UNICAMP

Article Presentation

Student: Marcus Vinícius Borela de Castro

Language models are unsupervised multitask learners

By: Radford, Alec, et al. *OpenAI blog* 1.8 (2019)
Link

miro

Extra Study Sources

- GPT-2: Language Models are Unsupervised Multitask Learners - YouTube (Yannic Kilcher)
- https://www.youtube.com/watch?v=9kT0XLPyHBg
- The Illustrated GPT-2 (Visualizing Transformer Language Models) – Jay Alammar
- GPT2-based Next Token Language Model - AllenNLP - Demo

Main concepts

Objective

Current systems are better characterized as narrow experts rather than competent generalists. We would like to move towards more general systems which can perform many tasks – eventually without the need to manually create ad label a training dataset for each one.

Its suspicion is that the prevalence of single task training on single domain datasets is a major contributor to the lack of generalization observed in current systems.

Multitask and meta-learning settings
a general system should be able to perform many different tasks, even for the same input, it should condition not only on the input but also on the task to be performed. That is, it should model p(output/input; task)

Tokenization used

A BPE-like
Preventing from merging across character categories for any byte sequence, but adding an exception for spaces (which significantly improves the compression)

Language modeling is usually framed as unsupervised distribution estimation from a set of examples (x1; x2; :::; xn) each composed of variable length sequences of symbols (s1; s2; :::; sn)

This masking is often implemented as a matrix called an attention mask. Think of a sequence of four words ("robot must obey orders", for example). In a language modeling scenario, this sequence is absorbed in four steps – one per word (assuming for now that every word is a token). As these models work in batches, we can assume a batch size of 4 for this toy model that will process the entire sequence (with its four steps) as one batch.

In a masked self attention way

Example	Features position: 1	2	3	4	Labels
1	robot	must	obey	orders	must
2	robot	must	obey	orders	obey
3	robot	must	obey	orders	orders
4	robot	must	obey	orders	<eos>

by Jay Alammar (The Illustrated GPT-2)

four different sizes of GPT-2 models

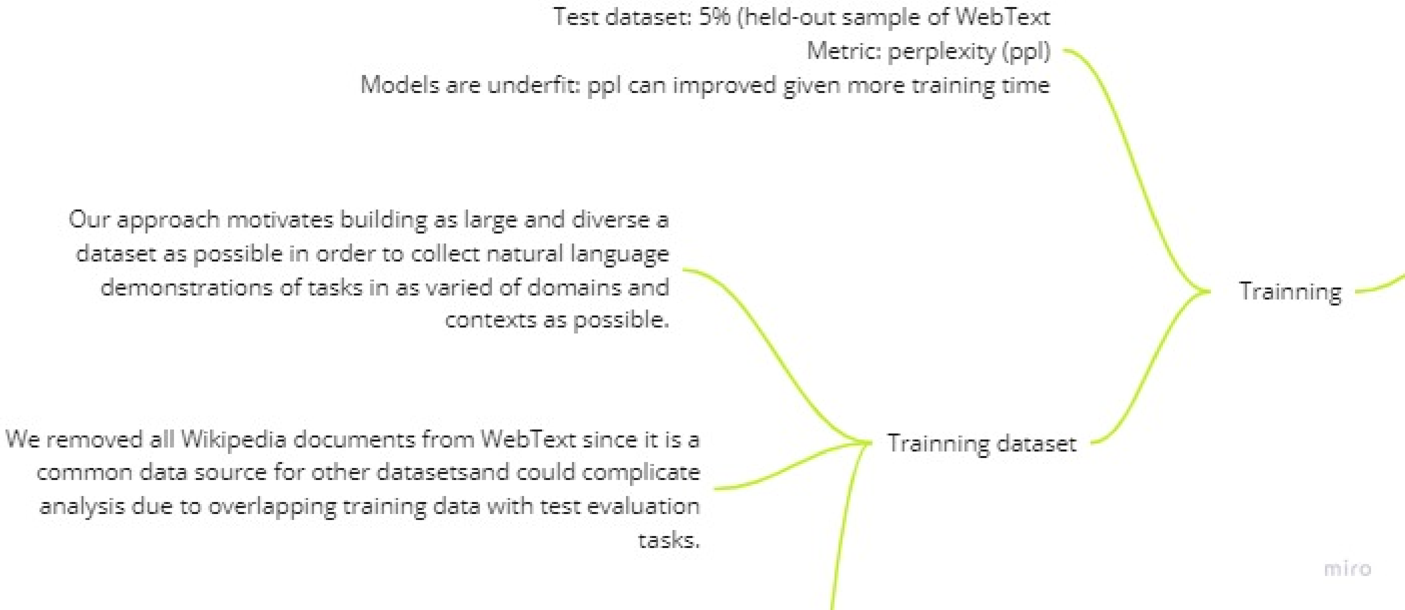
Model Size	Parameters	Layers	dmodel
117M	117 million	12	768
345M	345 million	24	1024
762M	762 million	36	1280
1542M	1542 million	48	1600

The second one is similar in number of parameters to BERT. The first, to GPT-1.

The model used

Based on the decoder of Transformer (Vaswani et al., 2017), with its sel-attention layers. The model largely follows the details of the OpenAI GPT model (Radford et al., 2018) with some modifications as an increased context size from from 512 to 1024 tokens

by Jay Alammar (The Illustrated GPT-2)


Test dataset: 5% (held-out sample of WebText
Metric: perplexity (ppl)
Models are underfit: ppl can improved given more training time
Our approach motivates building as large and diverse a dataset as possible in order to collect natural language demonstrations of tasks in as varied of domains and contexts as possible.
We removed all Wikipedia documents from WebText since it is a common data source for other datasetsand could complicate analysis due to overlapping training data with test evaluation tasks.
Trainning dataset
Trainning
miro

Article contribution

- It demonstrates language models (if it is large and if it is trained on a sufficiently large and diverse dataset) can perform down-stream tasks in a zero-shot setting without any parameter or architecture modification.
- It produces the WebText "cleaned" dataset (8 million documents for a total of 40 GB of text)

Interesting/unexpected results

It achieves promising, competitive, and state of the art results depending on the task.

SOTA in 7 out of 8 tested language modeling task datasets

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	**21.8**
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**	65.85	1.16	1.17	37.50	75.20
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**	47.33	1.01	**1.06**	26.37	55.72
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**	**40.31**	**0.97**	**1.02**	22.05	44.575
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**	**35.76**	**0.93**	**0.98**	**17.48**	42.16

Table 3. Zero-shot results on many datasets. No training or fine-tuning was performed for any of these results. PTB and WikiText-2 results are from (Gong et al., 2018). CBT results are from (Bajgar et al., 2016). LAMBADA accuracy result is from (Hoang et al., 2018) and LAMBADA perplexity result is from (Grave et al., 2016). Other results are from (Dai et al., 2019).

Analysis suggests that data overlap between WebText training data and specific evaluation datasets provides a small but consistent benefit to reported results

However, for most datasets we do not notice significantly larger overlaps than those already existing between standard training and test sets,

On reading comprehension the performance of GPT-2 is competitive with supervised baselines in a zero-shot setting. However, on other tasks such as summarization, while it is qualitatively performing the task, its performance is still only rudimentary according to quantitative metrics.

Basic (or advanced) doubts that may arise

what are de-tokenizers

De-tokenizers are techniques or processes used to reverse the process of tokenization in natural language processing. Tokenization is the process of breaking down a sequence of text into smaller units called tokens, and it is often used as a preprocessing step before applying machine learning models to natural language processing tasks. However, tokenization can introduce artifacts or inconsistencies that can negatively affect the performance of the model.

De-tokenizers remove or undo the tokenization process by combining the tokens back into their original format. This allows the model to better capture the natural language patterns and nuances present in the original text. The use of de-tokenizers can result in improved performance of language models, and the text mentions that the use of invertible de-tokenizers can still calculate the log probability of a dataset, allowing for a simple form of domain adaptation. (by ChatGPT)

The text reports that the use of de-tokenizers resulted in gains of 2.5 to 5 perplexity for GPT-2, which indicates an improvement in the model's ability to predict the next word in a sequence. (by ChatGPT)