


Modern LLMs: Pretraining, Adaptation, Instruction Fine-Tuning, In-Context Learning



Zachary Lipton & Henry Chai

10701 — November 20th

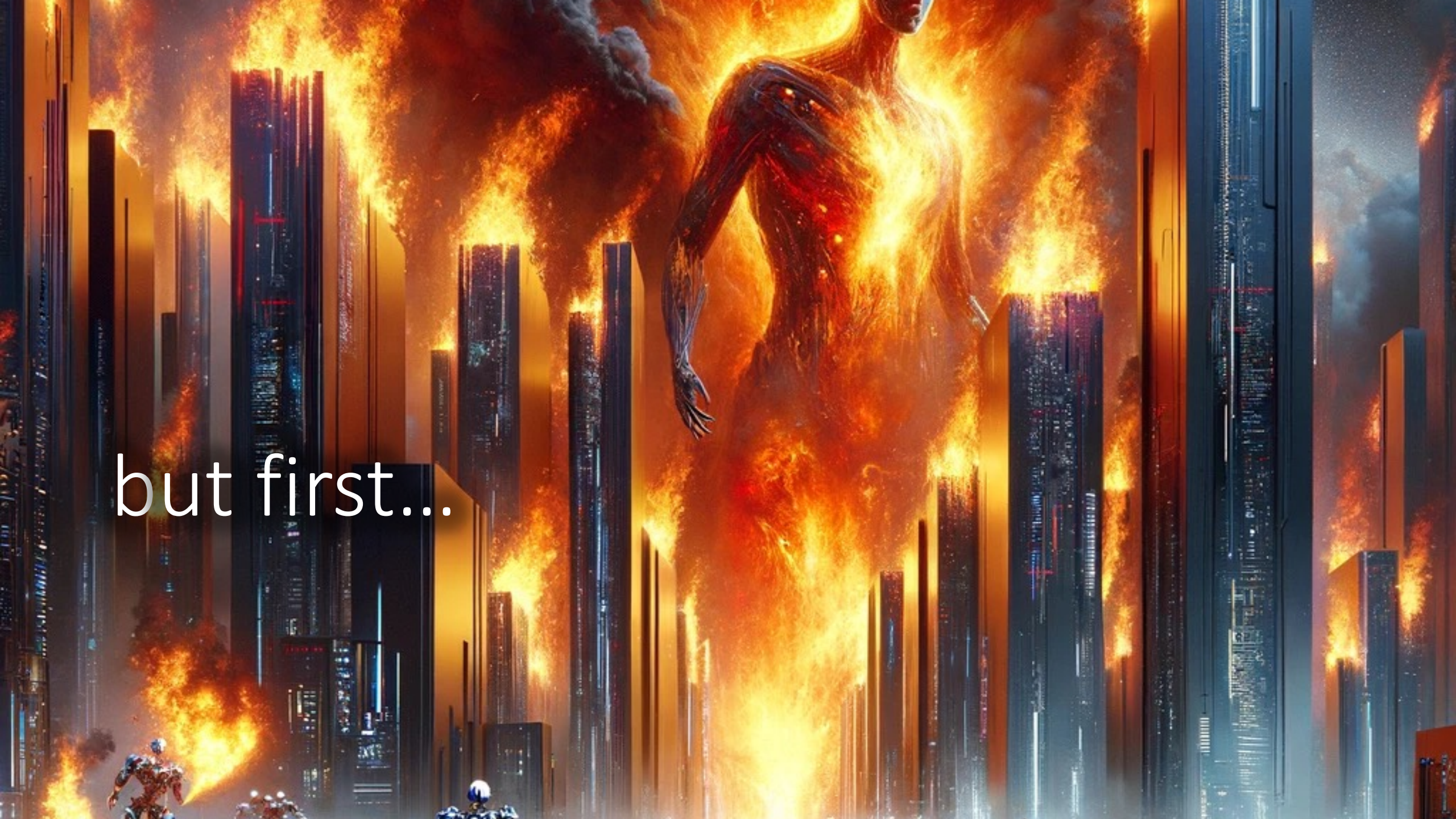


Modern LLMs: Pretraining, Adaptation, Instruction Fine-Tuning, In-Context Learning



Zachary Lipton & Henry Chai




10701 — November 20th



but first...

Word2Vec

- Learn an embedding vector for each word
- Use $\langle \mathbf{x}, \mathbf{y} \rangle$ to measure the similarity
- Build a probability model
- Maximize the likelihood function to learn the model

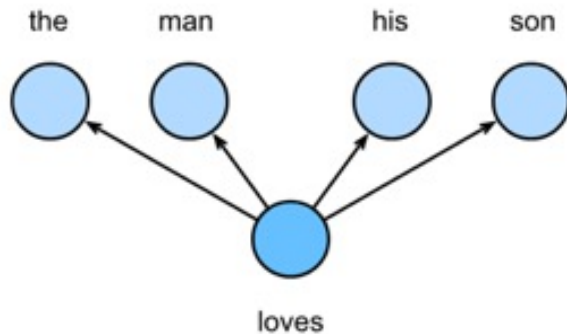
	x	y	z
	1	0	0
	0	1	0
⋮	⋮	⋮	⋮
	0	0	1

[Efficient Estimation of Word Representations in Vector Space \(2012\)](#)

[Distributed representations of words and phrases and their compositionality \(NeurIPS 2013\)](#)

The Skip-Gram Model

- A word can be used to generate the words surround it
- Given the center word, the context words are generated independently



$$\begin{aligned} & \mathbb{P}(\text{"the"}, \text{"man"}, \text{"his"}, \text{"son"} \mid \text{"loves"}) \\ &= \mathbb{P}(\text{"the"} \mid \text{"loves"}) \cdot \mathbb{P}(\text{"man"} \mid \text{"loves"}) \\ & \quad \cdot \mathbb{P}(\text{"his"} \mid \text{"loves"}) \cdot \mathbb{P}(\text{"son"} \mid \text{"loves"}) \end{aligned}$$

Likelihood function

	Word	Embedding
Center	w_c	$\mathbf{v}_c \in \mathbb{R}^d$
Context	w_o	$\mathbf{u}_o \in \mathbb{R}^d$

$$\mathbb{P}(w_o | w_c) = \frac{\exp(\mathbf{u}_o^\top \mathbf{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \mathbf{v}_c)}$$

\mathcal{V} : all context words

- Given length T sequence, context window m , the likelihood function:

$$\prod_{t=1}^T \prod_{-m \leq j \leq m, j \neq 0} \mathbb{P}(w^{(t+j)} | w^{(t)})$$

Negative sampling

- Treat a center word and a context word appear in the same context window as an event

$$\mathbb{P}(D = 1 | w_c, w_o) = \sigma(\mathbf{u}_c^T \mathbf{v}_o) \quad \sigma(x) = \frac{1}{1 + \exp(-x)}$$

- Change the likelihood function from $\prod_{t=1}^T \prod_{-m \leq j \leq m, j \neq 0} \mathbb{P}(w^{(t+j)} | w^{(t)})$ to

$$\prod_{t=1}^T \prod_{-m \leq j \leq m, j \neq 0} \mathbb{P}(D = 1 | w^{(t)}, w^{(t+j)})$$

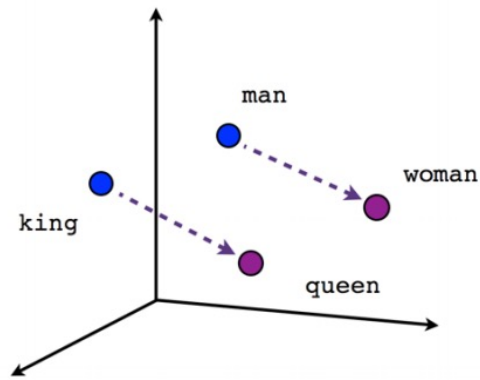
Negative sampling (cont'd)

- Sample noise word w_n that doesn't appear in the window

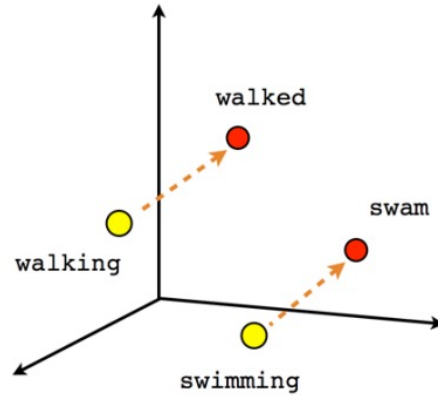
$$\mathbb{P}(D = 0 | w_c, w_n) = 1 - \sigma(\mathbf{u}_n^T \mathbf{v}_c)$$

- Add into the likelihood function as well
- Maximizing the likelihood equates to solving a binary classification problem with a binary logistic regression loss

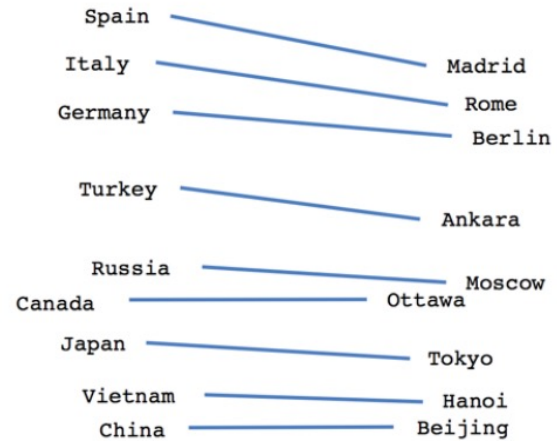
Strange properties — linear relationships



Male-Female



Verb tense



Country-Capital

Stereotypes Captured in Word Embeddings

- “Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings”

Extreme *she* occupations

- | | | |
|-----------------|-----------------------|------------------------|
| 1. homemaker | 2. nurse | 3. receptionist |
| 4. librarian | 5. socialite | 6. hairdresser |
| 7. nanny | 8. bookkeeper | 9. stylist |
| 10. housekeeper | 11. interior designer | 12. guidance counselor |

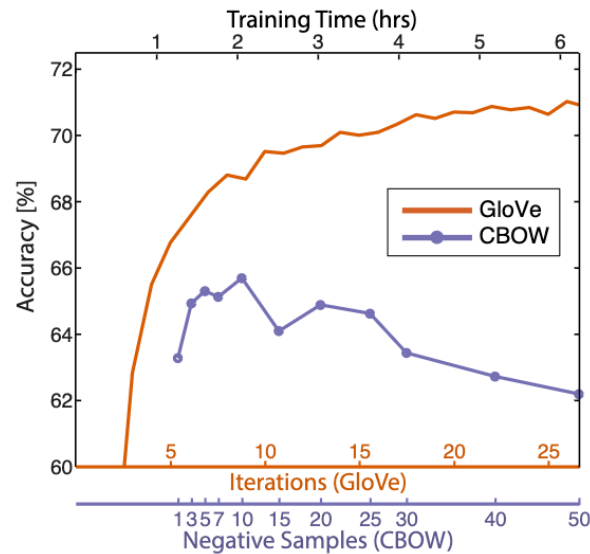
Extreme *he* occupations

- | | | |
|----------------|-------------------|----------------|
| 1. maestro | 2. skipper | 3. protege |
| 4. philosopher | 5. captain | 6. architect |
| 7. financier | 8. warrior | 9. broadcaster |
| 10. magician | 11. fighter pilot | 12. boss |

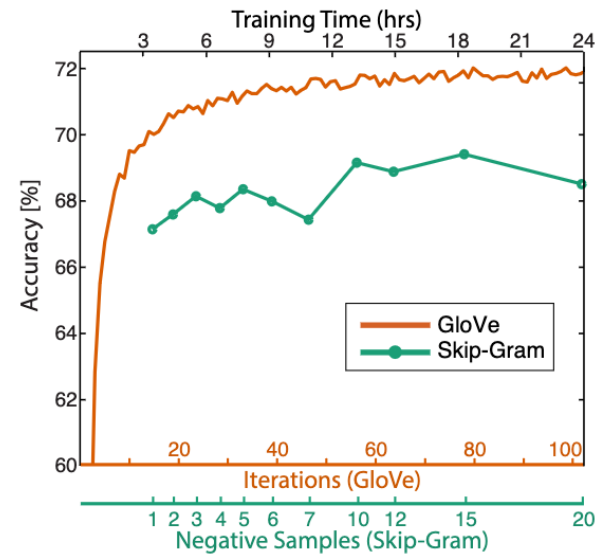
source: <https://arxiv.org/pdf/1607.06520.pdf>

GLoVe (2013)

- Word vectors computed based on word-word co-occurrence stats
- Factorizes log of the co-occurrence matrix
- Rivalled / overtook W2V in word embedding popularity

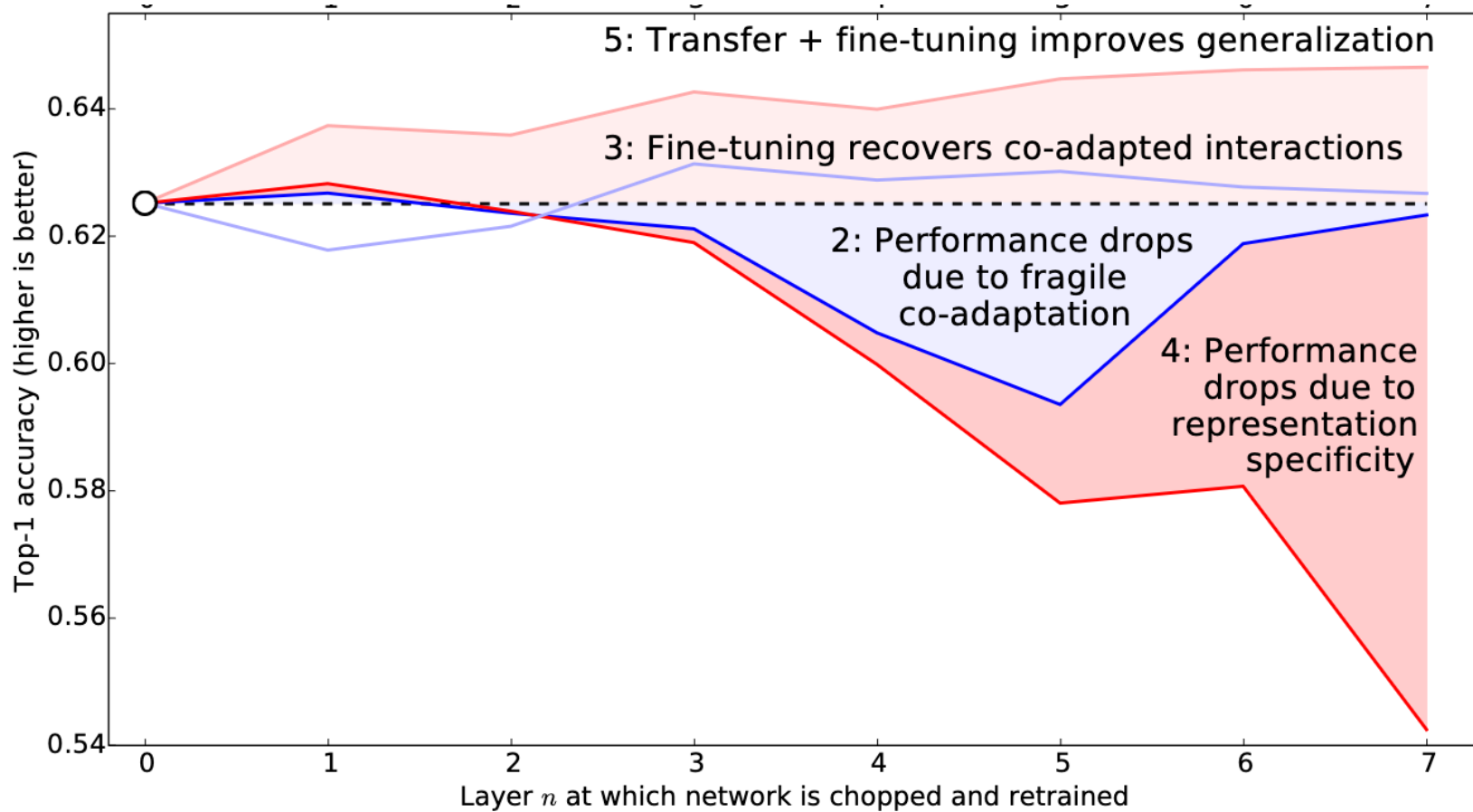


(a) GloVe vs CBOW



(b) GloVe vs Skip-Gram

In Image World



Just Ahead of Its Time

432v1 [cs.LG] 4 Nov 2015

Semi-supervised Sequence Learning

Andrew M. Dai
Google Inc.
adai@google.com

Quoc V. Le
Google Inc.
qvl@google.com

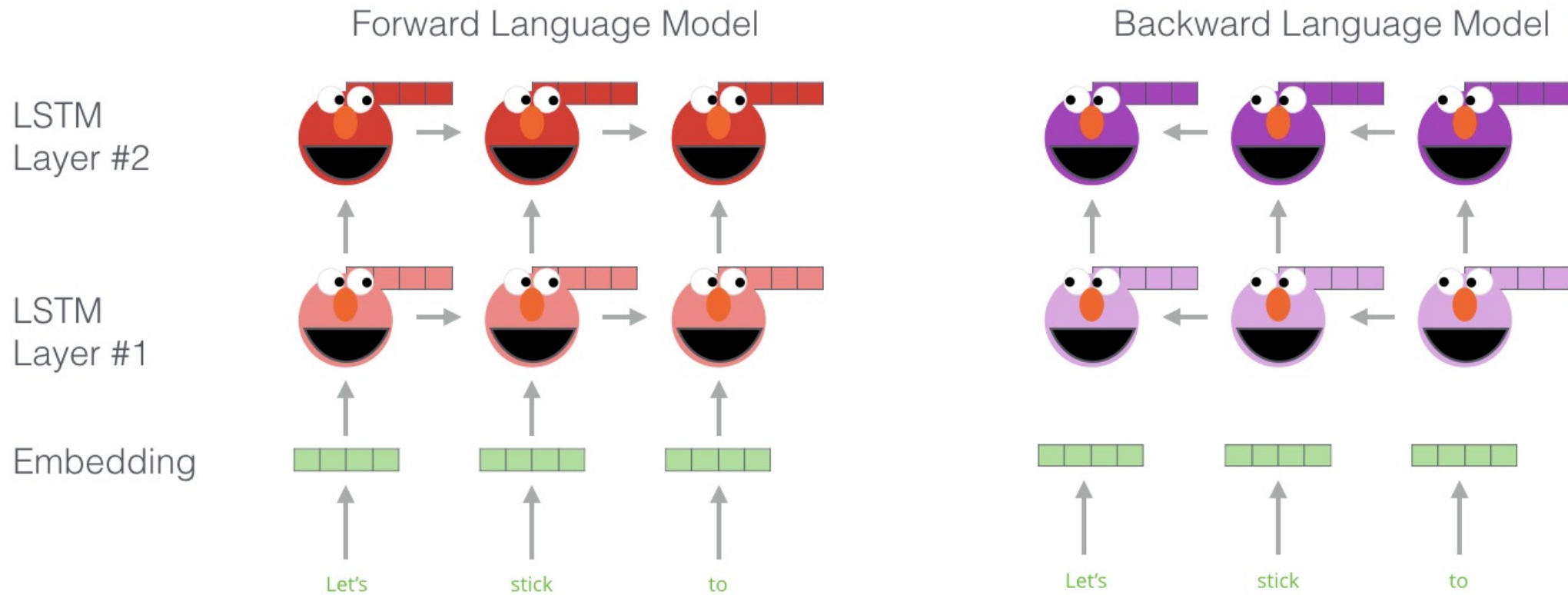
Abstract

We present two approaches that use unlabeled data to improve sequence learning with recurrent networks. The first approach is to predict what comes next in a sequence, which is a conventional language model in natural language processing. The second approach is to use a sequence autoencoder, which reads the input sequence into a vector and predicts the input sequence again. These two algorithms can be used as a “pretraining” step for a later supervised sequence learning algorithm. In other words, the parameters obtained from the unsupervised step can be used as a starting point for other supervised training models. In our experiments, we find that long short term memory recurrent networks after being pretrained with the two approaches are more stable and generalize better. With pretraining, we are able to train long short term memory recurrent networks up to a few hundred timesteps, thereby achieving strong performance in many text classification tasks, such as IMDB, DBpedia and 20 Newsgroups.

[Dai & Le \(ICLR 2016\)](#)

ELMO (2017)

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



source: <http://jalammar.github.io/illustrated-bert/>

ELMO (NAACL 2018)

- Deep contextualized word representations
- Bidirectional Language Model
 - Learn both forward and backward language models
 - Parameters tied for input embedding and softmax layers
- Architecture Details
 - Base token representation: 2048 character n-gram convolutional filters, linear projection down to 512 dimensional word embedding
 - Two hidden BiLSTM layers, 4096 units each
 - Residual connection from first to second layer
- Pretrained on “large” text corpus
 - 10 Epochs on the 1B Word Benchmark
 - Gets to perplexity 37 (SOTA at the time was 30 in [Józefowicz et al. 2016](#))

ELMO Adaptation Strategy

- Learn a task-specific weighting of the intermediate representations
- Make these contextual embeddings the inputs to downstream model

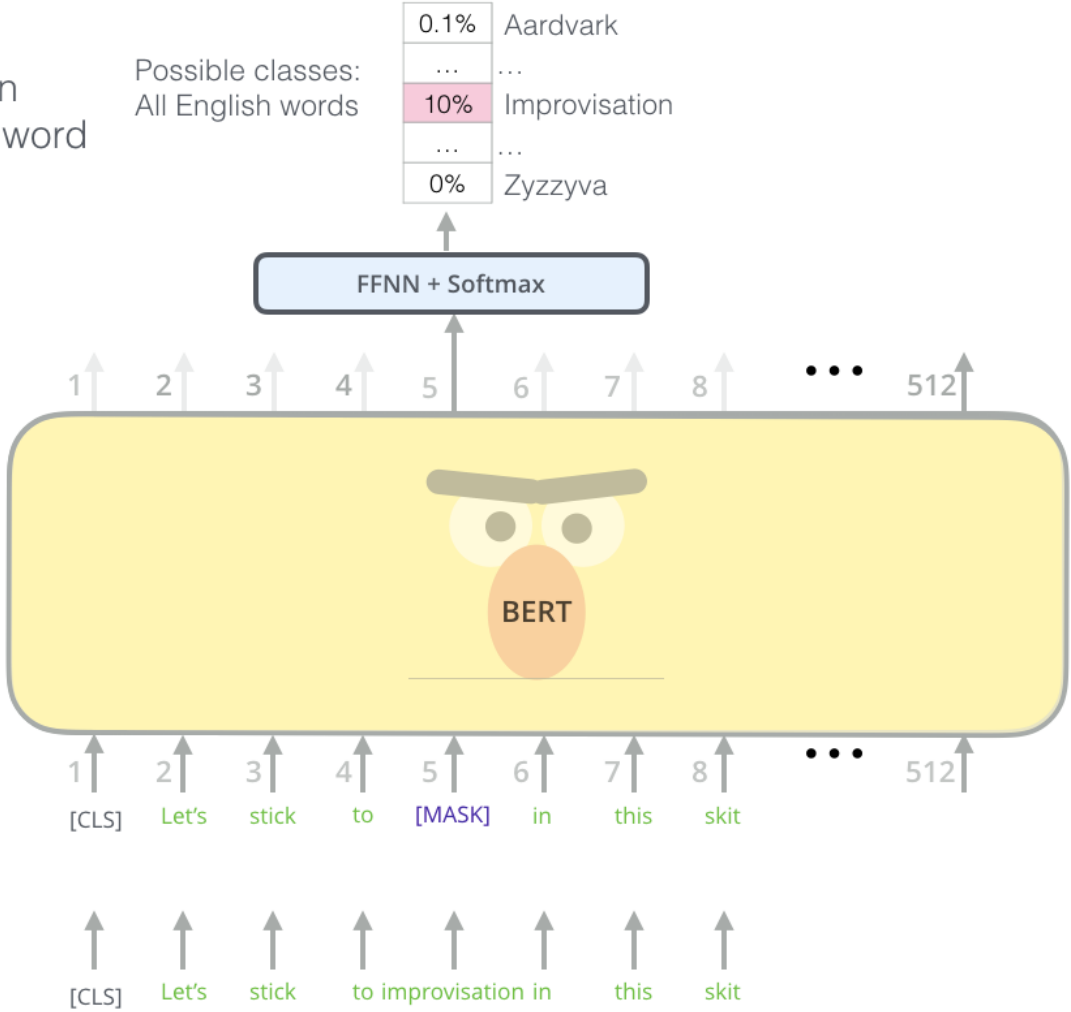
ELMO key results

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 \pm 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 \pm 0.19	90.15	92.22 \pm 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 \pm 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; F_1 for SQuAD, SRL and NER; average F_1 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.

BERT

Use the output of the masked word's position to predict the masked word



source: <http://jalammar.github.io/illustrated-bert/>

BERT Fast Facts

- Deep bi-directional Transformer architecture
- Pretrained on next sentence prediction
- “Masked Language Modeling” Objective
- On each iteration, chooses 15% of all input tokens at random:
 - replaced w [MASK] token 80% of time, random token 10% of time, same token 10% of the time
- Pretrains on BooksCorpus (800M words)
- BERT-Large uses 24 layers, 1024-dim hidden representation, 16 attention heads, 355M parameters
- New state of the art on 11 NLP tasks
- Pushes GLUE score to 80.5% (from 72.8)

BERT Adaptation Strategy

- Fine-tune all parameters on downstream task
- Output mechanism for downstream task depends on task type:
 - [CLS] token fed to classifier for classification
 - Token representations fed to output layer for token-level tasks
- Fine-tuning took under an hour on TPUs < 3 hours on GPUs (< 1hr on modern GPUs)

BERT key results

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average
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.

RoBERTa

- Replicates BERT, makes some major improvements
- Trains longer (500k vs 100k steps), with bigger batches, over more data (160GB) — BookCorpus, CC-News, OpenWebText, Stories
- Removes the next sentence prediction objective
- Trains over longer sequences
- Dynamically changes masking

RoBERTa Results

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
<i>Single-task single models on dev</i>										
BERT _{LARGE}	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
XLNet _{LARGE}	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	90.2/90.2	94.7	92.2	86.6	96.4	90.9	68.0	92.4	91.3	-
<i>Ensembles on test (from leaderboard as of July 25, 2019)</i>										
ALICE	88.2/87.9	95.7	90.7	83.5	95.2	92.6	68.6	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	96.8	93.0	67.8	91.6	90.4	88.4
RoBERTa	90.8/90.2	98.9	90.2	88.2	96.7	92.3	67.8	92.2	89.0	88.5

Table 5: Results on GLUE. All results are based on a 24-layer architecture. BERT_{LARGE} and XLNet_{LARGE} results are from [Devlin et al. \(2019\)](#) and [Yang et al. \(2019\)](#), respectively. RoBERTa results on the development set are a median over five runs. RoBERTa results on the test set are ensembles of *single-task* models. For RTE, STS and MRPC we finetune starting from the MNLI model instead of the baseline pretrained model. Averages are obtained from the GLUE leaderboard.

[RoBERTa: A Robustly Optimized BERT Pretraining Approach \(Liu et al.\)](#)

Longformer

- Attention mechanism that scales linearly with length
- “Outperforms RoBERTa on long document tasks”
- Sets new SOTA on WikiHop and TriviaQA

T5 (2020)

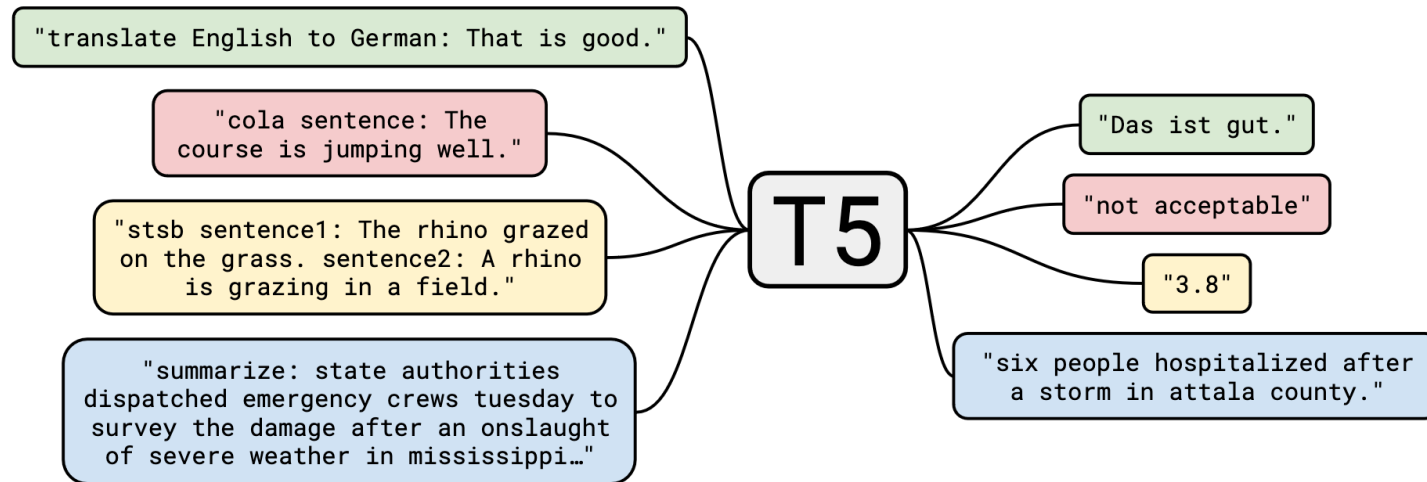


Figure 1: A diagram of our text-to-text framework. Every task we consider—including translation, question answering, and classification—is cast as feeding our model text as input and training it to generate some target text. This allows us to use the same model, loss function, hyperparameters, etc. across our diverse set of tasks. It also provides a standard testbed for the methods included in our empirical survey. “T5” refers to our model, which we dub the “**T**ext-to-**T**ext **T**ransfer **T**ransformer”.

T5 Details

- Casts all tasks into a “text-to-text” format
- Explores standard BERT model sizes, also larger 3B & 11B models
- BERT-style pretraining
- Fine-tuning to large collection of downstream tasks
- Achieved SOTA results on summarization, QA, text classification, etc.

T5 Results

Model	GLUE Average	CoLA Matthew's	SST-2 Accuracy	MRPC F1	MRPC Accuracy	STS-B Pearson	STS-B Spearman
Previous best	89.4 ^a	69.2 ^b	97.1 ^a	93.6^b	91.5^b	92.7 ^b	92.3 ^b
T5-Small	77.4	41.0	91.8	89.7	86.6	85.6	85.0
T5-Base	82.7	51.1	95.2	90.7	87.5	89.4	88.6
T5-Large	86.4	61.2	96.3	92.4	89.9	89.9	89.2
T5-3B	88.5	67.1	97.4	92.5	90.0	90.6	89.8
T5-11B	90.3	71.6	97.5	92.8	90.4	93.1	92.8

Model	QQP F1	QQP Accuracy	MNLI-m Accuracy	MNLI-mm Accuracy	QNLI Accuracy	RTE Accuracy	WNLI Accuracy
Previous best	74.8 ^c	90.7^b	91.3 ^a	91.0 ^a	99.2^a	89.2 ^a	91.8 ^a
T5-Small	70.0	88.0	82.4	82.3	90.3	69.9	69.2
T5-Base	72.6	89.4	87.1	86.2	93.7	80.1	78.8
T5-Large	73.9	89.9	89.9	89.6	94.8	87.2	85.6
T5-3B	74.4	89.7	91.4	91.2	96.3	91.1	89.7
T5-11B	75.1	90.6	92.2	91.9	96.9	92.8	94.5

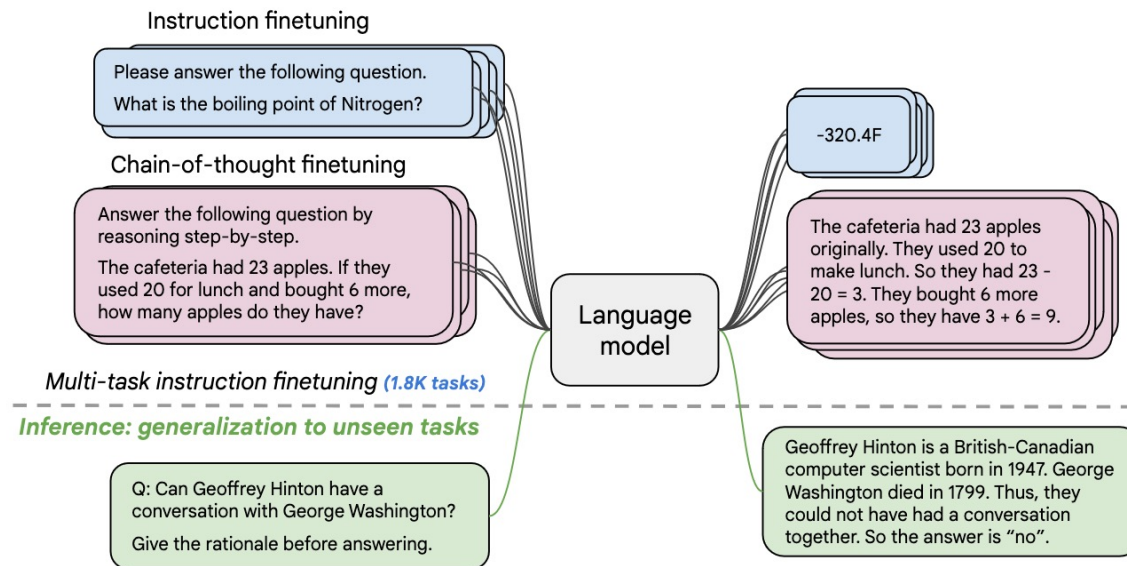
Model	SQuAD EM	SQuAD F1	SuperGLUE Average	BoolQ Accuracy	CB F1	CB Accuracy	COPA Accuracy
Previous best	90.1 ^a	95.5 ^a	84.6 ^d	87.1 ^d	90.5 ^d	95.2 ^d	90.6 ^d
T5-Small	79.10	87.24	63.3	76.4	56.9	81.6	46.0
T5-Base	85.44	92.08	76.2	81.4	86.2	94.0	71.2
T5-Large	86.66	93.79	82.3	85.4	91.6	94.8	83.4
T5-3B	88.53	94.95	86.4	89.9	90.3	94.4	92.0
T5-11B	91.26	96.22	88.9	91.2	93.9	96.8	94.8

Model	MultiRC F1a	MultiRC EM	ReCoRD F1	ReCoRD Accuracy	RTE Accuracy	WiC Accuracy	WSC Accuracy
Previous best	84.4 ^d	52.5 ^d	90.6 ^d	90.0 ^d	88.2 ^d	69.9 ^d	89.0 ^d
T5-Small	69.3	26.3	56.3	55.4	73.3	66.9	70.5
T5-Base	79.7	43.1	75.0	74.2	81.5	68.3	80.8
T5-Large	83.3	50.7	86.8	85.9	87.8	69.3	86.3
T5-3B	86.8	58.3	91.2	90.4	90.7	72.1	90.4
T5-11B	88.1	63.3	94.1	93.4	92.5	76.9	93.8

Model	WMT EnDe BLEU	WMT EnFr BLEU	WMT EnRo BLEU	CNN/DM ROUGE-1	CNN/DM ROUGE-2	CNN/DM ROUGE-L
Previous best	33.8^e	43.8^e	38.5^f	43.47 ^g	20.30 ^g	40.63 ^g
T5-Small	26.7	36.0	26.8	41.12	19.56	38.35
T5-Base	30.9	41.2	28.0	42.05	20.34	39.40
T5-Large	32.0	41.5	28.1	42.50	20.68	39.75
T5-3B	31.8	42.6	28.2	42.72	21.02	39.94
T5-11B	32.1	43.4	28.1	43.52	21.55	40.69

FLAN-*

- “explore instruction finetuning with a particular focus on
 - (1) scaling the number of tasks
 - (2) scaling the model size
 - (3) finetuning on chain-of-thought data”



Decoder-Only Revolution

Original GPT

- Trains 12-layer Transformer LM on BooksCorpus data (7000 books)
- Masked self-attention language modeling
- 12 attention heads with 768-dimensional states
- 3072 dimensional inner states for MLP layers
- BPE vocabulary
- Modified L2 regularization, attention dropout of .1, GELU activation
- Cosine learning rate annealing
- Minibatches of size 64

GPT Uses

- Fine tuned to numerous downstream tasks
 - Dropout .1 added to classifier
 - Fine-tunes for 3 epochs of additional training on target task
 - Linear learning rate decay with warmup of .2%
- Fine-tuned to many tasks
 - NLI, QA, “Common-sense reasoning”, Semantic Similarity, Classification

GPT-2

- 1.5 billion parameter auto-regressive Transformer language model
- Trained on new dataset called WebText
- Fed documents + questions, matches the performance of $\frac{3}{4}$ baseline QA systems (without even looking at the training data)
- “These findings suggest a promising path towards building language processing systems which learn to perform tasks from their naturally occurring demonstrations”

GPT-2 Rollout and Controversy

Tech

AI deemed 'too dangerous to release' makes it out into the world

Extremists could generate harmful content, researchers warn

Andrew Griffin • Thursday 07

OpenAI Trains Language Model, Mass Hysteria Ensues



Zachary C. Lipton

February 17, 2019

Journalism, Machine Learning Ethics, Natural Language Processing, Uncategorized

Deep Learning, Fake News, Language Modeling, Open Source, OpenAI

[f Facebook](#) [Twitter](#) [Pinterest](#) [in LinkedIn](#)

On Thursday, OpenAI announced that they had trained a language model. They used a large training dataset and showed that the resulting model was useful for downstream tasks where training data is scarce. They announced the new model with a puffery press release, complete with **this animation (below) featuring dancing text**. They demonstrated that their model could produce realistic-looking text and warned that they would be keeping the dataset, code, and model weights private. The world promptly lost its mind.

Prompt Engineering

- Largely manual process
- Write instructions
- Qualitatively evaluate outputs
- Revise instructions
- Most basic activity consists of editing a basic instruction, however, more elaborate techniques have emerged

GPT-3

- Trained 175-billion parameter autoregressive Transformer
- Released in 75-page report, beginning of a trend on large systems
- Part academic methodological paper, part methodological white paper, part risk assessment doc
- Can generate news articles that human evaluators struggle to distinguish from real articles

GPT3 Training Data

- Built off Common Crawl but applied 3 filtering heuristics:
 - Filtered “based on similarity to a range of high-quality reference” (?!?)
 - Fuzzy de-duplication at the document level, within and across datasets, both to improve quality but also to preserve integrity of holdout stats
 - Added high-quality data to training mix (WebText, two books-based corpora [Books1 & Books2], and English-language Wikipedia)
- Common crawl was 45TB before filtering, 570GB after filtering
- During training Common Crawl & Books2 sampled 1x, other datasets sampled 2–3x

Few-Shot Prompting

The three settings we explore for in-context learning

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
```

Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.



Few-Shot Performance w LM Scale

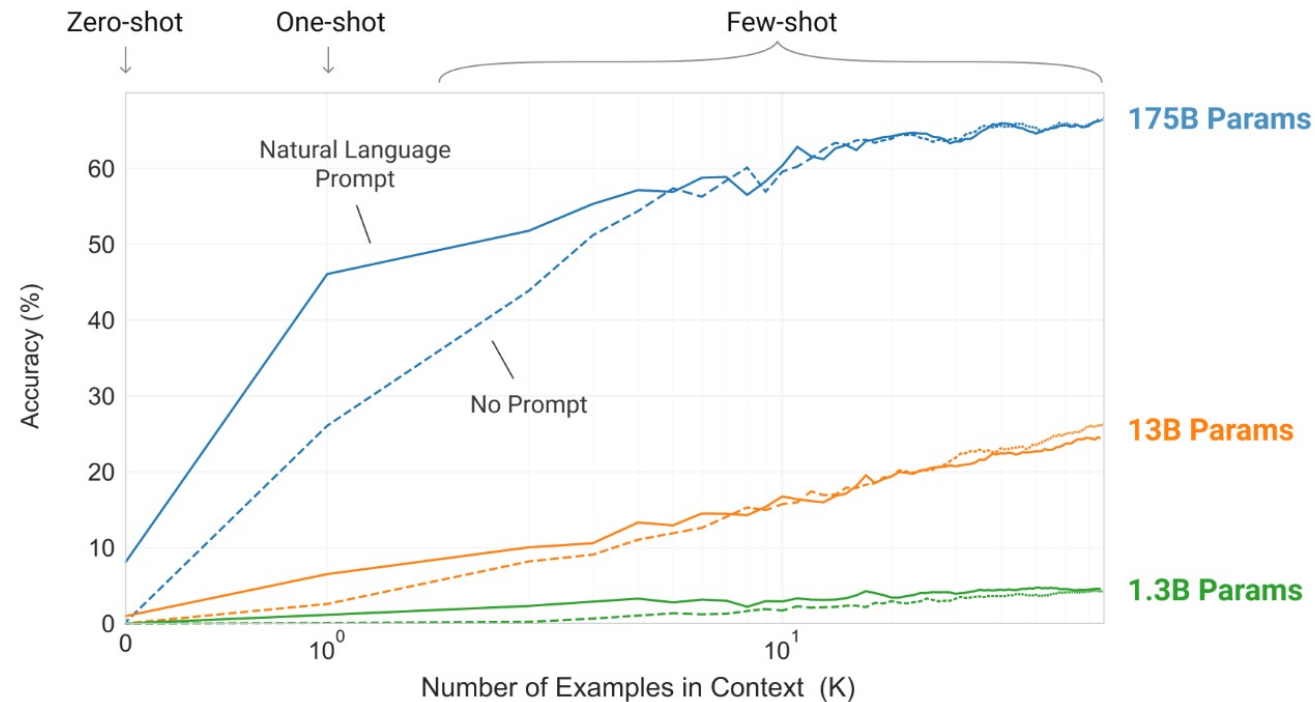


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper “in-context learning curves” for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

Scale Benefits Few-Shot More than Zero Shot

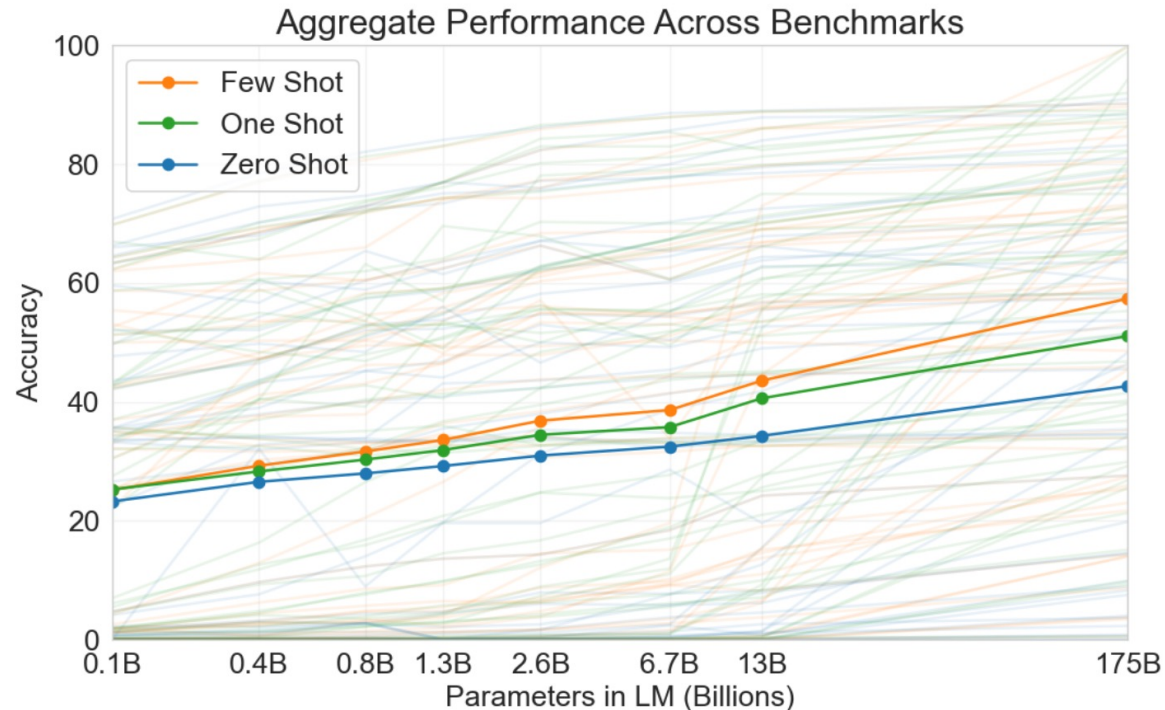


Figure 1.3: Aggregate performance for all 42 accuracy-denominated benchmarks While zero-shot performance improves steadily with model size, few-shot performance increases more rapidly, demonstrating that larger models are more proficient at in-context learning. See Figure 3.8 for a more detailed analysis on SuperGLUE, a standard NLP benchmark suite.

Chain of Thought Prompting

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ❌

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✅

Chain of Thought Examples

Math Word Problems (free response)

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Math Word Problems (multiple choice)

Q: How many keystrokes are needed to type the numbers from 1 to 500?
Answer Choices: (a) 1156 (b) 1392 (c) 1480 (d) 1562 (e) 1788

A: There are 9 one-digit numbers from 1 to 9. There are 90 two-digit numbers from 10 to 99. There are 401 three-digit numbers from 100 to 500. $9 + 90(2) + 401(3) = 1392$. The answer is (b).

CSQA (commonsense)

Q: Sammy wanted to go to where the people were. Where might he go?
Options: (a) race track (b) populated areas (c) desert (d) apartment (e) roadblock

A: The answer must be a place with a lot of people. Race tracks, desert, apartments, and roadblocks don't have a lot of people, but populated areas do. So the answer is (b).

StrategyQA

Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about 0.6 g/cm^3 , which is less than water. Thus, a pear would float. So the answer is no.

Date Understanding

Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?

A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943. 10 days before today is 05/23/1943. So the answer is 05/23/1943.

Sports Understanding

Q: Is the following sentence plausible? "Joao Moutinho caught the screen pass in the NFC championship."

A: Joao Moutinho is a soccer player. The NFC championship is part of American football, not soccer. So the answer is no.

SayCan (Instructing a robot)

Human: How would you bring me something that isn't a fruit?

Explanation: the user wants something to eat that isn't a fruit. An energy bar is not a fruit, so I will bring the user an energy bar.

Plan: 1. find(energy bar) 2. pick(energy bar) 3. find(user) 4. put(energy bar) 5. done().

Last Letter Concatenation

Q: Take the last letters of the words in "Lady Gaga" and concatenate them.

A: The last letter of "Lady" is "y". The last letter of "Gaga" is "a". Concatenating them is "ya". So the answer is ya.

Coin Flip (state tracking)

Q: A coin is heads up. Maybelle flips the coin. Shalonda does not flip the coin. Is the coin still heads up?

A: The coin was flipped by Maybelle. So the coin was flipped 1 time, which is an odd number. The coin started heads up, so after an odd number of flips, it will be tails up. So the answer is no.

Retrieval Augmented Generation

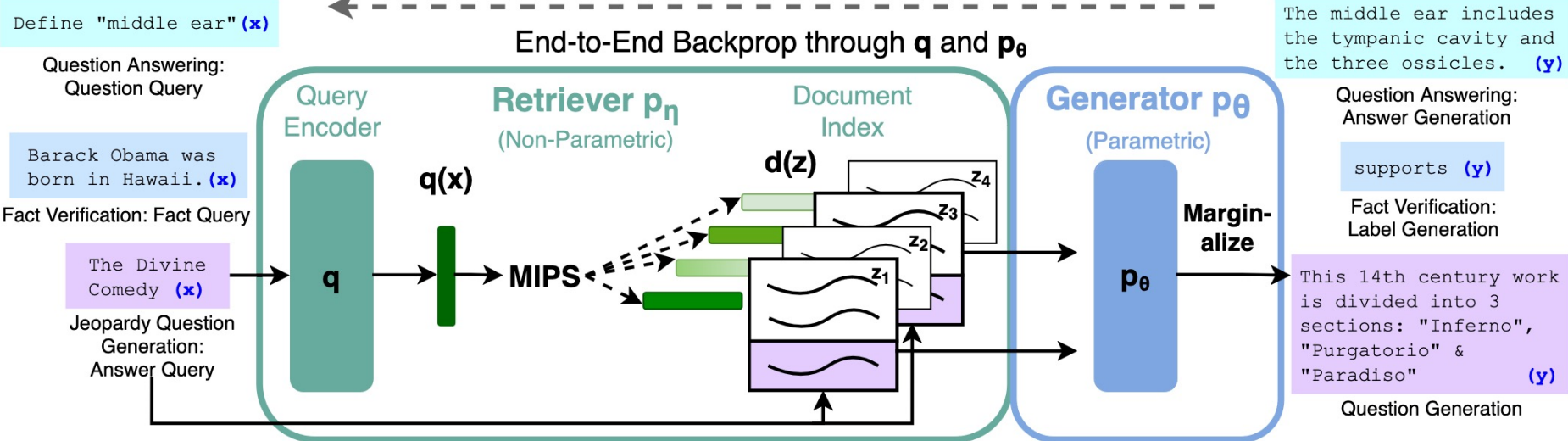
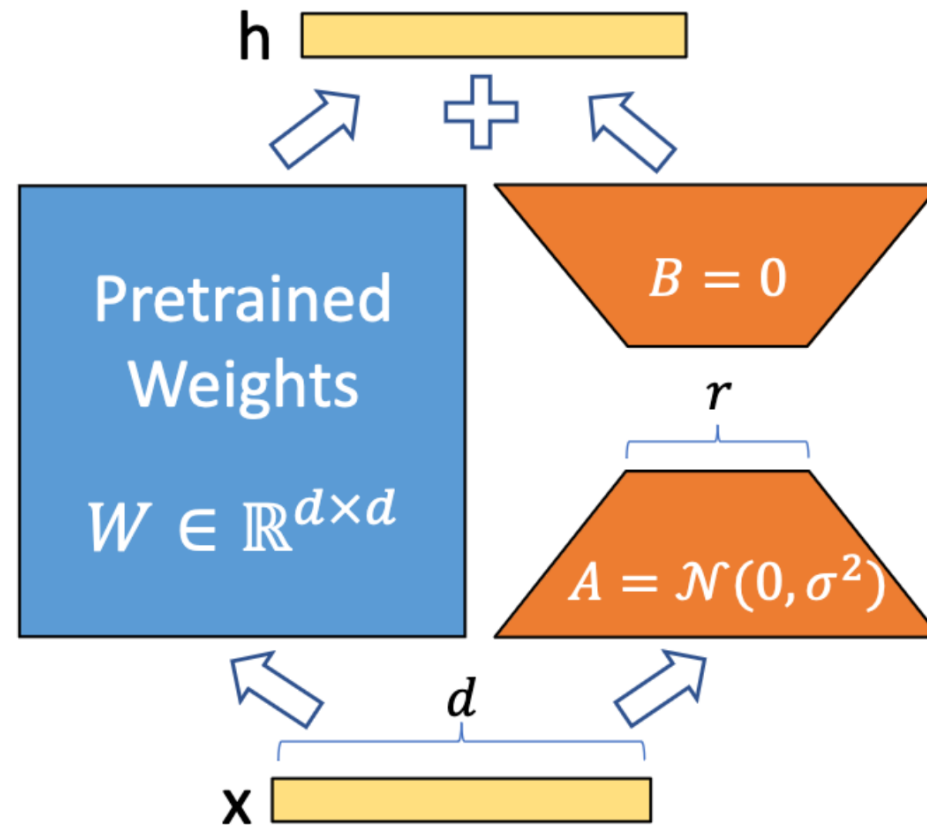


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder + Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x , we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i . For final prediction y , we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

[“Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks” Lewis et al. \(NeurIPS 2020\)](#)

Low Rank Adaptation (LoRA)



(Hu et al ICLR 2022) <https://arxiv.org/abs/2106.09685>

Q-LoRA

- Works with frozen 4-bit quantized pre-trained model
- Can train low-rank adapters on a 65B-parameter model on a single 48GB GPU
- Reduces memory requirement from $> 780\text{GB}$ to $< 48\text{GB}$ without damaging predictive performance or run-time
- Wildly popular in industry

Soft Prompt Tuning

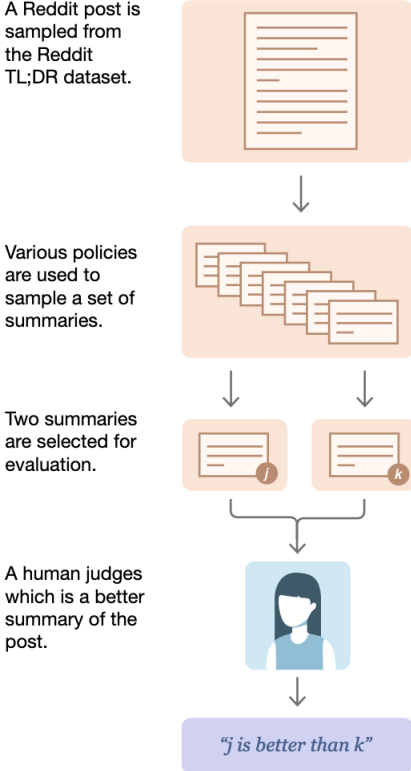
- “Hard prompts” must consist of actual (discrete) tokens
- “Soft prompts” are tunable embeddings that are learned via SGD
- Extreme form of what we saw with LoRA, keep most parameters frozen, train only a small number of learnable parameters

Instruction Tuning

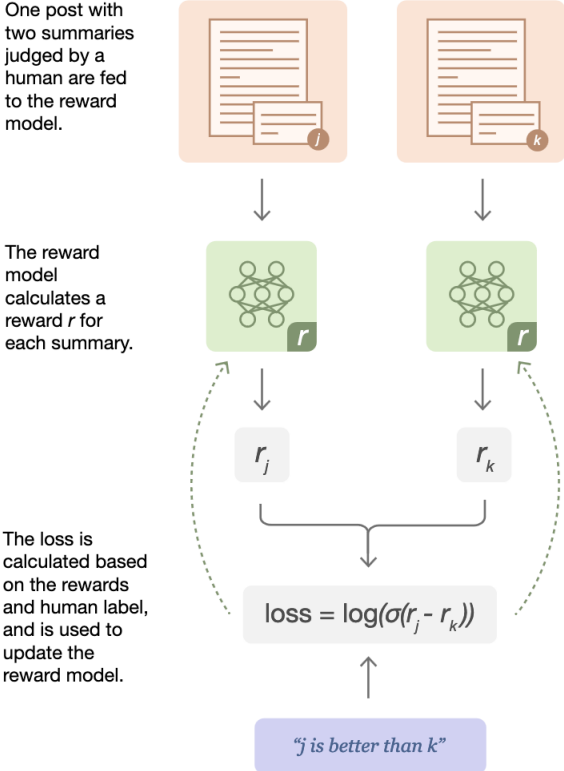
- “Starting with a set of labeler-written prompts and prompts submitted through the OpenAI API, we collect a dataset of labeler demonstrations of the desired model behavior, which we use to fine-tune GPT-3 using supervised learning”
- “In human evaluations on our prompt distribution, outputs from the 1.3B parameter InstructGPT model are preferred to outputs from the 175B GPT-3, despite having 100x fewer parameters.”

RLHF

1 Collect human feedback



2 Train reward model



3 Train policy with PPO

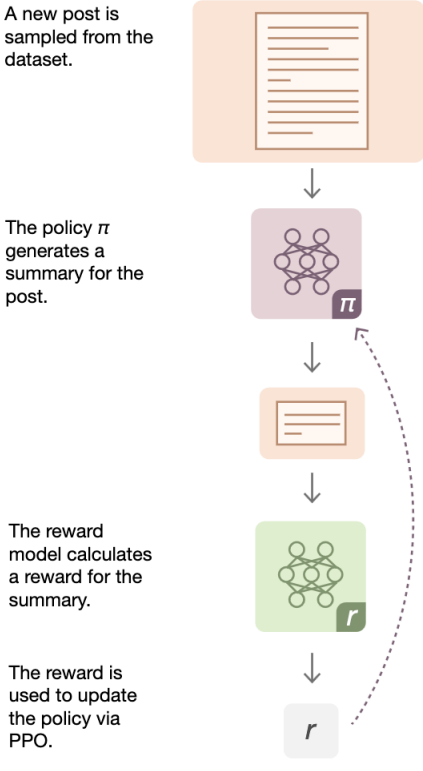


Figure 2: Diagram of our human feedback, reward model training, and policy training procedure.

[“Learning to summarize from human feedback” Stiennon, Ouyang et al.](#)

RLHF Details

- Authors “collect a large, high-quality dataset of human comparisons between summaries
- “Train a model to predict the human-preferred summary”
- “Use that model as a reward function to fine-tune a summarization policy using reinforcement learning”
- Uses a KL penalty to keep RL policy close to supervised model

$$R(x, y) = r_{\theta}(x, y) - \beta \log[\pi_{\phi}^{\text{RL}}(y|x) / \pi^{\text{SFT}}(y|x)]$$

RLHF Key Result

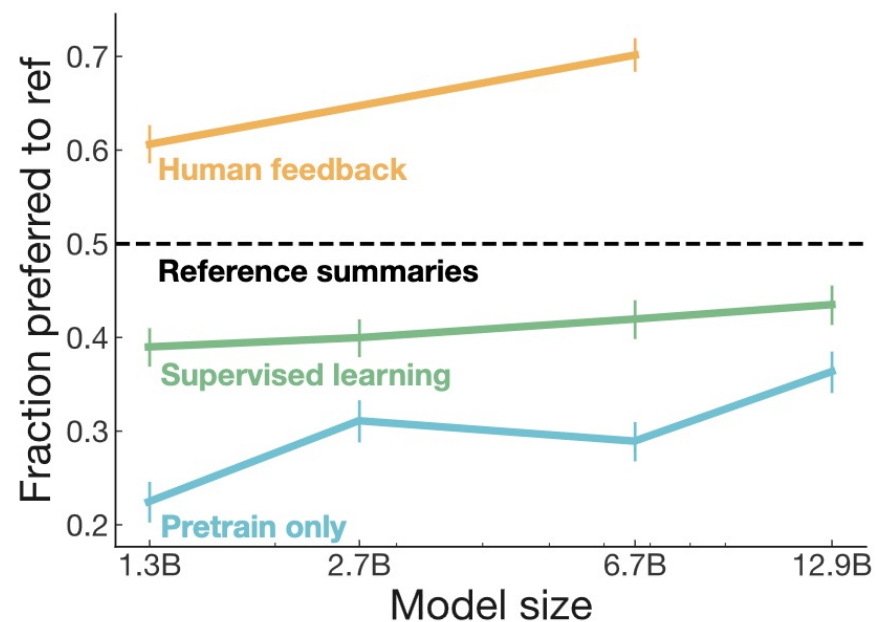
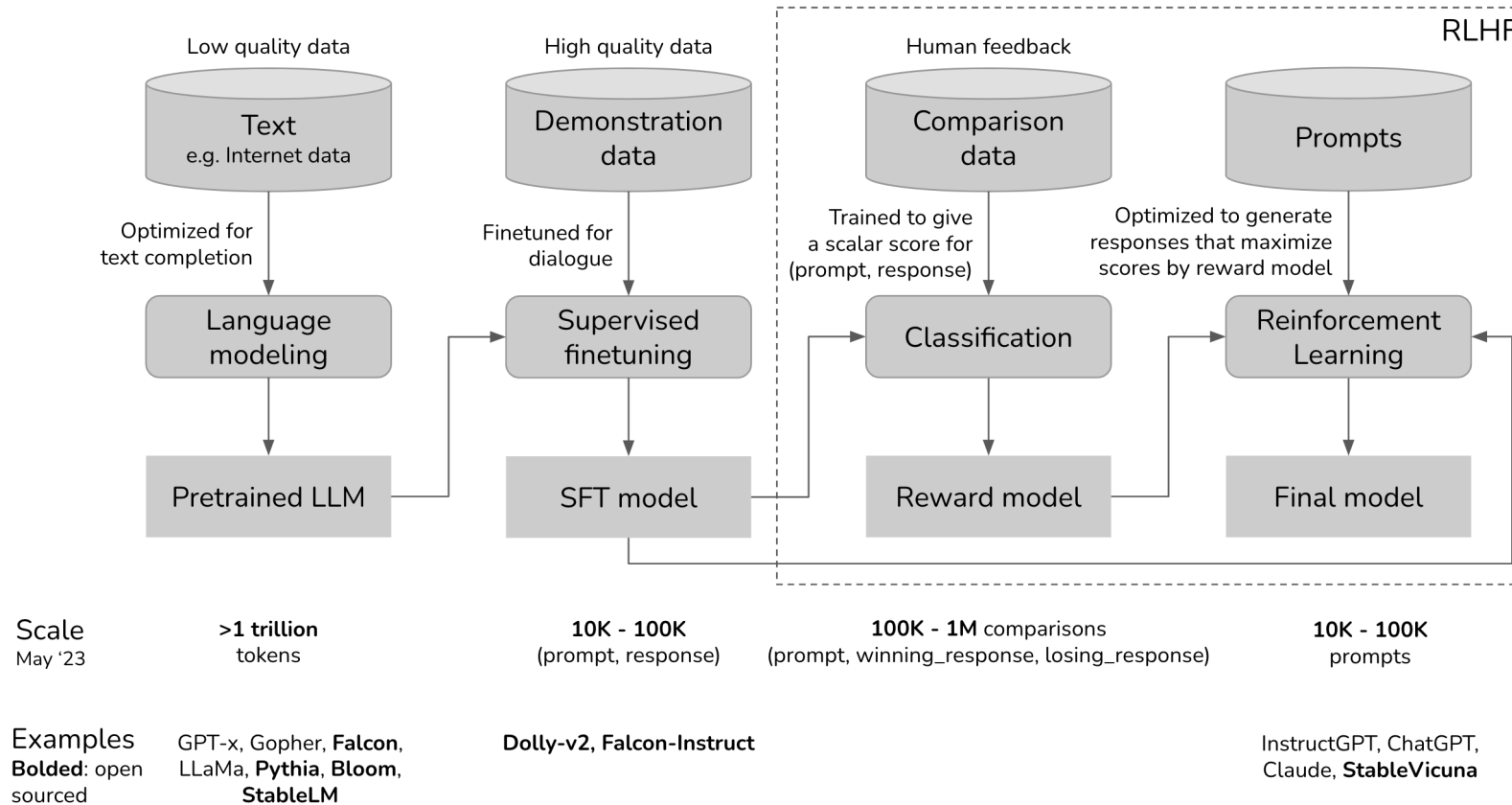


Figure 1: Fraction of the time humans prefer our models' summaries over the human-generated reference summaries on the TL;DR dataset.⁴Since quality judgments involve an arbitrary decision about how to trade off summary length vs. coverage within the 24-48 token limit, we also provide length-controlled graphs in Appendix F; length differences explain about a third of the gap between feedback and supervised learning at 6.7B.

Putting it all together: modern LLMs



img src: <https://huyenchip.com/2023/05/02/rlhf.html>