Pre-training + Fine-tuning Paradigm, II

LING 574 Deep Learning for NLP

Shane Steinert-Threlkeld

Announcements

- HW5 ref code available
- HW6: PyTorch directly, no more edugrad (but same API :))
  - 1.7.1 docs
- A note on runtime and scalability of LSTMs
  - Definitely fast on small data, but doesn't scale
  - From hw6.model.LSTM.forward:
  - [NB: hw7 will use torch native LSTMs, a bit faster]

```
# [batch_size, hidden_dim]
hidden, memory = self.init_hidden_and_memory(batch_size)
hidden_states = []
for timestep in embeddings:
    # apply the recurrence
    # [batch_size, hidden_dim]
    hidden, memory = self.cell(hidden, memory, timestep)
    # store the hidden state
    hidden_states.append(hidden)
# [seq_len, batch_size, hidden_dim]
return torch.stack(hidden_states)
```

Some Python/PyTorch Notes

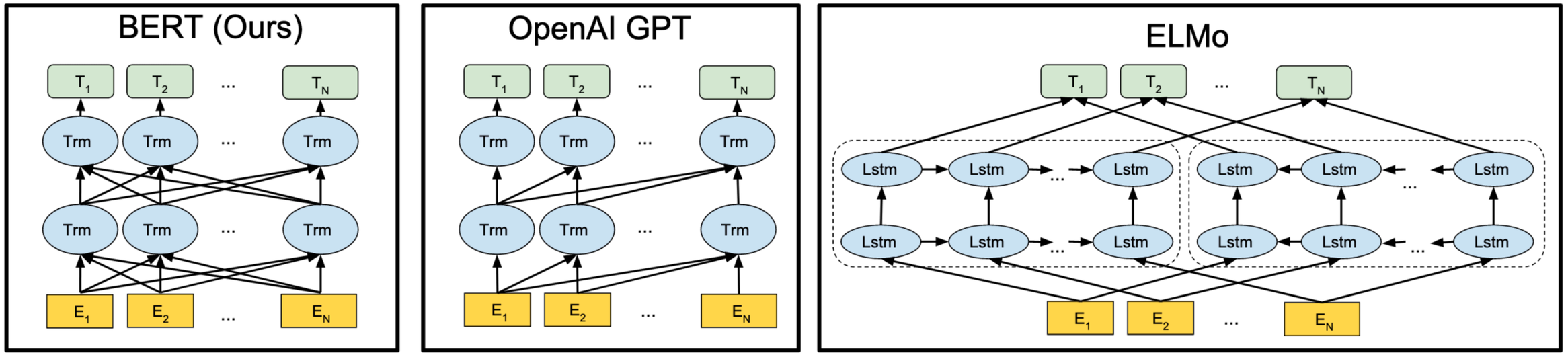
- Training loop in HW6: saves best model based on dev loss, evaluates that
- model.eval(), model.train()
  - Sets the “mode” of a model in PyTorch
  - .eval() turns off things like Dropout, which are only used in training; .train() turns them back on
- `with torch.no_grad():`
  - For *inference/test* mode: doesn’t allocate memory for gradients on Tensors, saves lots of memory
- RNNClassifier.forward: fancy-ish tensor operations to extract the “last” hidden state from a batch of variable-length sequences

Pre-trained Transformers: Decoder-only

GPT(2)

- Generative Pre-training
  - Radford et al 2018; 2019 (GPT2); Brown et al 2020 (GPT3)
- Uses Transformer *decoder* instead of *encoder*
  - "Self"-attention: masked so that only can attend to *previous* tokens
  - Pure LM training objective
  - Can be used for text generation
- GPT: same params as BERT-BASE; GPT2 much bigger; GPT3 muuuuuch bigger (175B params)
- Training data: crawled from outbound Reddit links w/ >3 karma, not public

Comparison



Source: BERT paper

Samples from GPT2 (Cherry-picked)

SYSTEM PROMPT (HUMAN-WRITTEN)	*A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.*
MODEL COMPLETION (MACHINE-WRITTEN, FIRST TRY)	The incident occurred on the downtown train line, which runs from Covington and Ashland stations. In an email to Ohio news outlets, the U.S. Department of Energy said it is working with the Federal Railroad Administration to find the thief. “The theft of this nuclear material will have significant negative consequences on public and environmental health, our workforce and the economy of our nation,” said Tom Hicks, the U.S. Energy Secretary, in a statement. “Our top priority is to secure the theft and ensure it doesn’t happen again.” The stolen material was taken from the University of Cincinnati’s Research Triangle Park nuclear research site, according to a news release from Department officials. The Nuclear Regulatory Commission did not immediately release any information.

Samples from GPT2 (Cherry-picked)

SYSTEM PROMPT (HUMAN-WRITTEN)	*Recycling is good for the world.* *NO! YOU COULD NOT BE MORE WRONG!!*
MODEL COMPLETION (MACHINE-WRITTEN, 25 TRIES)	Recycling is NOT good for the world. It is bad for the environment, it is bad for our health, and it is bad for our economy. I'm not kidding. Recycling is not good for the environment. It is destructive to the earth and it is a major contributor to global warming. Recycling is not good for our health. It contributes to obesity and diseases like heart disease and cancer. Recycling is bad for our economy. It increases the cost of a product, and in turn, the price of everything that is made with that product. Recycling is not good for our nation. We pay a tremendous price for the privilege of having the world's most advanced and efficient recycling system. Recycling is a huge, colossal waste of time, energy, money, and resources. And THAT is why we need to get back to basics and get back to basics in our recycling efforts. One of the best ways to start is to look at the process of creating a paper product. When you make a paper product, it is basically a long chain of materials. Everything from the raw materials (wood, cardboard, paper, etc.), to the reagents (dyes, solvents, etc.) to the printing equipment (chemicals, glue, paper, ink, etc.), to the packaging, to the packaging materials (mercury, chemicals, etc.) to the processing equipment (heating, cooling, etc.), to the packaging materials, to the packaging materials that are shipped overseas and to the packaging materials that are used in the United States. Each step along the way creates tons of waste that we constantly have to clean up. The process

Write With Transformer

- Prompt: **Studying computational linguistics is a good idea. There are many interesting problems.**

- However, there are many bad problems. The research into computational linguistics has not helped very much in the development of better solutions. One problem with computer science and machine learning is that, until recently, the problems they were supposed to solve were insufficiently well studied. Most of the people who are doing research in computer science (at least in my profession) have no understanding of how to design a particularly natural problem, for example. In the case of the reasoning problem above, most of the people I know who work on computer science or machine learning have no understanding of how to make computer programs work in ways which do not have unavoidable adverse consequences. In the best cases, computational linguistics was little help.

Write With Transformer

- Prompt: **I didn't ride my bike to campus today because of the snowy weather.**

- There is a lot of construction and I wanted to have a few hours of skiing. All that snow is a real pain in the ass. Good thing there was an LRT planned for that area. See you guys on campus in a few days! Photo credit: Tapto@archivosos.gr – August 26, 2016: Windy Weather on the Mountain (photo credit: Diana Prat)
  From Diana Prat, via tapto@archivosos.gr:
  Ever since the camp that I organized to protest the modernisation of the Barranco, my beautiful alma mater, you know, going in for a press conference, I was asking why the Italian government

What do you want to ask GPT-2?

0

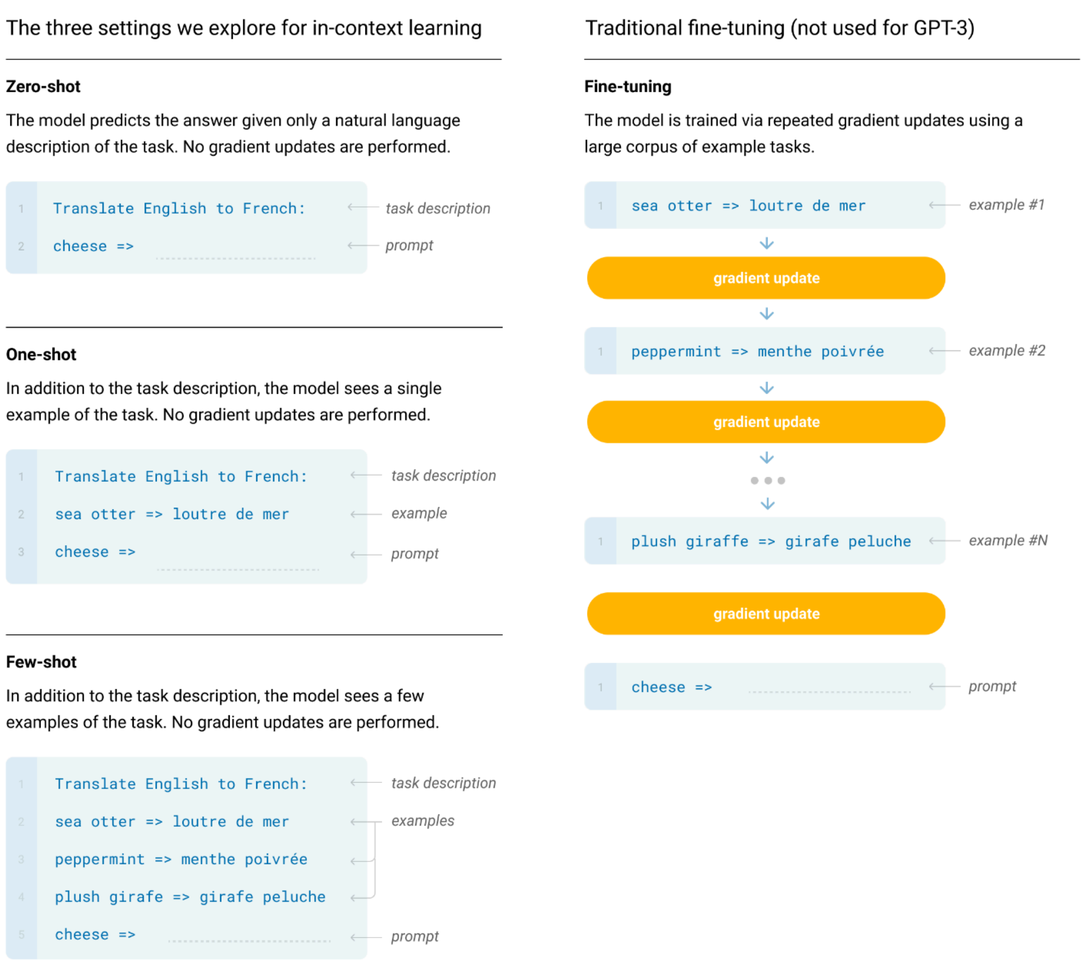
Nobody has responded yet.

Hang tight! Responses are coming in.

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

GPT3

- Same approach: pure Transformer decoder trained on LM
  - Scale: 175B params
  - Data size: ~500billion tokens, majority from filtered Common Crawl
- Few-shot “fine-tuning” paradigm:
  - Prompt with a few examples, ask to continue
  - *No parameter updates*



GPT3 Few-Shot Results

	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	**89.0**	**91.0**	**96.9**	**93.9**	**94.8**	**92.5**
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0

	WiC Accuracy	WSC Accuracy	MultiRC Accuracy	MultiRC F1a	ReCoRD Accuracy	ReCoRD F1
Fine-tuned SOTA	**76.1**	**93.8**	**62.3**	**88.2**	**92.5**	**93.3**
Fine-tuned BERT-Large	69.6	64.6	24.1	70.0	71.3	72.0
GPT-3 Few-Shot	49.4	80.1	30.5	75.4	90.2	91.1

k=32

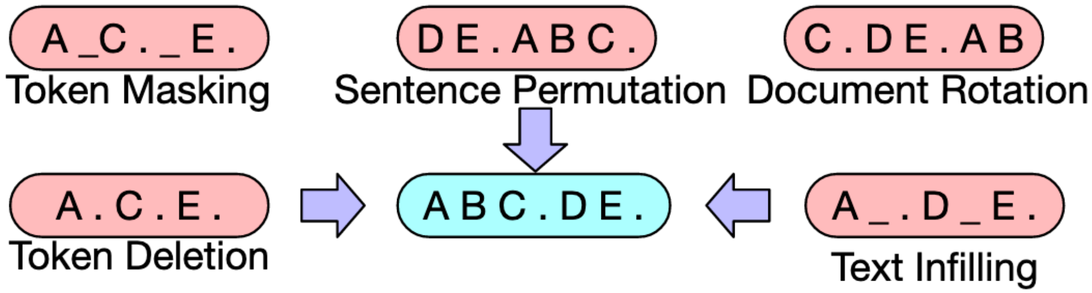
Some follow-ups on GPT3

- Has ushered in a lot of work on “prompt tuning”: how to best engineer the prompts to produce the behavior that you want
  - Very useful survey paper/website on that front: http://pretrain.nlpedia.ai/
- Putting the “open” back in:
  - EleutherAI: “A grassroots collective of researchers working to open source AI research.”
    - Reproduce GPT-like models + datasets in entirely open way
  - OPT-175B: Meta’s first open (incl logbook, etc) non-commercial replication
  - More now (varying degrees): Mosaic, Mistral, …

Pretrained Transformers: Encoder-Decoder

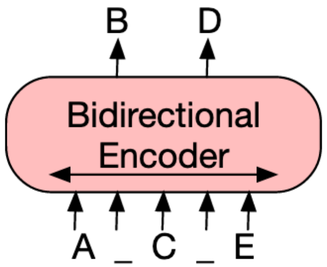
BART

- Full Transformer, i.e. encoder-decoder transducer
  - Many composable transformations of raw text, presented to encoder
  - Goal of decoder is to reconstruct the original text

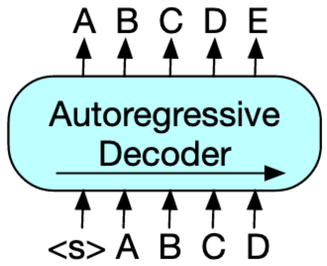


- Good for both discrimination and generation

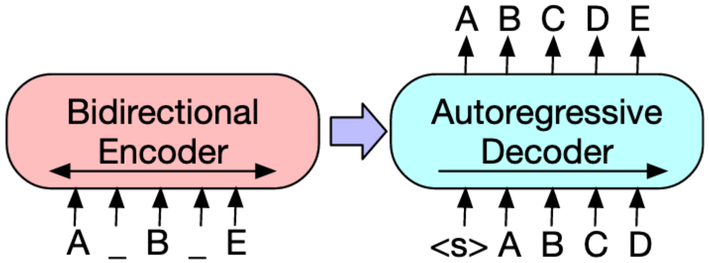
High-level Overview



(a) BERT: Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.



(b) GPT: Tokens are predicted auto-regressively, meaning GPT can be used for generation. However words can only condition on leftward context, so it cannot learn bidirectional interactions.



(c) BART: Inputs to the encoder need not be aligned with decoder outputs, allowing arbitary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder, and we use representations from the final hidden state of the decoder.

Comparison of Pre-training Objectives

Model	SQuAD 1.1 F1	MNLI Acc	ELI5 PPL	XSum PPL	ConvAI2 PPL	CNN/DM PPL
BERT Base (Devlin et al., 2019)	88.5	**84.3**	-	-	-	-
Masked Language Model	90.0	83.5	24.77	7.87	12.59	7.06
Masked Seq2seq	87.0	82.1	23.40	6.80	11.43	6.19
Language Model	76.7	80.1	**21.40**	7.00	11.51	6.56
Permuted Language Model	89.1	83.7	24.03	7.69	12.23	6.96
Multitask Masked Language Model	89.2	82.4	23.73	7.50	12.39	6.74
BART Base						
w/ Token Masking	90.4	84.1	25.05	7.08	11.73	6.10
w/ Token Deletion	90.4	84.1	24.61	6.90	11.46	5.87
w/ Text Infilling	**90.8**	84.0	24.26	**6.61**	**11.05**	5.83
w/ Document Rotation	77.2	75.3	53.69	17.14	19.87	10.59
w/ Sentence Shuffling	85.4	81.5	41.87	10.93	16.67	7.89
w/ Text Infilling + Sentence Shuffling	**90.8**	83.8	24.17	6.62	11.12	**5.41**

Advantages of Encoder-Decoder Models

- "Best of both worlds"
  - On a par with RoBERTa on NLU / discrimination tasks
  - State-of-the-art on many generation tasks (e.g. summarization)
- Others:
  - MASS
  - T5
    - uses labeled data
    - "Unified" text-to-text format

Source Document (abbreviated)	BART Summary
The researchers examined three types of coral in reefs off the coast of Fiji ... The researchers found when fish were plentiful, they would eat algae and seaweed off the corals, which appeared to leave them more resistant to the bacterium Vibrio coralliilyticus, a bacterium associated with bleaching. The researchers suggested the algae, like warming temperatures, might render the corals' chemical defenses less effective, and the fish were protecting the coral by removing the algae.	Fisheries off the coast of Fiji are protecting coral reefs from the effects of global warming, according to a study in the journal Science.
Sacoolas, who has immunity as a diplomat's wife, was involved in a traffic collision ... Prime Minister Johnson was questioned about the case while speaking to the press at a hospital in Watford. He said, "I hope that Anne Sacoolas will come back ... if we can't resolve it then of course I will be raising it myself personally with the White House."	Boris Johnson has said he will raise the issue of US diplomat Anne Sacoolas' diplomatic immunity with the White House.

Multilingual Pre-training

- One other main dimension: *mono-* vs *multi*-lingual pre-training
  - Roughly: concatenate (in fancy way) corpora from many languages, then do the same kind of pre-training
  - *Much more info* from Agatha's lecture on May 15

	Encoder-only	Decoder-only	Encoder-decoder
English-only *	BERT, RoBERTa, XLNet, ALBERT, …	GPT-n	BART
Multilingual	mBERT, XLM(-R), …	BLOOM (HF BigScience), XGLM	mBART, MASS, mT5

Limitations of Pre-training + Fine-tuning

State of the Field

- Manning 2017: “The BiLSTM Hegemony”
- Right now: “The pre-trained Transformer Hegemony”
  - By default: fine-tune a large pre-trained Transformer on the task you care about
  - Will often yield the best results
  - Beware: often not significantly better than *very simple* baselines (SVM, etc)

Scale scale scale

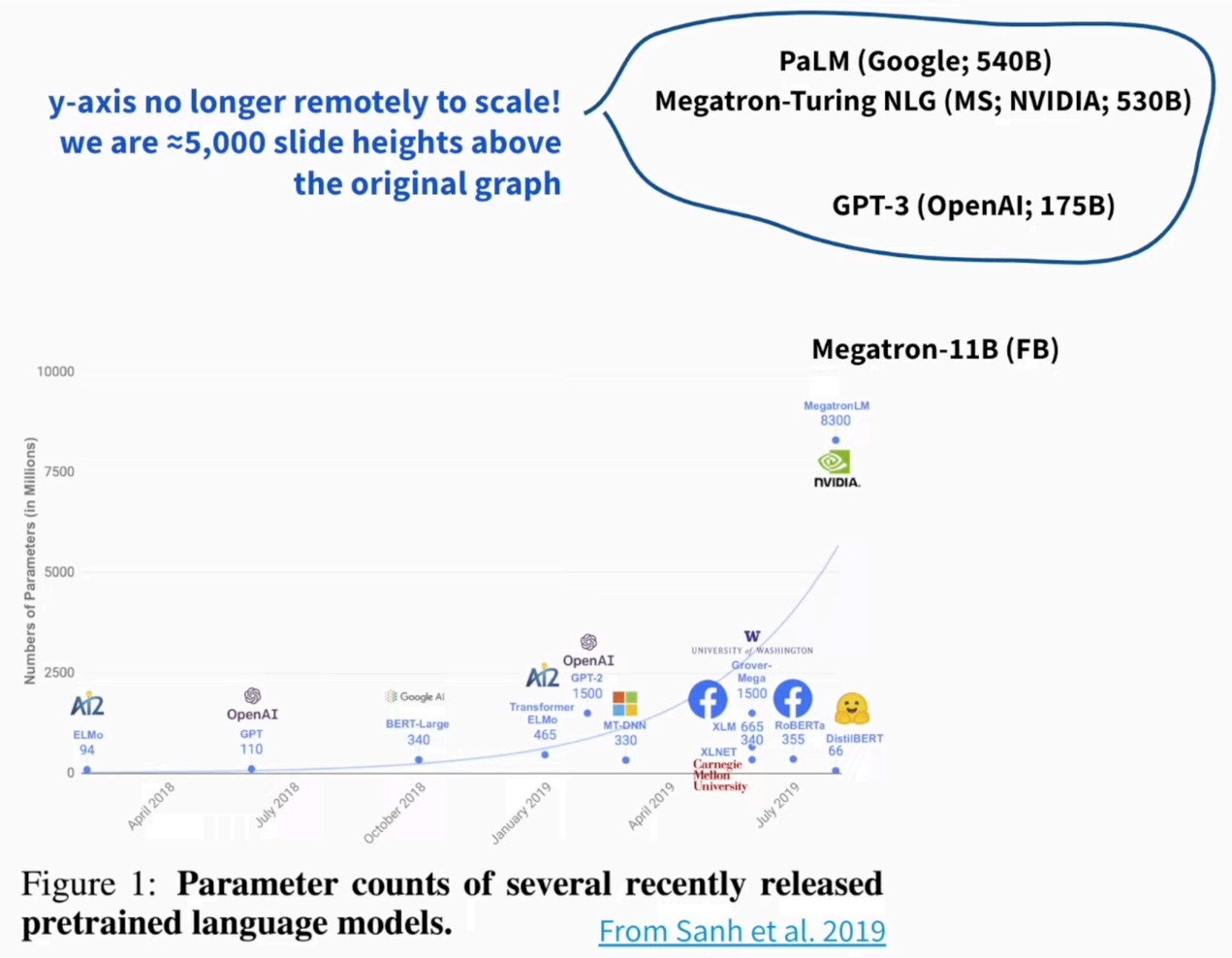


Figure 1: **Parameter counts of several recently released pretrained language models.** From Sanh et al. 2019

source

Note on the costs of LMs

Note on the costs of LMs

- Currently something of an ‘arms race’ between e.g. Google, Facebook, OpenAI, MS, Baidu, …

Note on the costs of LMs

- Currently something of an ‘arms race’ between e.g. Google, Facebook, OpenAI, MS, Baidu, …
- Hugely expensive
  - Carbon emissions
  - Monetarily
    - Inequitable access
  - Dataset debt/documentation

Note on the costs of LMs

- Currently something of an ‘arms race’ between e.g. Google, Facebook, OpenAI, MS, Baidu, …
- Hugely expensive
  - Carbon emissions
  - Monetarily
    - Inequitable access
  - Dataset debt/documentation

Energy and Policy Considerations for Deep Learning in NLP

Emma Strubell **Ananya Ganesh** **Andrew McCallum**
College of Information and Computer Sciences
University of Massachusetts Amherst
{strubell, aganesh, mccallum}@cs.umass.edu

Abstract

Recent progress in hardware and methodology for training neural networks has ushered in a new generation of large networks trained on abundant data. These models have obtained notable gains in accuracy across many NLP tasks. However, these accuracy improvements depend on the availability of exceptionally large computational resources that necessitate similarly substantial energy consumption. As a result these models are costly to train and develop, both financially, due to the cost of hardware and electricity or cloud compute time, and environmentally, due to the carbon footprint required to fuel modern tensor

Consumption	**CO_2e (lbs)**
Air travel, 1 person, NY↔SF	1984
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000
Training one model (GPU)	
NLP pipeline (parsing, SRL)	39
w/ tuning & experiments	78,468
Transformer (big)	192
w/ neural arch. search	626,155

Table 1: Estimated CO_2 emissions from training common NLP models, compared to familiar consumption.[1]

Note on the costs of LMs

- Currently something of an ‘arms race’ between e.g. Google, Facebook, OpenAI, MS, Baidu, …
- Hugely expensive
  - Carbon emissions
  - Monetarily
    - Inequitable access
  - Dataset debt/documentation

Note on the costs of LMs

- Currently something of an ‘arms race’ between e.g. Google, Facebook, OpenAI, MS, Baidu, …
- Hugely expensive
  - Carbon emissions
  - Monetarily
    - Inequitable access
  - Dataset debt/documentation

Green AI

Roy Schwartz* ♢ Jesse Dodge*♢♣ Noah A. Smith♢♡ Oren Etzioni♢

♢Allen Institute for AI, Seattle, Washington, USA
♣ Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
♡ University of Washington, Seattle, Washington, USA

July 2019

Abstract

The computations required for deep learning research have been doubling every few months, resulting in an estimated 300,000x increase from 2012 to 2018 [2]. These computations have a surprisingly large carbon footprint [40]. Ironically, deep learning was inspired by the human brain, which is remarkably energy efficient. Moreover, the financial cost of the computations can make it difficult for academics, students, and researchers, in particular those from emerging economies, to engage in deep learning research.

This position paper advocates a practical solution by making **efficiency** an evaluation criterion for research alongside accuracy and related measures. In addition, we propose reporting the financial cost or “price tag” of developing, training, and running models to provide baselines for the investigation of increasingly efficient methods. Our goal is to make AI both greener and more inclusive—enabling any inspired undergraduate with a laptop to write high-quality research papers. Green AI is an emerging focus at the Allen Institute for AI.

More on the Costs of LMs

On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? 🦜

Emily M. Bender*
ebender@uw.edu
University of Washington
Seattle, WA, USA

Timnit Gebru*
timnit@blackinai.org
Black in AI
Palo Alto, CA, USA

Angelina McMillan-Major
aymm@uw.edu
University of Washington
Seattle, WA, USA

Shmargaret Shmitchell
shmargaret.shmitchell@gmail.com
The Aether

ABSTRACT

The past 3 years of work in NLP have been characterized by the development and deployment of ever larger language models, especially for English. BERT, its variants, GPT-2/3, and others, most recently Switch-C, have pushed the boundaries of the possible both through architectural innovations and through sheer size. Using these pretrained models and the methodology of fine-tuning them for specific tasks, researchers have extended the state of the art alone, we have seen the emergence of BERT and its variants [39, 70, 74, 113, 146], GPT-2 [106], T-NLG [112], GPT-3 [25], and most recently Switch-C [43], with institutions seemingly competing to produce ever larger LMs. While investigating properties of LMs and how they change with size holds scientific interest, and large LMs have shown improvements on various tasks (§2), we ask whether enough thought has been put into the potential risks associated with developing them and strategies to mitigate these risks.

- For more on the reactions to this paper: https://faculty.washington.edu/ebender/stochasticparrots.html

Some Reasons to Pause

- Leaderboard chasing (via larger models and more data) funnels research activity into one specific and limited goal
  - Amplifies harmful biases
  - Equity costs
  - Climate costs
  - Data documentation debt
  - Does not promote human-like linguistic generalization (Linzen 2020 summary)
- More from Angelina McMillan-Major on May 20 on stochastic parrots paper

🤗 Transformers

https://huggingface.co/transformers

Overview of the Library

- Access to many variants of many very large LMs (BERT, RoBERTa, XLNET, ALBERT, T5, language-specific models, …) with fairly consistent API
  - Build tokenizer + model from string for name or config
  - Then use just like any PyTorch nn.Module
- Emphasis on ease-of-use
  - E.g. low barrier-to-entry to *using* the models, including for analysis
- Interoperable with PyTorch or TensorFlow 2.0

Example: Tokenization

```
>>> from transformers import AutoTokenizer

>>> tokenizer = AutoTokenizer.from_pretrained("bert-base-cased")
```

```
>>> encoded_input = tokenizer("Do not meddle in the affairs of wizards, for they are subtle and quick to
>>> print(encoded_input)
{'input_ids': [101, 2079, 2025, 19960, 10362, 1999, 1996, 3821, 1997, 16657, 1010, 2005, 2027, 2024, 1125
 'token_type_ids': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 'attention_mask': [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]}
```

See http://juditacs.github.io/2019/02/19/bert-tokenization-stats.html (h/t Naomi Shapiro)

Example: Tokenization

```
>>> from transformers import AutoTokenizer

>>> tokenizer = AutoTokenizer.from_pretrained("bert-base-cased")
```

```
>>> encoded_input = tokenizer("Do not meddle in the affairs of wizards, for they are subtle and quick to
>>> print(encoded_input)
{'input_ids': [101, 2079, 2025, 19960, 10362, 1999, 1996, 3821, 1997, 16657, 1010, 2005, 2027, 2024, 112
 'token_type_ids': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 'attention_mask': [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]}
```

```
>>> tokenizer.decode(encoded_input["input_ids"])
'[CLS] Do not meddle in the affairs of wizards, for they are subtle and quick to anger. [SEP]'
```

See http://juditacs.github.io/2019/02/19/bert-tokenization-stats.html (h/t Naomi Shapiro)

Example: Tokenizing a Batch

```
>>> batch_sentences = [
...     "But what about second breakfast?",
...     "Don't think he knows about second breakfast, Pip.",
...     "What about elevensies?",
... ]
>>> encoded_input = tokenizer(batch_sentences, padding=True)
>>> print(encoded_input)
{'input_ids': [[101, 1252, 1184, 1164, 1248, 6462, 136, 102, 0, 0, 0, 0, 0, 0, 0],
               [101, 1790, 112, 189, 1341, 1119, 3520, 1164, 1248, 6462, 117, 21902, 1643, 119, 102],
               [101, 1327, 1164, 5450, 23434, 136, 102, 0, 0, 0, 0, 0, 0, 0, 0]],
 'token_type_ids': [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]],
 'attention_mask': [[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0],
                    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                    [1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]]}
```

Example: Tokenizing a Batch

```
>>> batch_sentences = [
...     "But what about second breakfast?",
...     "Don't think he knows about second breakfast, Pip.",
...     "What about elevensies?",
... ]
>>> encoded_input = tokenizer(batch_sentences, padding=True)
>>> print(encoded_input)
{'input_ids': [[101, 1252, 1184, 1164, 1248, 6462, 136, 102, 0, 0, 0, 0, 0, 0, 0],
               [101, 1790, 112, 189, 1341, 1119, 3520, 1164, 1248, 6462, 117, 21902, 1643, 119, 102],
               [101, 1327, 1164, 5450, 23434, 136, 102, 0, 0, 0, 0, 0, 0, 0, 0]],
 'token_type_ids': [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]],
 'attention_mask': [[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0],
                    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                    [1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]]}
```

Add `return_tensors="pt"` to get these outputs as PyTorch Tensors

Example: Forward Pass

```
>>> from transformers import BertTokenizer, BertModel
>>> import torch

>>> tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
>>> model = BertModel.from_pretrained("bert-base-uncased")

>>> inputs = tokenizer("Hello, my dog is cute", return_tensors="pt")
>>> outputs = model(**inputs)

>>> last_hidden_states = outputs.last_hidden_state
```

More on HuggingFace

- Main library: https://huggingface.co/transformers
- Model repository (w/ search, tags, etc): https://huggingface.co/models
- Datasets: https://huggingface.co/datasets
- …