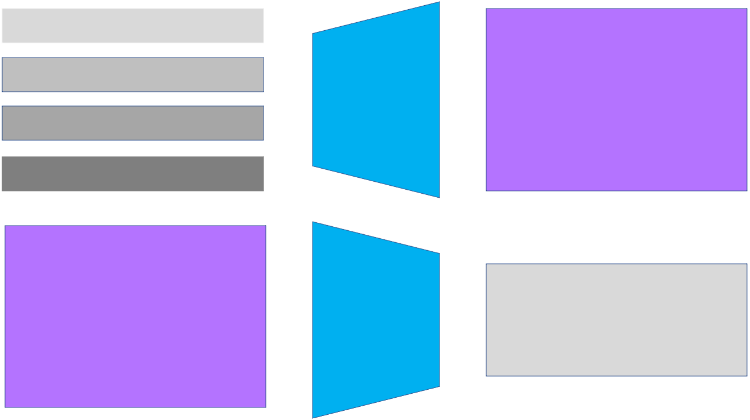
Computes **J**, **K**, **L** for 4 Heads

Self-attention: cooking complex histories



Computes $\tilde{\mathbf{D}}$ and $\mathbf{K}$

Self-attention: cooking complex histories



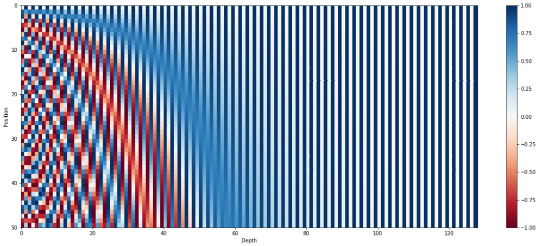
Computes output

Self-attention: cooking complex histories

The initial encoding layer

Input = $w_1 \dots w_T$, columns in $\mathbf{I_1}$ combine word embeddings and positional encodings.

$$\left\{ \begin{array}{l} \mathbf{P}[2i,t] = \sin(t/10000^{2i/d}) \\ \mathbf{P}[2i+1,t] = \cos(t/10000^{2i/d}) \end{array} \right. \in \mathbf{P} \in \mathbb{R}^d \times \mathbb{R}^T$$



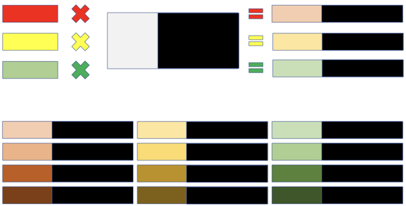
Also trainable PEs, relative PEs [Raffel et al., 2020], Alibi PEs [Press et al., 2022] etc.

Self-attention: cooking complex histories

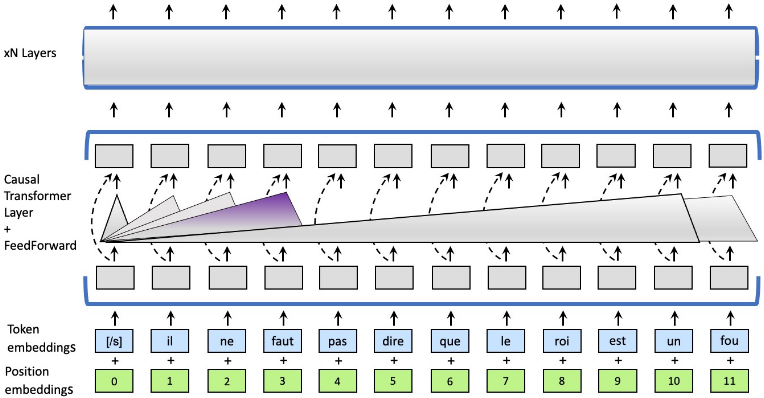
Masking & Causality in Language Modeling

The past should not depend on the future.
Causal / Masked LMs mask ([_]) future time steps

context	predicts
$< s > w_1 \ldots w_t [\_][\_] \ldots$	w_{t+1}
$< s > w_1 \ldots w_{t+1} [\_][\_] \ldots$	w_{t+2}



Self-attention: cooking complex histories



Incremental computation: fully causal transformer

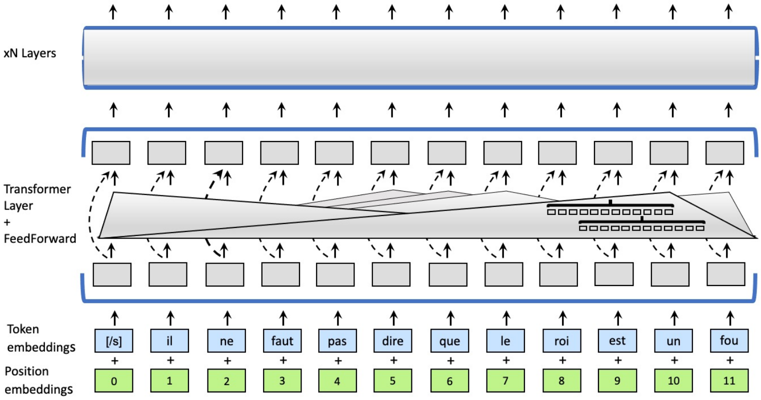
☞ graphics, explanations https://jalammar.github.io/illustrated-gpt2/

Self-attention: cooking complex histories

Output layer and prediction (MLM)

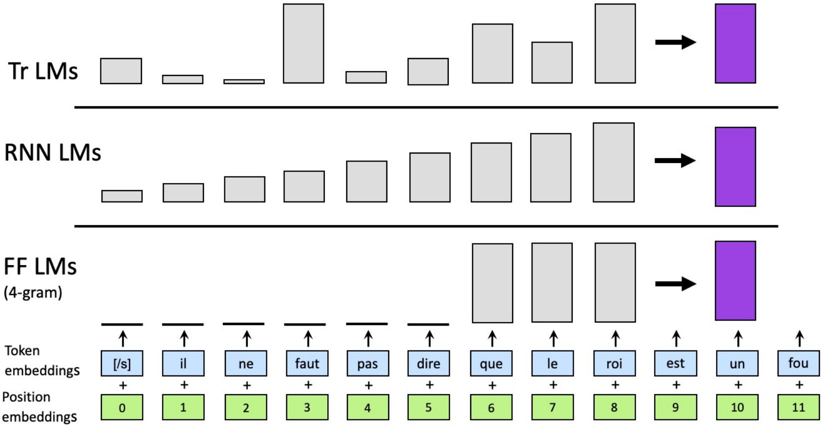
- pack hidden states $[\mathbf{I}_K[t], \mathbf{I}_{K-1}[t], \mathbf{I}_{K-2}[t], ..]$ into $\sum_u \theta_u \boldsymbol{I}_u[t]$
- project into $\mathbb{R}^{|\mathcal{V}|}$ to get logits: $g(\mathbf{W}_o(\sum_u \theta_u \boldsymbol{I}_u[t]) + \boldsymbol{b}_o)$
- use $\mathrm{softmax}$ to predict next word w_{t+1}
- compute loss $\ell_{ce} = -log\, \mathrm{P}(w^* \,|\, w_{<t}) = \mathrm{KL}(\mathbb{I}(w^*) \,||\, \mathrm{P}(W \,|\, w_{<t}))$
- back-prop gradient

Self-attention: cooking complex histories

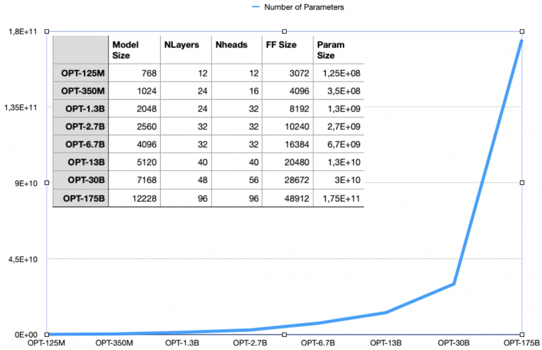


Non causal transformers are contextual representation learners

Self-attention: cooking complex histories



From language models to “foundational” models



	Model Size	NLayers	Nheads	FF Size	Param Size
OPT-125M	768	12	12	3072	1,25E+08
OPT-350M	1024	24	16	4096	3,5E+08
OPT-1.3B	2048	24	32	8192	1,3E+09
OPT-2.7B	2560	32	32	10240	2,7E+09
OPT-6.7B	4096	32	32	16384	6,7E+09
OPT-13B	5120	40	40	20480	1,3E+10
OPT-30B	7168	48	56	28672	3E+10
OPT-175B	12228	96	96	48912	1,75E+11

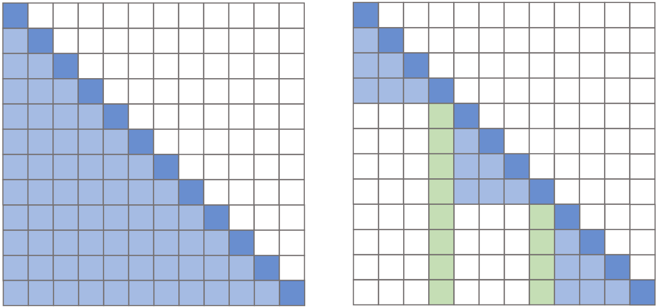
Dimensions of very large language models (OPT from Meta)

The GPT-style family: GPT1-3 (OpenAI), Megatron (Nvidia), Gopher / Chinchilla (DeepMind), HyperClova (NaverLabs), GPT-J (EuletherAI), OPT (Meta), Bloom (BigScience)

From language models to “foundational” models

Algorithmic issues [Tay et al., 2022]:

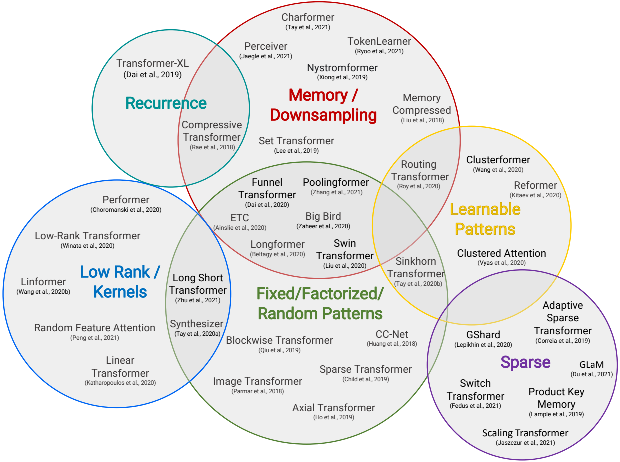
- extends context length T: costs $O(T^2)$ + positional embeddings + gradients storage
- increase depth - costs gradients storage



full attention   local attention + sentinels

- also: random or hierarchical sparse attention masks, parameter sharing, approximate dot product computation, etc

From language models to “foundational” models



A bestiary of efficient transformers [Tay et al., 2022]

Large Language Models are *very* powerful

Natural language processing tasks, such as question answering, machine translation, reading comprehension and summarization, are typically approached with supervised learning on task-specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision (...) [Radford et al., 2019]

- ✔ scoring model for disambiguation tasks (as any language model)
- ✔ continuous representations of sentences and paragraphs: towards transfert learning
- ✔ simple and effective generation mechanism
  $\mathrm{Gen}(w_1 \ldots w_T) = \mathrm{argmax}_{w_{t+1} \ldots w_T} \mathrm{P}(w_{t+1} \ldots w_T \mid w_1 \ldots w_t)$
- ✔ use generation as multitask processing with prompts / instructions

Large Language Models are *very* powerful

Natural language processing tasks, such as question answering, machine translation, reading comprehension and summarization, are typically approached with supervised learning on task-specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision (...) [Radford et al., 2019]

Prompt (ANLI)	The Gold Coast Hotel & Casino is a hotel and casino located in Paradise, Nevada. This locals' casino is owned and operated by Boyd Gaming. The Gold Coast is located one mile west of the Las Vegas Strip on West Flamingo Road. It is located across the street from the Palms Casino Resort and the Rio All Suite Hotel and Casino. Question: The Gold Coast is a budget-friendly casino. True, False, or Neither?
Answer (OK)	Neither
Answer (KO)	True
Answer (KO)	False

Prompt (PIQA)	How to apply sealant to wood ?
Answer (OK)	Using a brush, brush on sealant onto wood until it is fully saturated with the sealant.
Answer (KO)	Using a brush, drip on sealant onto wood until it is fully saturated with the sealant.

Prompt (COPA)	My body cast a shadow over the grass because
Answer (OK)	the sun was rising.
Answer (KO)	the grass was cut.

☞ examples from Radford et al. [2019]

Also: summarization / compression, machine translation (!), arithmetic (!!)

Current challenges for language modeling

1. find the right modeling units (between char-based and word-based)
2. model very long range context-dependency beyond syntax: [Dai et al., 2019]
   repetitions, style, discourse consistency, etc.
3. towards better text generation through better searching
4. improve training efficiency / scalability; reduce the carbon emissions of large-scale LMs
5. (continuous) online adaptation of LMs
6. mitigate the biases of "stochastic parrots" (also: privacy issues, etc) [Bender et al., 2021]

The race for size not finished - from GPT2 (1.5b) to GPT3 (175b) to T5 to GShard (600b), Switch-C (1.6t) to PALM and beyond

Implementations and models available online

- ☞ https://transformer.huggingface.co/
- ☞ https://github.com/facebookresearch/fairseq
- ☞ https://www.tensorflow.org/hub?hl=en
- ☞ https://github.com/openai/gpt-2

Bibliography I

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pages 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003. ISSN 1532-4435.

Kris Cao and Laura Rimell. You should evaluate your language model on marginal likelihood over tokenisations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2104–2114, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.161. URL https://aclanthology.org/2021.emnlp-main.161.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *CoRR*, abs/1901.02860, 2019. URL http://arxiv.org/abs/1901.02860.

Bibliography II

Hai Son Le, Alexandre Allauzen, Guillaume Wisniewski, and François Yvon. Training continuous space language models: Some practical issues. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 778–788, Cambridge, MA, 2010. URL http://www.aclweb.org/anthology/D/D10/D10-1076.

Hai-Son Le, Alexandre Allauzen, and François Yvon. Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–48, Montréal, Canada, June 2012. Association for Computational Linguistics.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In Takao Kobayashi, Keikichi Hirose, and Satoshi Nakamura, editors, *Eleventh Annual Conference of the International Speech Communication Association (InterSpeech)*, pages 1045–1048, Makuhari, Chiba, Japan, 2010. URL https://www.isca-speech.org/archive/archive_papers/interspeech_2010/i10_1045.pdf.

Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *Proceedings of the International Conference on Learning Representations*, ICLR, 2022. URL https://arxiv.org/abs/2108.12409.

Bibliography III

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019. URL https://openai.com/blog/better-language-models/.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

Holger Schwenk. Continuous space language models. *Computer, Speech and Language*, 21(3): 492–518, 2007. ISSN 0885-2308.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. URL https://www.aclweb.org/anthology/P16-1162.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Comput. Surv.*, apr 2022. ISSN 0360-0300. doi: 10.1145/3530811. URL https://doi.org/10.1145/3530811. Just Accepted.

Bibliography IV

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.