How Do Transformers Learn Topic Structure: Towards a Mechanistic Understanding



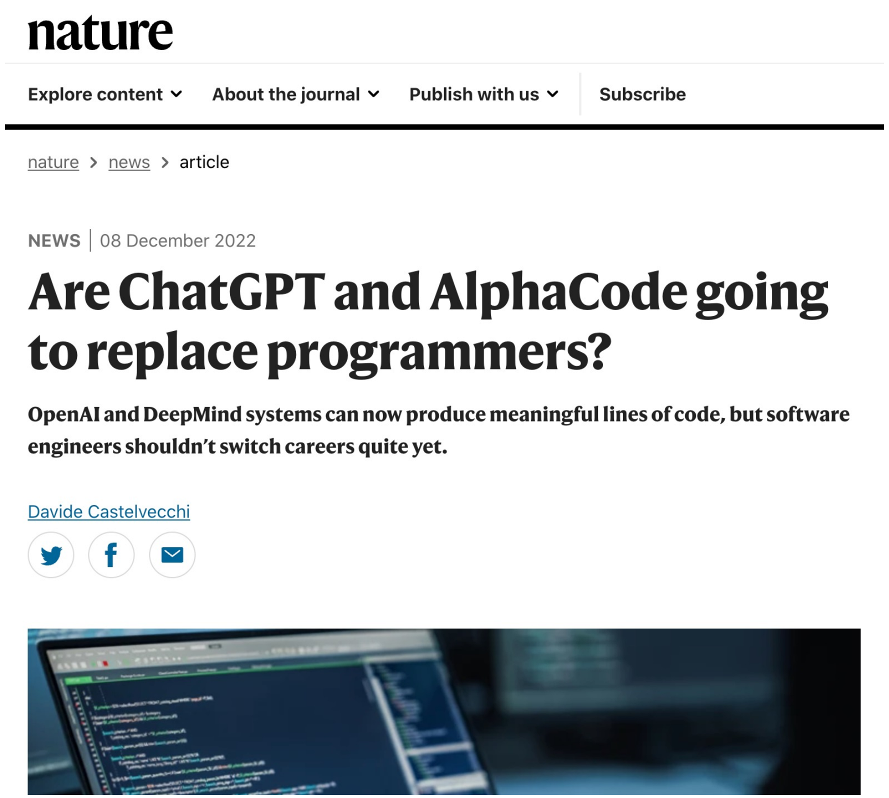
Yuchen Li
(CMU)

Yuanzhi Li
(CMU & Microsoft)

Andrej Risteski
(CMU)

https://arxiv.org/abs/2303.04245 (to appear in ICML 2023)

Background: applications of transformers


nature

Explore content ⌄ About the journal ⌄ Publish with us ⌄ Subscribe

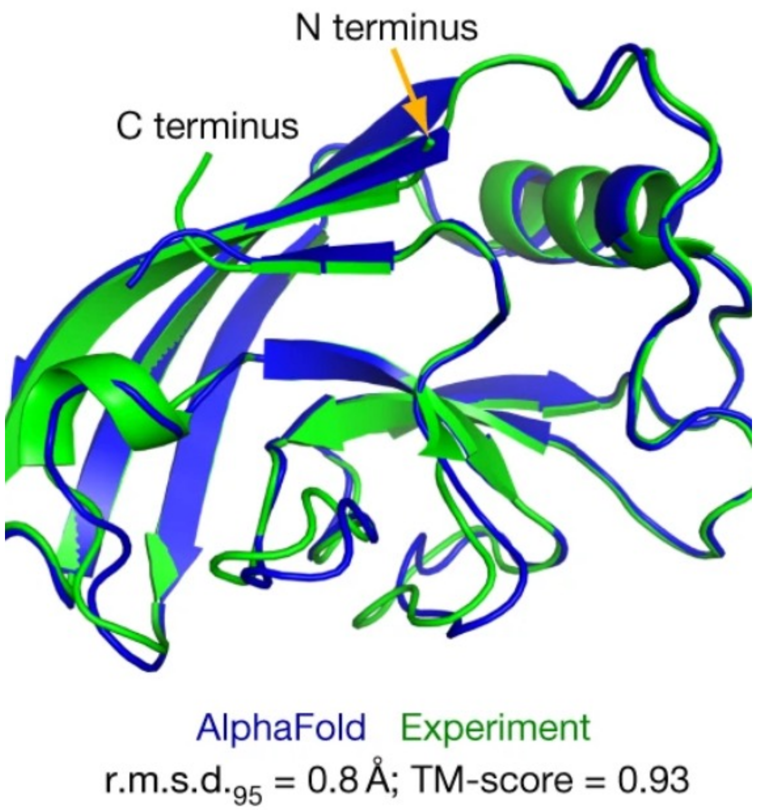
nature > news > article

NEWS | 08 December 2022

Are ChatGPT and AlphaCode going to replace programmers?

OpenAI and DeepMind systems can now produce meaningful lines of code, but software engineers shouldn’t switch careers quite yet.

Davide Castelvecchi



natural & programming languages

vision

protein structure prediction

Photos: nature.com; Alexey Dosovitskiy et al. An image is worth 16x16 words

Our methodology for theoretical understanding

- Real data are messy and complex
  - Language data: semantics (meaning), syntax (grammar), ...
  - All these aspects affect the behavior of trained models
- To study them in more formal manner
  - Focus on one of these aspects by studying some simple synthetic setting
  - Examples: topic model, regular languages, probabilistic context-free grammars, …
  - Ablate away some other aspects of language
  - Benefits: control variables, single out each factor
  - Agenda of this line of research: progressively study more realistic data distributions
- By contrast: empirical probing works
  - Intuitive, but difficult to state the results formally
  - No canonical way to probe especially for attention

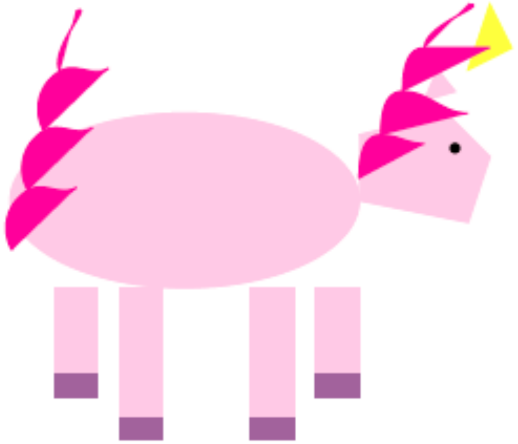
Characterizing the optimization process is crucial for theoretical understanding of transformers

Many prior theoretical works
- Turing completeness
  - Pérez et al. 2021
- Universal approximation
  - Yun et al. 2020
- Can represent some algorithms
  - Yao et al. 2021

?

GPT-4:

```
class Solution(object):
    def numberOfPaths(self, grid, k):
        """
        :type grid: List[List[int]]
        :type k: int
        :rtype: int
        """
        # Define MOD as 10**9 + 7
        MOD = 10**9 + 7
        # Get the dimensions of the grid
        m = len(grid)
        n = len(grid[0])
```



But transformers can also learn “shortcuts” even on simpler tasks (Liu et al. 2023[2])

The missing link:
What is actually learned through optimization?

Goal: to understand the empirical effectiveness of transformers[1]

1. Images taken from: Sébastien Bubeck et al, 2023, Sparks of AGI: Early experiments with GPT-4
2. Bingbin Liu et al, 2023, Transformers learn shortcuts to automata

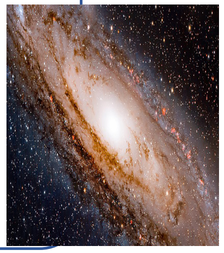
Overview of our results

- **Data:** topic modeling: Latent Dirichlet allocation (LDA)
  - Captures a simple type of semantics (based on co-occurrence) in natural languages
- **Model architecture:** a single-layer transformer (no FFN, no layer norm)
- **Pre-training task:** masked language modeling
- Our analysis involves a combination of training process and loss optima
- **Main result 1** (optimal word embedding, informal)
  - If everything other than embedding layer is frozen
  - The inner product of the embeddings of a pair of words is larger when the words belong to the same topic, and smaller when they belong to different topics
- **Main result 2** (optimal self-attention, informal)
  - If token embeddings are frozen to be one-hot vectors
  - The attention score between a pair of words is larger when the words belong to the same topic, and smaller when they belong to different topics
- Theory is also predictive of multi-layer multi-head transformers trained on Wikipedia data

Data: topic model

- “Topic” is a simple aspect of semantics in natural language[1]
  - document = mixture of topics (bag of words, i.e. no word order)
  - topic = probability distribution of words

ski	3			3%			0	
trail	5			6%			0	
ice	1			1%			0.1%	
sun	2	=	0.7	1%	+	0.2	4%	+ ...
stars	1			0.1%			5%	
transformer	0			0			0	
...	...			...			...	



1. David Blei, et al, 2003, Latent Dirichlet Allocation (LDA)
2. Figure idea credit to Sanjeev Arora’s talk in 2014

Data: topic model

- T topics: {1, ..., T}, each containing v words
  - Disjoint topics: no overlapping words
- dataset = collection of documents
- document = sequence of words $w_1, \ldots, w_N$, generated by
  - First uniformly randomly choosing τ distinct topics from {1, ..., T}
  - For each n in 1...N, generate w_n by
    - Uniformly randomly choosing one of these τ topics
    - Uniformly randomly choosing one word of this chosen topic
  - Our theory studies the long-doc regime: $N \to \infty$
- This is a special case of a Latent Dirichlet Allocation (LDA) model

Training loss: masked language modeling

- **Original:** Andrew Carnegie famously said, "My heart is in the work."
- **Masked:** Andrew Carnegie famously [MASK], "My heart is apple the [MASK]."
- **Goal:** masked sentence -> model -> original sentence
- More formally, given original document $w = w_1, \dots, w_N$
- Select a constant proportion p_m (e.g. 15%) of masked positions
  - Masked document $w' = w_1', \dots, w_N'$
  - If position i is not select above, then $w_i' = w_i$
  - If position i is selected, then w_i' can be
    - The correct word w_i with probability p_c
    - A random word with probability p_r
    - The [MASK] token with probability $1 - p_c - p_r$

Training loss: masked language modeling

- **Original:** Andrew Carnegie famously said, "My heart is in the work."
- **Masked:** Andrew Carnegie famously [MASK], "My heart is apple the [MASK]."
- **Predicted:** Andrew ? famously ?, "My heart is ? the ?."

prediction $\hat{y} =$

Carnegie	0.05
Webber	0.09
Ng	0.11
Jackson	0.08
Johnson	0.08
…	…

label $y =$ Carnegie

loss at that position $l(\hat{y}, y)$

training loss $\sum l(\hat{y}, y)$ for all selected positions

Model architecture: single-layer transformer

- Given input representation $Z \in \mathbb{R}^{d \times N}$

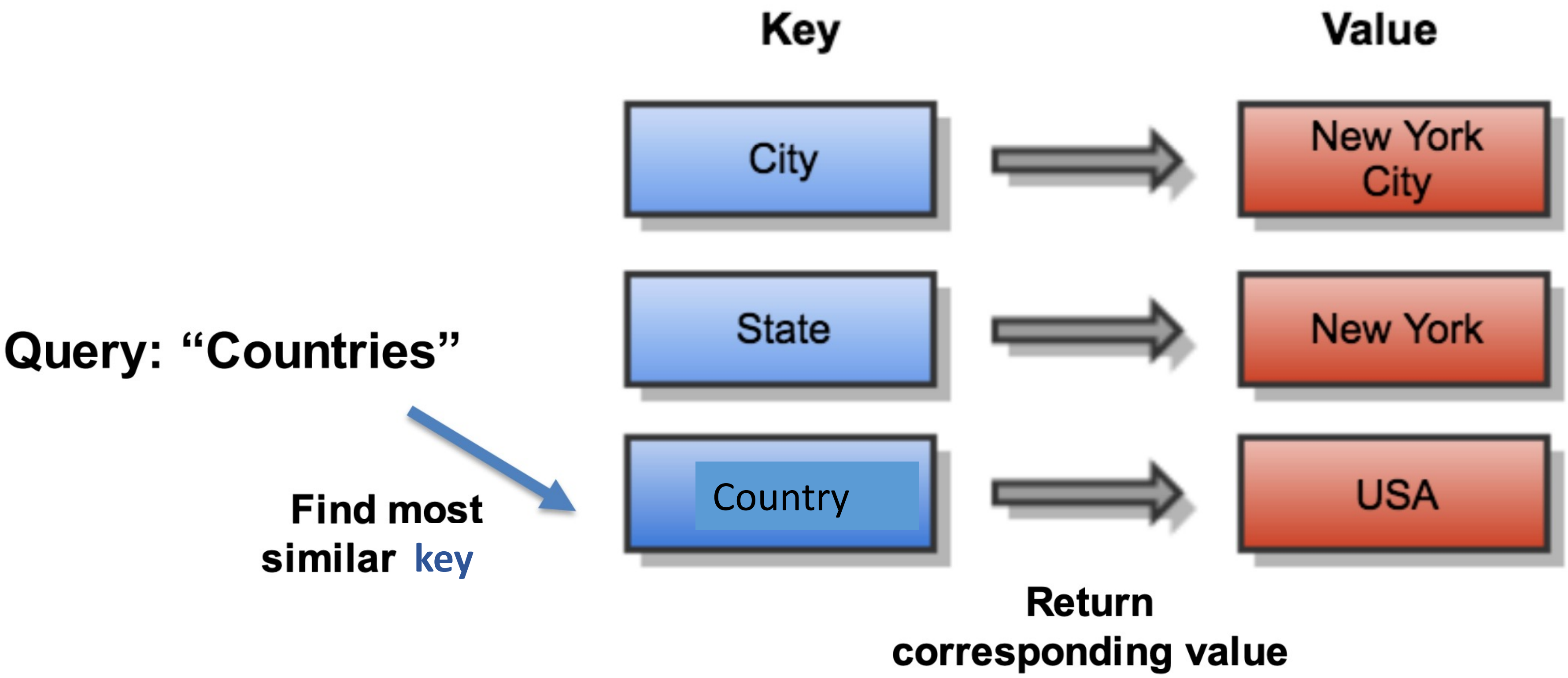
$$f(Z) = W^{pred}(W^V Z)\, Attention(Z) + b^{pred} \in \mathbb{R}^{D \times N}$$

- D: vocabulary size
- d: embedding dimension
- (usually: $d < D$)
- N: sequence length

- $Attention(Z)$ is the core of the architecture
- $W^{pred} \in \mathbb{R}^{D \times d}$ decoder weights
- $b^{pred} \in \mathbb{R}^{D}$ decoder biases

Model architecture: attention

Intuition for attention comes from databases: a key operation is given a **query**, find the relevant **key**, and lookup the corresponding **value**.



1. Figure credit to Andrej Risteski's course: CMU 10707 - Deep Learning (2020)

Model architecture: attention

A more "differentiable" variant of this:

$$\text{Attention}(q, K, V) = \sum_i \text{similarity}(q, K_i) V_i$$

Vector

Matrices w/ rows keys/values

Linear combination of "most similar" values

The simplest notion of similarity: inner product.

$$\text{Attention}(q, K, V) = \sum_i \text{softmax}(\langle q, K_i \rangle) V_i$$

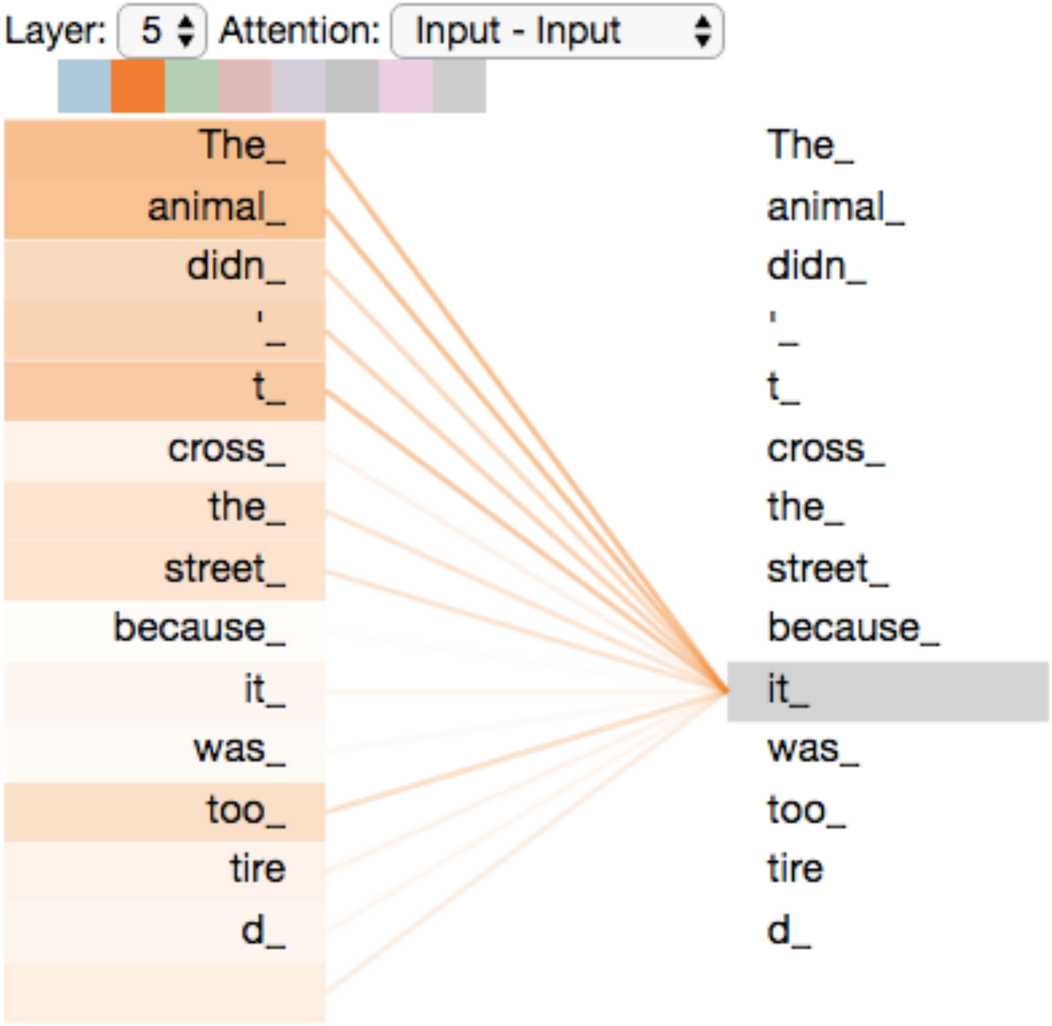
Produces distribution over keys

Produces convex combination of values

Multiple queries combined in matrix Q: $\text{Attention}(Q, K, V) = \text{softmax}(QK^T)V$

1. Figure credit to Andrej Risteski's course: CMU 10707 - Deep Learning (2020)

Model architecture: attention



1. Figure credit to Jay Alammar's blog post: https://jalammar.github.io/illustrated-transformer/

Model architecture: single-layer transformer

- Embedding layer
  - Recall: original sentence $X \in \mathbb{R}^{D \times N}$
  - Column X_j is one-hot: the word at position j
  - Masked sentence $\tilde{X} \in \mathbb{R}^{D \times N}$, with embedding Z
  - $Z = W^E \tilde{X} \in \mathbb{R}^{d \times N}$
  - $W^E \in \mathbb{R}^{d \times D}$ is the embedding matrix
    - For each word in 1…D
    - Pick the corresponding column in W^E
    - Get a d-dimensional word embedding
  - Weight tie with W^{pred} (common implementation[1])

- D: vocabulary size
- d: embedding dimension
- (usually: $d < D$)
- N: sequence length
- $Z \in \mathbb{R}^{d \times N}$
- $W^{pred} \in \mathbb{R}^{D \times d}$
- $b^{pred} \in \mathbb{R}^{D}$

$$f(\tilde{X}) = W^{pred}\left(W^V W^E \tilde{X}\right) Attn(\tilde{X}) + b^{pred}$$

- $f(\tilde{X}) = (W^E)^\top \left(W^V W^E \tilde{X}\right) Attention(\tilde{X}) + b^{pred} \in \mathbb{R}^{D \times N}$

1. Ofir Press and Lior Wolf. Using the output embedding to improve language models. 2017

Model architecture: single-layer transformer

- $f(\tilde{X}) = (W^E)^\top (W^V W^E \tilde{X})\, Attn(\tilde{X}) + b^{pred}$
- Topic structure can be encoded in many places
  - Embedding layer W^E
  - Self-attention W^V, $Attn(\tilde{X})$
- Our simplification: study the above two cases separately
- **Main result 1** (word embedding)
- **Main result 2** (self-attention)

- D: vocabulary size
- d: embedding dimension
- (usually: $d < D$)
- N: sequence length
- $Z \in \mathbb{R}^{d \times N}$
- $W^{pred} \in \mathbb{R}^{D \times d}$
- $b^{pred} \in \mathbb{R}^{D}$

$$f(\tilde{X}) = W^{pred}(W^V W^E \tilde{X})\, Attn(\tilde{X}) + b^{pred}$$

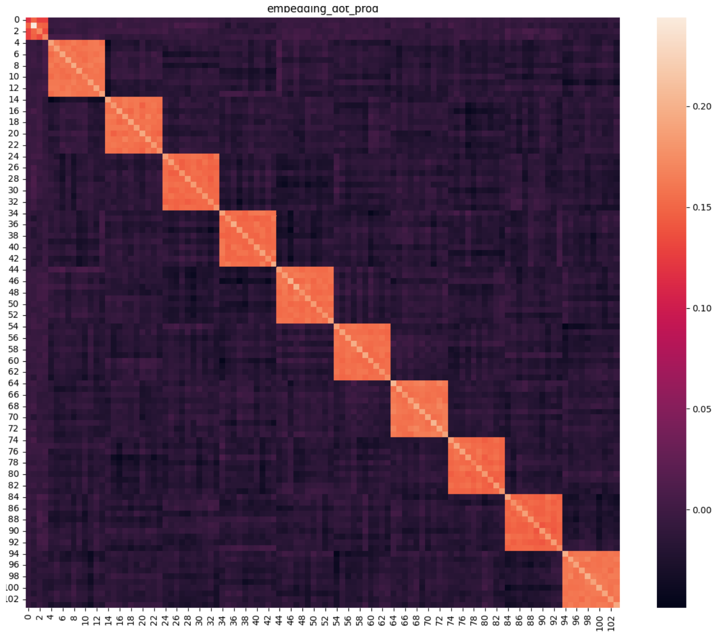
Result: embeddings encode topic structure

- Theorem (informal): when fixing $Attn(\tilde{X})$ to uniform attention and W^V to identity, the optimal embedding layer W^E satisfies
- $E \coloneqq {W^E}^\top W^E$ is block-wise
- E_{ij} is larger when words i and j belong to the same topic
  - ≈ their embeddings are more similar
- E_{ij} is smaller when words i and j belong to the different topics
  - ≈ their embeddings are more different
- The avg difference (same topic - diff topic)

$$\frac{1}{v(1-(1-p_c)p_m)}$$

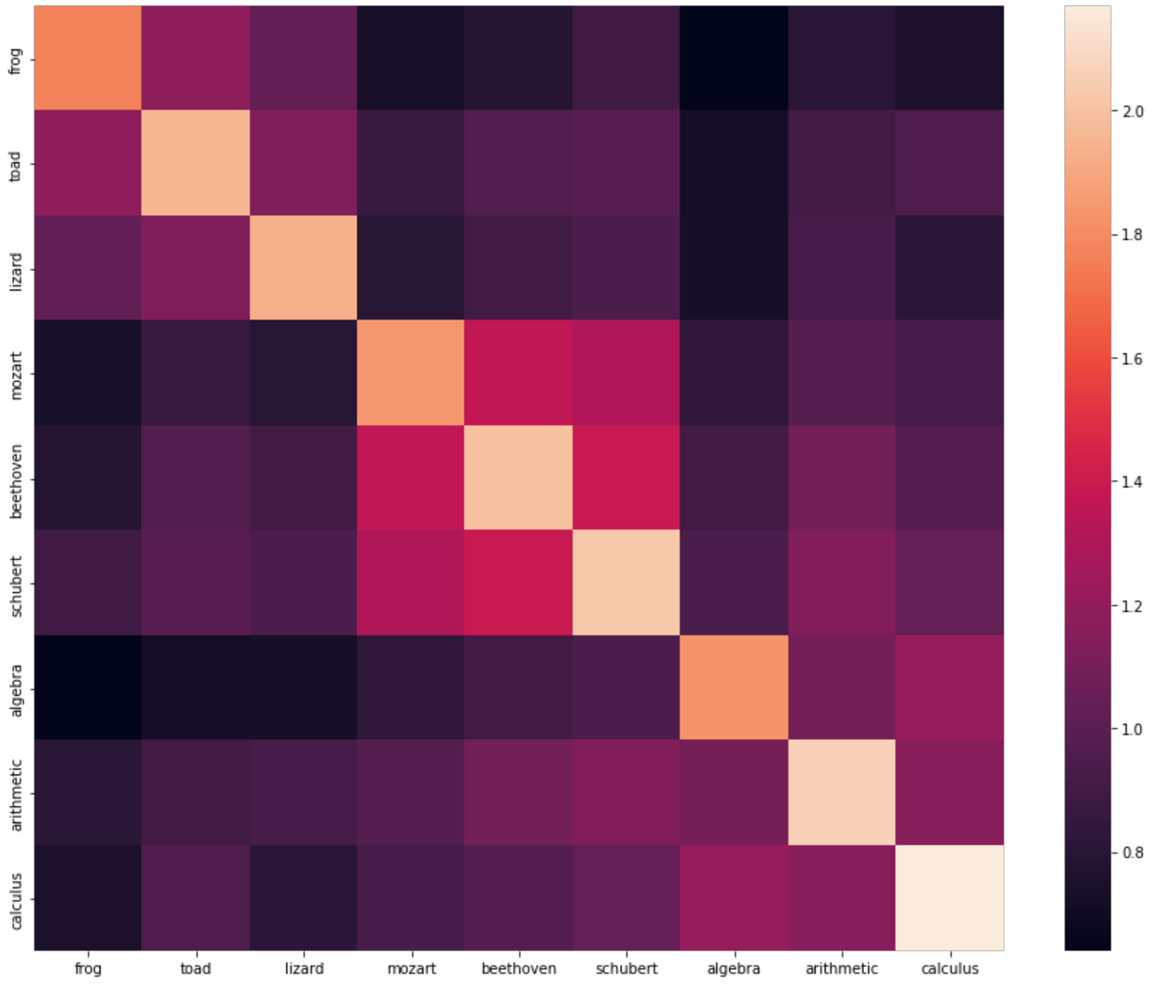
- D: vocabulary size
- d: embedding dimension
- (usually: $d < D$)
- N: sequence length
- $Z \in \mathbb{R}^{d \times N}$
- $W^{pred} \in \mathbb{R}^{D \times d}$
- $b^{pred} \in \mathbb{R}^{D}$
- v: number of words in each topic
- p_m, p_c, p_r: controls masking probabilities

$$f(\tilde{X}) = (W^E)^\top (W^V W^E \tilde{X})\, Attn(\tilde{X}) + b^{pred}$$



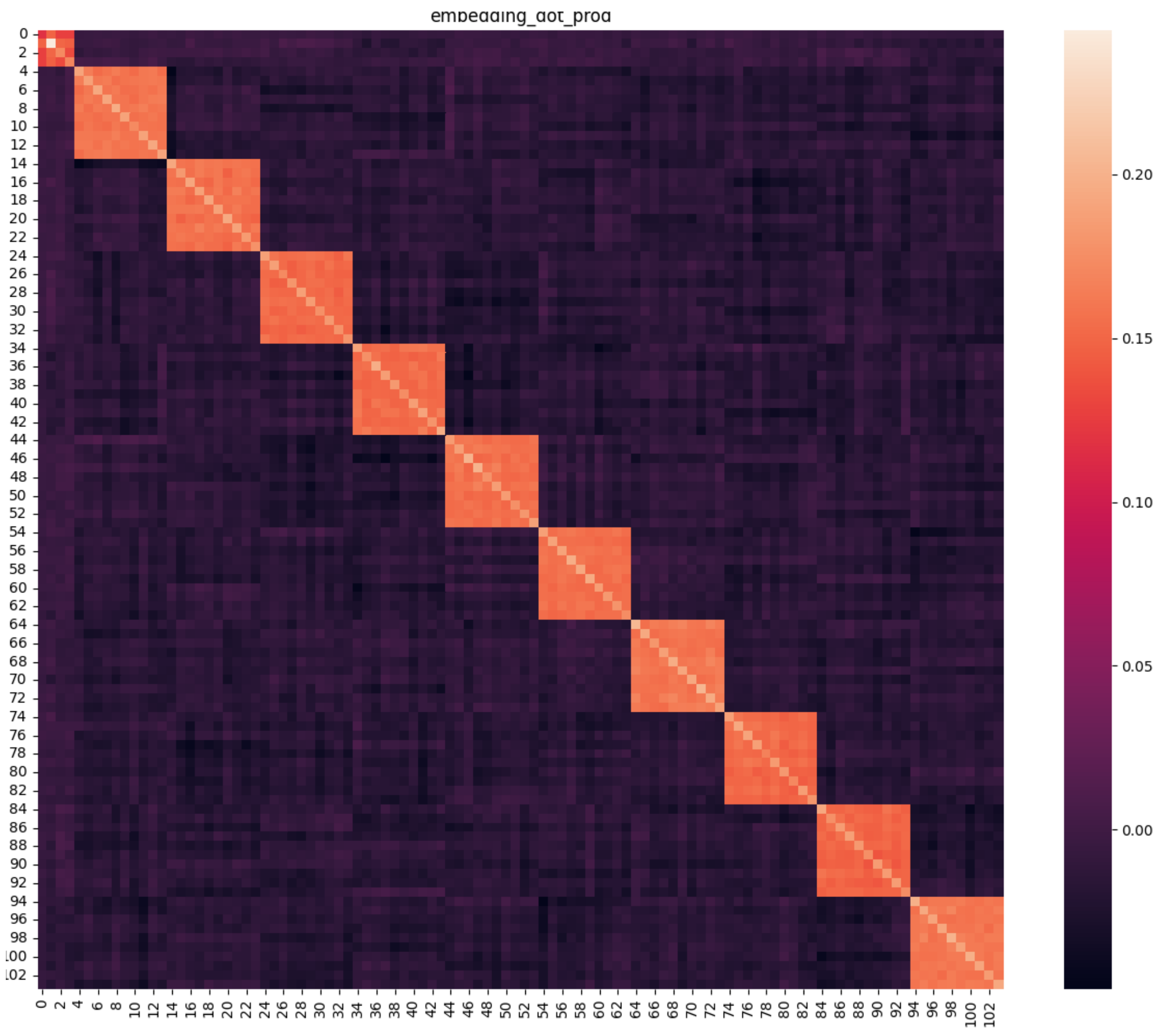
Result: embeddings encode topic structure

- The dot product of the embeddings of two word is
  - larger if the two words belong to the same topic, and
  - smaller if they belong to different topics
- In this figure, the nine words fall into three topics:
  - Animals: frog, toad, lizard
  - Musicians: mozart, beethoven, schubert
  - Mathematics: algebra, arithmetic, calculus



Result: embeddings encode topic structure

- The dot product of the embeddings of two word is
  - larger if the two words belong to the same topic, and
  - smaller if they belong to different topics
- Same holds for model trained on synthetic data generated by LDA
  - 10 topics
  - 10 words in each topic
  - Theory: fix $Attn(\tilde{X})$ and W^V
  - This figure: all components are trained
  - Block pattern depends on optimization algorithm and loss function
    - Can be less clean, see Figure 1 in our paper



Next step: results for other layers

- Question: what is the role of other layers in learning topic structures?
- In particular, what does the attention layer learn?
- We isolate the roles of embeddings and attention by considering the following question
- What does the attention layer learn without the help of embeddings?
- Namely, we freeze the embedding to be one-hot

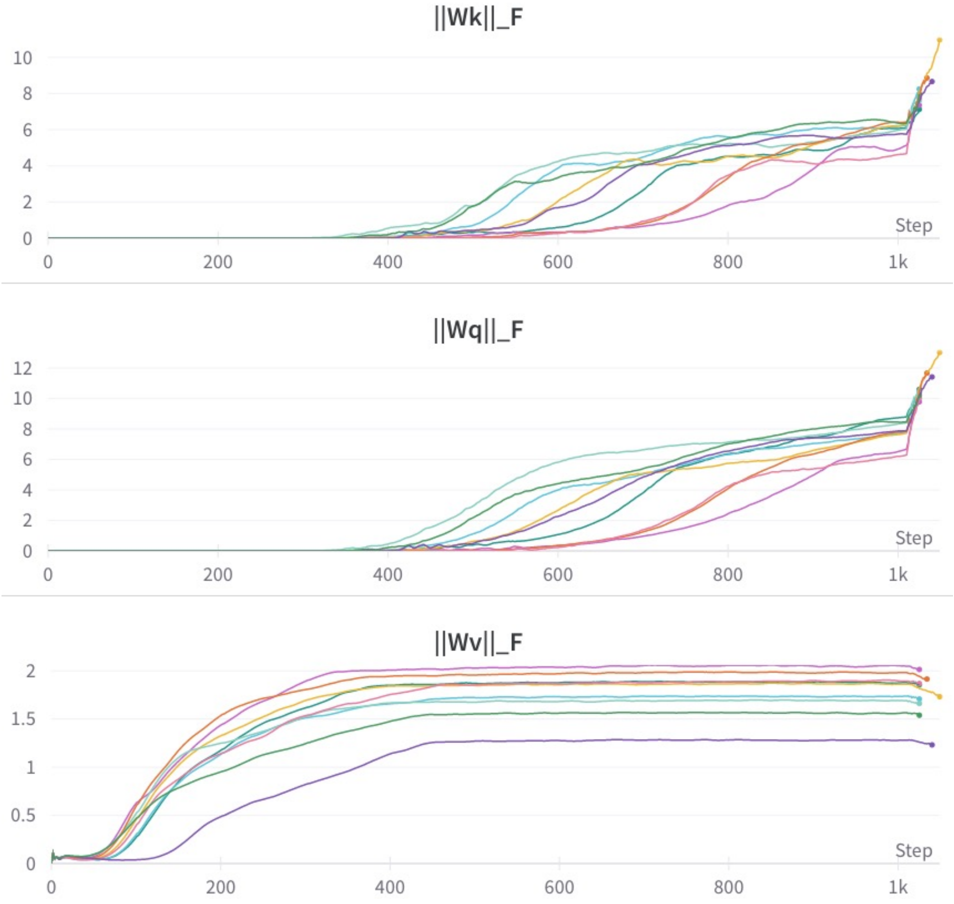
The two-stage optimization process

- However, end-to-end gradient descent learning dynamics of transformers involves very complicated calculations
- Can we avoid them but still gain insights into the optimization process?
- Empirical observation
  - With careful initialization
  - When all weights are jointly trained (using SGD or Adam)
  - The optimization process can be approximately broken down into two stages

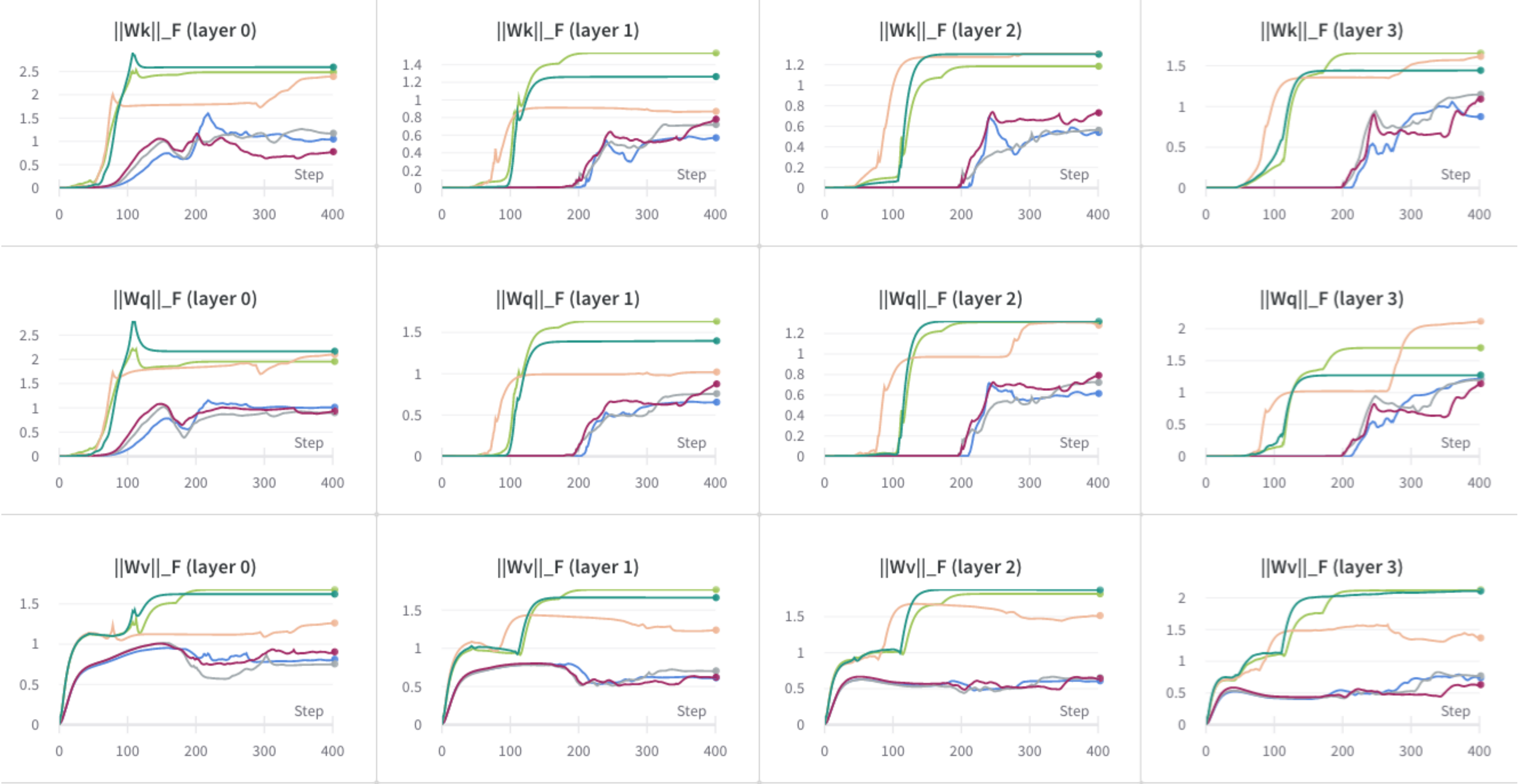
$$f(Z) = W^{pred}(W^V Z)\sigma\left(\frac{(W^K Z)^\top (W^Q Z)}{\sqrt{d_a}}\right) + b^{pred}$$

Observation: two-stage optimization process

- In Stage 1 (steps 0-400)
  - $||W^K||_F, ||W^Q||_F \approx 0$
  - $||W^V||_F$ increases significantly
  - Our simplification for theory: freeze W^K and W^Q to 0
- In Stage 2 (steps 400-1000),
  - $||W^K||_F, ||W^Q||_F$ start increasing significantly
  - while $||W^V||_F$ stays relatively flat
  - Note: W^V does not stop changing
  - Our simplification for theory: freeze W^V to the Stage 1 optima above



Observation: two-stage optimization process



Intuition: two-stage optimization process

- $g(W^K, W^Q, W^V) := (W^V Z)\, Attn(Z)$
- $Attn(Z) = \sigma\left(\frac{(W^K Z)^\top (W^Q Z)}{\sqrt{d_a}}\right)$
  - σ: softmax (each column sums up to 1)
- $\nabla_{W^K}(g)$ contains the term W^Q
  - Initialization: $W^K \approx 0, W^Q \approx 0$
  - So $\nabla_{W^K}(g) \approx 0$
  - So W^K stays ≈ 0 for a long time
  - Similar for W^Q
- Does not apply to W^V: $\nabla_{W^V}(g)$ contains $Attn(Z)$
  - $Attn(Z)$ is not ≈ 0

- D: vocabulary size
- d: embedding dimension
- (usually: $d < D$)
- N: sequence length
- $Z = W^E \tilde{X} \in \mathbb{R}^{d \times N}$
- $W^{pred} \in \mathbb{R}^{D \times d}$
- $b^{pred} \in \mathbb{R}^D$
- v: number of words in each topic
- p_m, p_c, p_r: controls masking probabilities

$$f(\tilde{X}) = (W^E)^\top \left(W^V W^E \tilde{X}\right) Attn(\tilde{X}) + b^{pred}$$

The two-stage optimization process

- However, end-to-end gradient descent learning dynamics of transformers involves very complicated calculations
- Can we avoid them but still gain insights into the optimization process?
- Empirical observation
  - With careful initialization
  - When all weights are jointly trained (using SGD or Adam)
  - The optimization process can be approximately broken down into **two stages**
- Our approach
  - For Stage 1 (convex), we characterize the optima, which also implies guarantees for training dynamics
  - For Stage 2 (non-convex), we only characterize the optima (no guarantee for training dynamics)

Stage 1 result: W^V encodes topic structure

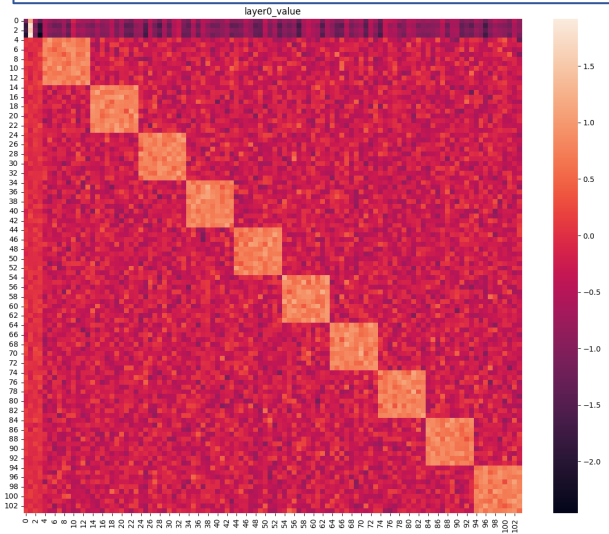
- Theorem (informal): with one-hot embedding, fixing $Attn(\tilde{X})$ to uniform attention, the optimal W^V is block-wise
- ${W^V}_{\mathrm{ij}}$ is larger when words i and j belong to the same topic
- ${W^V}_{\mathrm{ij}}$ is smaller when words i and j belong to the different topics
- The avg difference (same topic - diff topic)

$$\frac{1}{v(1-(1-p_c)p_m)}$$

- Weight decay makes the optima unique
  - w/o weight decay: not strongly convex

- D: vocabulary size
- d: embedding dimension
- (usually: $d < D$)
- N: sequence length
- $Z \in \mathbb{R}^{d \times N}$
- $W^{pred} \in \mathbb{R}^{D \times d}$
- $b^{pred} \in \mathbb{R}^{D}$
- v: number of words in each topic
- p_m, p_c, p_r: controls masking probabilities

$$f(\tilde{X}) = (W^E)^\top (W^V W^E \tilde{X})\, Attn(\tilde{X}) + b^{pred}$$

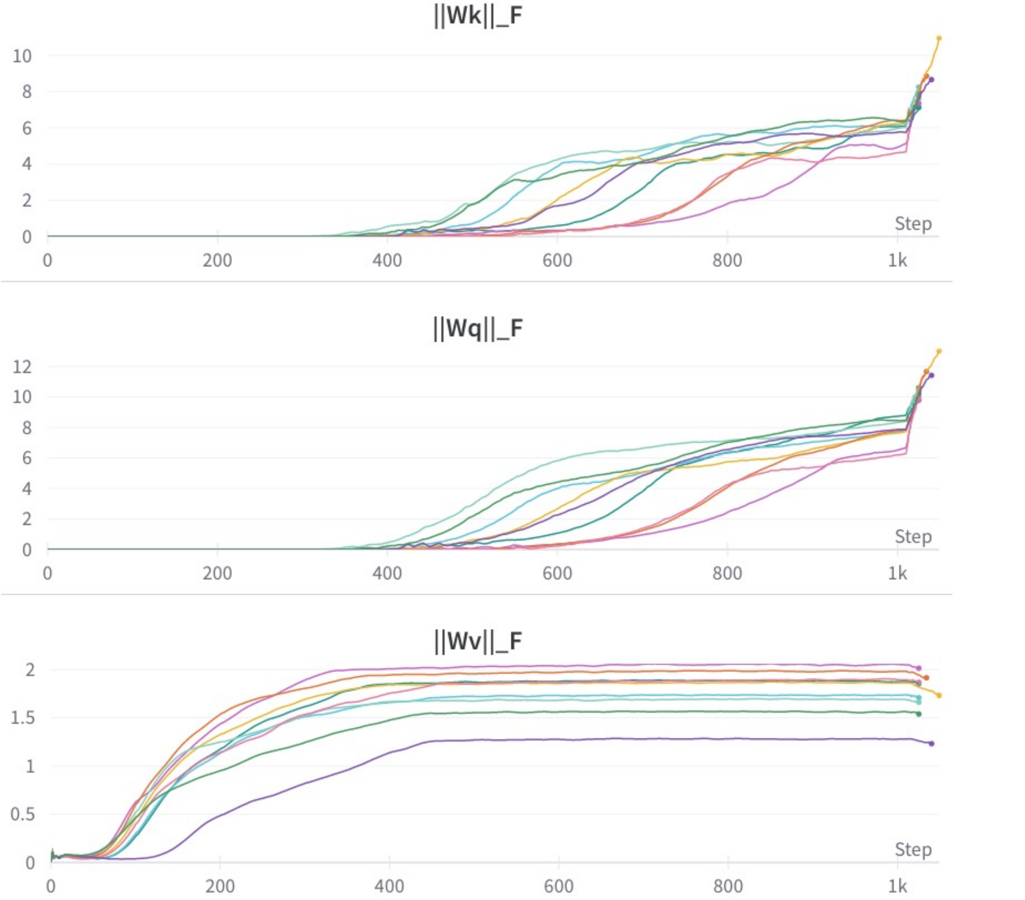


Stage 2 question: behavior of attention

- $Z \in \mathbb{R}^{d \times N}$
- $W^{pred} \in \mathbb{R}^{D \times d}$
- $b^{pred} \in \mathbb{R}^{D}$

$f(\tilde{X}) = (W^E)^\top (W^V W^E \tilde{X})\, Attn(\tilde{X}) + b^{pred}$

- Q: Fixing W^V at Stage 1 optima, what is the optimal $Attn(Z)$?
- $Attn(Z) = \sigma\left(\frac{(W^K Z)^\top (W^Q Z)}{\sqrt{d_a}}\right)$
- $\sigma: \mathbb{R}^{N\times N} \to (0,1)^{N \times N}$: column-wise softmax
- $\sigma(A)_{ij} = \frac{\exp(A_{ij})}{\sum_{l=1}^{N} \exp(A_{lj})}$
  - The "attention score" that word j pays to word I
- Q: Do words typically pay more attention to other words of the same topic?
  - i.e. is $\sigma(A)_{ij}$ typically larger
  - when topic(wi) = topic(wj)
  - or when topic(wi) ≠ topic(wj) ?



Stage 2 simplification: tying attention scores

- Masked document $w = w_1, \dots, w_N$
- $Attn(X)_{ij}$ is the attention that w_j pays to w_i
- $Attn(X)_{ij} =$
  - c_1 if $w_i = w_j$
  - c_2 if $w_i \neq w_j$ but $\text{topic}(w_i) = \text{topic}(w_j)$
  - c_3 if $\text{topic}(w_i) \neq \text{topic}(w_j)$
- Q: Are c_1 and c_2 greater than c_3 in the optimal attention?
- Let $\alpha := \frac{c_2}{c_3}$, and $\beta := \frac{c_1}{c_3}$
- Q: Are α and β greater than 1 in the optimal attention?

Stage 2 result: attention encodes topic structure

- Theorem (informal): with one-hot embedding, when fixing W^V at stage 1 optima,
- the optimal attention scores are topic-wise
- $\frac{v-1}{v}\alpha + \frac{1}{v}\beta \in (\lambda_1 \tau, \lambda_2 T)$
- Intuition:
  - v: number of words per topic
  - τ: number of topics per document
  - T: total number of topics
  - $\frac{v-1}{v}\alpha + \frac{1}{v}\beta$ avg same-topic / diff-topic
  - "avg" in the sense of frequency
  - λ_1, λ_2 are constants
  - More topics per doc (i.e. larger τ) =>
    - each word needs to focus more on other same-topic words

- $Attn(X)_{ij} =$
  - c_1 if $w_i = w_j$
  - c_2 if $w_i \neq w_j$ but $\text{topic}(w_i) = \text{topic}(w_j)$
  - c_3 if $\text{topic}(w_i) \neq \text{topic}(w_j)$
- $\alpha := \frac{c_2}{c_3}$ (same-topic-diff-word attn / diff-topic attn)
- $\beta := \frac{c_1}{c_3}$ (same-word attn / diff-topic attn)

Optimizer and Learning Rate	Avg Same-Word Attention	Avg Same-Topic--Different-Word Attention	Avg Different-Topic Attention
Adam 0.003	0.00759 ± 0.00171	0.0108 ± 0.000657	0.00689 ± 0.000160
Adam 0.01	0.00811 ± 0.000705	0.010 ± 0.000392	0.00707 ± 0.000178
Adam 0.03	0.00453 ± 0.000346	0.0116 ± 0.000460	0.00665 ± 0.000200
SGD 0.01	0.0105	0.0106	0.00673
SGD 0.03	0.0140 ± 0.00158	0.0103 ± 0.000357	0.00641 ± 0.0000239

Experiment setting on Wikipedia[1] dataset

- Topic model: run online LDA[2] for 6 passes
- Ambiguity filtering
  - Theory (synthetic setting): topics don't overlap, i.e. each word belongs to 1 topic
  - Experiment
    - Remove "stop tokens"
    - For each topic, keep the "most representative words"
    - i.e. 0.05%, 0.1%, or 0.2% of all words with highest P(word | this topic) in the fitted LDA
    - Will show results when enforcing no overlap between topics (≈ theory)
    - Also, results when topics can overlap (≠ theory)
- Transformer models
  - Pre-trained Bert (closest to theoretical setting)
  - Pre-trained Albert, Bart, Electra, Roberta (≠ theory)
  - Randomly-initialized Bert (expect no topic structure)

1. Wikimedia Foundation. URL https://dumps.wikimedia.org. 2. Matthew Hoffman et al. Online Learning for Latent Dirichlet Allocation. 2010.

Experiment result on Wikipedia dataset

Model	Ambiguity Threshold	Avg embedding Cosine Similarity (Same-topic/Diff-topic)	Avg embedding Dot Product (Same-topic/Diff-topic)	Avg attn weight (Same-topic /Diff-topic)
Bert	0.0005	1.21	1.19	1.32
	0.001	1.13	1.15	1.28
	0.002	1.11	1.13	1.22
Albert	0.0005	5.64	6.29	1.33
	0.001	4.18	3.74	1.28
	0.002	3.24	2.93	1.22
Bart	0.0005	2.80	2.67	1.35
	0.001	1.95	1.92	1.31
	0.002	1.63	1.62	1.23
Electra	0.0005	5.98	5.37	2.14
	0.001	7.70	7.35	2.09
	0.002	7.46	8.08	1.95
Roberta	0.0005	6.44	6.81	1.40
	0.001	5.73	6.31	1.31
	0.002	5.24	5.30	1.22
Bert (randomly initialized)	0.0005	1.00080	1.00063	0.99943
	0.001	0.99974	1.00036	0.99996
	0.002	1.00016	1.00027	1.00007

topics don’t overlap (≈ theory)

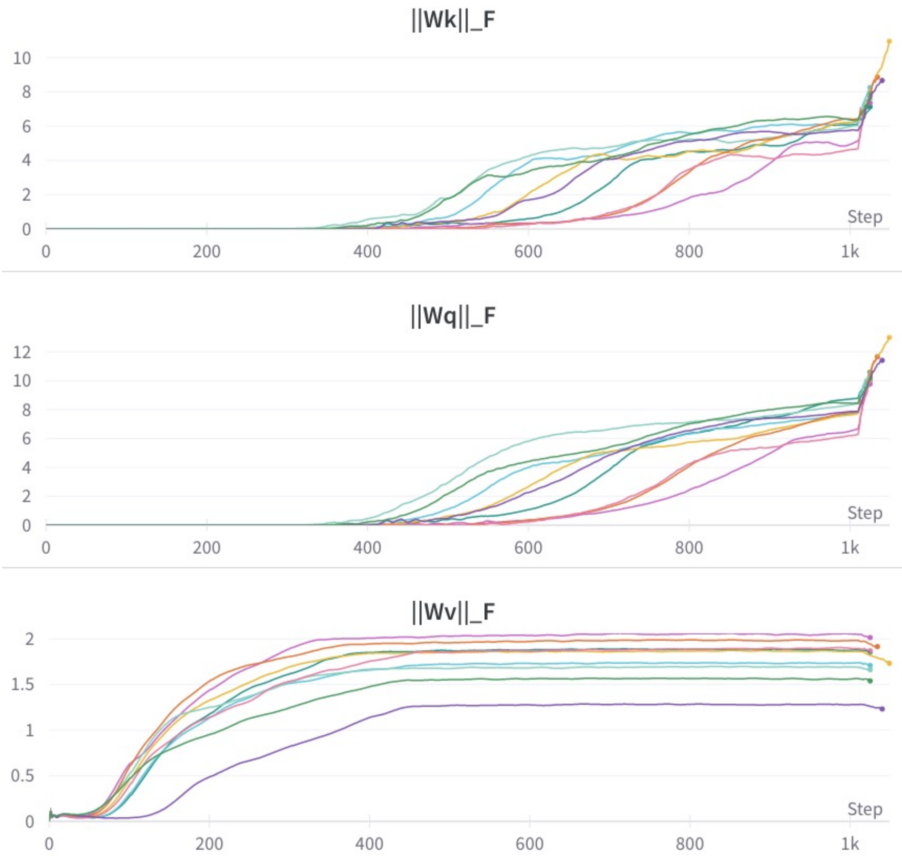
Experiment result on Wikipedia dataset

Model	Ambiguity Threshold	Avg embedding Cosine Similarity (Same-topic/Diff-topic)	Avg embedding Dot Product (Same-topic/Diff-topic)	Avg attn weight (Same-topic /Diff-topic)
Bert	0.0005	1.14	1.04	1.23
	0.001	0.97	1.05	1.17
	0.002	0.99	0.93	1.13
Albert	0.0005	4.15	3.06	1.23
	0.001	3.09	3.04	1.17
	0.002	1.54	1.44	1.11
Bart	0.0005	2.51	1.76	1.27
	0.001	1.63	1.12	1.20
	0.002	1.06	0.85	1.11
Electra	0.0005	5.28	3.99	1.70
	0.001	5.56	5.57	1.58
	0.002	6.39	5.61	1.48
Roberta	0.0005	4.39	5.01	1.19
	0.001	5.20	4.25	1.15
	0.002	4.71	4.15	1.12
Bert (randomly initialized)	0.0005	0.99814	0.99957	1.00009
	0.001	0.99820	1.00167	1.00013
	0.002	0.99964	0.99928	0.99978

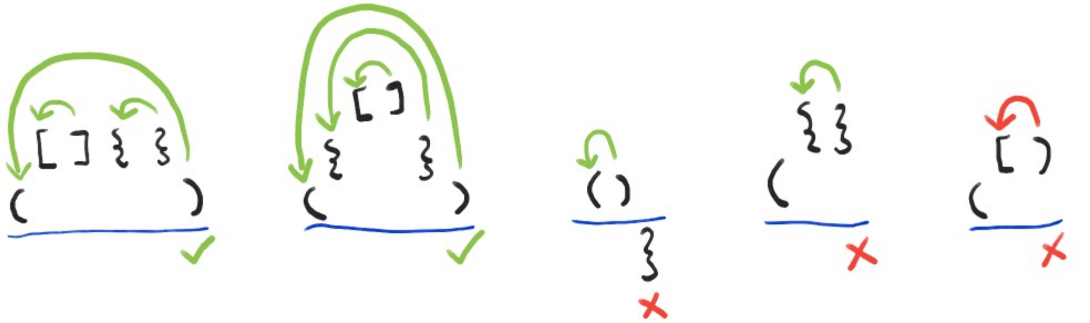
topics can overlap (≠ theory)

Interesting future directions

- Analyzing optimization beyond the two-stage assumption
  - Two stage: simplified the early optimization process
  - Learning of (simple) topic structure: ✅
  - Other finer-grained data properties: ?
  - What happens after this early process: ?
  - Two-stage phenomenon: sensitive to hyper-params
  - Common default hyperparameters: not visibly two-stage
  - Interaction between different components (jointly trained): ?



- Apply similar methodology to other distributions
  - Topic model: one aspect of semantics: ✅
  - Other aspects of semantics: ?
  - Syntax: ?
  - Ongoing work: the Dyck grammar, coming soon!



Summary

- **Data:** topic modeling: Latent Dirichlet allocation (LDA)
- **Model architecture:** a single-layer transformer (no FFN, no layer norm)
- **Pre-training task:** masked language modeling
- Analyzing optimization process
  - The early training process can be approximately broken down into two stages
  - Stage 1 is convex, stage 2 is not
  - We characterize the optima of the training objective in each stage
  - Since stage 1 is convex => training dynamics convergence guarantee for stage 1
  - These optima intuitively captures the topic structures in the data distribution
- Theory is also predictive of multi-layer multi-head transformers trained on Wikipedia data

Contact: yuchenl4@cs.cmu.edu

https://arxiv.org/abs/2303.04245 (to appear in ICML 2023)