ECE 1786 Lecture #9

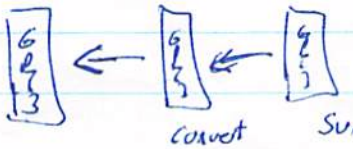
Last Day: Proposal Presentations → Project Launch

Work-in-Flight: Assignment A4 (due tonight), Project → Progress Report (due Nov 21)

Today:
0. Using GPT-3 in project.
1. How GPT3 was made better at zero-shot/few shot
2. GPT3 API - different features.
3. Relationship to your project
4. Progress Report
5. Software Development & Github Repositories

If you've got to the last part of Assignment 4, or caught the essence of previous demonstrations, you can see how powerful GPT-3, (model text-davinci-002) is!
- it might even do everything you wanted done in your project, with just a single zero-shot prompt!
- you should/are free to try that. → for classifier and generate.
- if it turns out to work perfectly that is both good (but might not.) to know [because then this model makes others obsolete, except...]
- you'll need to do more for your project, perhaps
  1. Do something more sophisticated using, maybe more GPT3 'passes'; try to make smaller prompt; or break into pieces & 'pipeline'
  2. Try to get close to this with smaller/cheaper/faster models → fine tune GPT2
     - fine-tune a smaller GPT-3
     → compare among several.

(GPT-3 is painfully big & cost $ to use)



- show Richard Tang example

English is the new programming language.

Why is GPT3 text-davinci-002 so good? Are the other big (175B parameter) models as good?
→ NO: show demos (see next pages 9-2 a-d)
- the other big models don't seem to get the goal as clearly & clearly as GPT-3. (all these big models)
- I believe they all were trained essentially as described in lecture 5 & you experienced in A3.

- However, OpenAI did something more that seems quite important, covered in a March 2022 paper by Ouyang et.al "Training language models to follow instructions with human feedback"
→ goal was to get the model to generate completions much closer to the 'human intent'
→ they use the word 'alignment' in the field: Align what was wanted with what the model does
- This is a summary of what they did (tons of details in paper):

① Fine-tuned "old" GPT-3 with human-created zero shot examples (prompts & completions) → called SFT "supervised fine tuning" - not good enough

② Created a classifier that measures the "goodness" of a given prompt + completion. rates

③ Used ② as a reward function in a reinforcement learning context to further fine-tune SFT. (said another way: used ② as an objective/loss function in further fine tuning)

-will describe ①②③ in more detail because is interesting - extended fine tuning

9-24.

Cohere - Toronto start up

co:here

Dashboard Documentation Playground Community

Generate Embed Classify

Export code Share

PRESETS

EXAMPLE PRESETS

Content Creation
- Blog Posts
- Email Copy
- Hashtag Generator
- Product Descriptions

Summarization
- Chat Summarization
- Article Summarization

Paraphrasing
- Spelling & Grammar Check
- Correct Errors in Voice to Text Tran...

Information Extraction
- Extract Entities from Legal Agreem...
- Extract Entities from Invoices

Suggestions for project ideas in a course called Creative Applications of Natural Language Processing: **Using the Classroom as an Experimental Lab.**

Bibliography

Books

Journal articles

See also

Computational linguistics
Corpus linguistics
Computational statistics
Corpus-based computational linguistics
Data mining
Dialectology
Digital humanities
Digital literacy

Language documentation
Lexicography
Lexical semantics
Linguistic geography
Linguistic typology
Metaphor
Natural language processing
NLP: Using the Classroom as an Experimental Lab

Phonetics
Pragmatics
Psycholinguistics
Semantics
Sociolinguistics
Speech recognition
Syntax
Text analytics
Text corpus
Text mining

Notes

Further reading

PARAMETERS

MODEL: xlarge-20220609 (xlarge)

NUMBER OF TOKENS: 160

TEMPERATURE: 0.7

STOP SEQUENCES: -- ×

TOP-K: 0

TOP-P: 1

FREQUENCY PENALTY: 0

PRESENCE PENALTY: 0

CONTROL PANEL

Generate

Clear all

Save

9-26

huggingface.co/bigscience/bloom?text=Suggestions+for+project+ideas+in+a++

PapernotRSC · MiBot · Learn Python in Y Minutes · Bazen Papers · LutherQinDocs · NFREF · ECE 1786 · ECE 243 · 2022/3 Undegrad Capstone · Massey Senior-Junior · ECE Iris · Learning · TCAIREM · CBL

Hugging Face | Search models, datasets, users... | Models | Datasets | Spaces | Docs | Solutions | Pricing

Hugging Face is way more fun with friends and colleagues! Join an organization | Dismiss this message

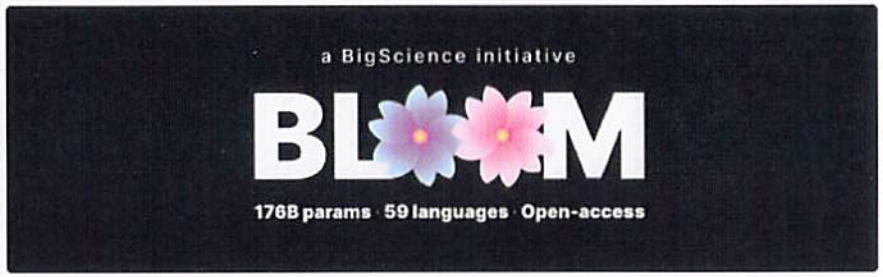
bigscience / **bloom** ♡ like 1.68k

Text Generation · PyTorch · TensorBoard · Transformers · 46 languages · arxiv:1909.08053 · arxiv:2110.02861 · arxiv:2108.12409 · doi:10.57967/hf/0003 · bloom

feature-extraction · Eval Results · License: bigscience-bloom-rail-1.0

Model card | Files and versions | Training metrics | Community 138 | Deploy | Use in Transformers

Edit model card



BigScience Large Open-science Open-access Multilingual Language Model
Version 1.3 / 6 July 2022

Current Checkpoint: **Training Iteration 95000**

Total seen tokens: **366B**

Model Details

BLOOM is an autoregressive Large Language Model (LLM), trained to continue text from a prompt on vast amounts of text data using industrial-scale computational resources. As such, it is able to output coherent text in 46 languages and 13 programming languages that is hardly distinguishable from text written by humans. BLOOM can also be instructed to perform text tasks it hasn't been explicitly trained for, by casting them

Downloads last month
12,512

Hosted inference API

Text Generation · Groups · Examples

Suggestions for project ideas in a course called Creative Applications of Natural Language Processing: "find a good phrase/sentence and create a book about it".

A:

Look at

sampling | greedy | BLOOM prompting tips

Switch to "greedy" for more accurate completion e.g. math/history/translations (but which may be repetitive/less inventive)

Compute ⌘+Enter 0.9

Powered by AzureML | The model is loaded and running on Intel Xeon Ice Lake CPU

JSON Output | Maximize

Spaces using bigscience/bloom 53

huggingface/bloom_demo

ysharma/Talk_to_Multilingual_AI_WhisperBloomCoqui

Meta/facebook

92c

Free, Unlimited OPT-175B Text Generation

Warning: This model might generate something offensive. No safety measures are in place as a free service.

W Fact | Chatbot | Airport Code | Translation | Cryptocurrency | Code | Math

Suggestions for project ideas in a course called Creative Applications of Natural Language Processing:

Generate

Suggestions for project ideas in a course called Creative Applications of Natural Language Processing: - Sentiment analysis of reddit comments - Machine translation of reddit comments - Sentiment classification of reddit comments I think I'll go with the last one.

> Sentiment classification of reddit comments I think you're going to need a lot of data for that.

I'm sure there's a lot of data on reddit.

Like the results? ☆ Support Alpa development by staring Alpa on GitHub

Screenshot | Tweet it! #alpa

9-24

beta.openai.com/playground

PapernotRSC MiBot Learn Python in Y Minutes Bazen Papers LutherQinDocs NFREF ECE 1786 ECE 243 2022/3 Undegrad Capstone Massey Senior-Junior ECE Iris Learning TCAIREM CBL

Overview Documentation Examples Playground

Help University of Toronto

Get started

Enter an instruction or select a preset, and watch the API respond with a completion that attempts to match the context or pattern you provided.

You can control which model completes your request by changing the model.

KEEP IN MIND

- Use good judgment when sharing outputs, and attribute them to your name or company. Learn more.
- Requests submitted to our models may be used to train and improve future models. Learn more.
- Our default models' training data cuts off in 2021, so they may not have knowledge of current events.

Playground

Load a preset... Save View code Share ...

Suggestions for project ideas in a course called Creative Applications of Natural Language Processing:

1. Develop a part-of-speech tagger for a low-resource language.

2. Build a machine translation system for a low-resource language pair.

3. Develop a question answering system for a low-resource language.

4. Create a chatbot for a low-resource language.

5. Develop a text summarization system for a low-resource language.

Mode

Model: text-davinci-002

Temperature: 0.7

Maximum length: 256

Stop sequences
Enter sequence and press Tab

Top P: 1

Frequency penalty: 0

Presence penalty: 0

Best of: 1

Inject start text

Inject restart text

Show probabilities: Off

Submit

17

Step 1: hire humans to create good examples of inpts & outputs.
- recall that these are "zero-shot" english specifications or "few shot" examples.
- terminology: call these ↑ "prompts" call the desired outpts "completions"
- so humans were hired online & trained to write good prompts + completions

Prompt:
Create a shopping list from this recipe:
Trim the ends off zucchini. Cut zucchini in half lengthwise; scoop out pulp, leaving 1/2-in. shells. Finely chop pulp. In a skillet, cook beef, zucchini pulp, onion, mushrooms and peppers over medium heat until meat is no longer pink; drain. Remove from the heat. Add 1/2 cup cheese, ketchup, salt and pepper; mix well. Spoon into the zucchini shells. Place in a greased 13x9-in. baking dish. Sprinkle with remaining cheese.

human
Labeler demonstration
zucchini, beef, onion, mushroom, peppers, cheese, ketchup, salt, pepper

Jumping ahead in this lecture here is what the original GPT-3 does with prompt & what the new (they called it instruct-GPT for awhile) does:

GPT-3 175B completion:
Bake, uncovered, at 350° for 20-25 minutes or until zucchini is tender and cheese is melted.

InstructGPT 175B completion:
Bake, uncovered, at 350° for 20-25 minutes.

- 8 zucchini squash, ends trimmed
- 1 pound ground beef
- 1 cup finely chopped zucchini pulp
- 1 small onion, chopped
- 8 ounces diced mushrooms
- 1 sweet red or green pepper, diced
- 1/2 cup grated cheese
- 1/4 cup ketchup
- Salt and pepper to taste
- 1 cup shredded cheese

Figure 49: Labeler-written prompt from our dataset, along with the human-written demonstration, and completions from GPT-3 175B and InstructGPT175B. Prompt is lightly cherry-picked (5 selected from 15 to show a diverse range of tasks), and the completions are not cherry-picked.

- I don't know if the quantities make sense.

Other examples of prompts (templates) 9-3a.

Use Case	Example
brainstorming	What are 4 questions a user might have after reading the instruction manual for a trash compactor? {user manual} 1.
brainstorming	What are 10 science fiction books I should read next?
classification	Take the following text and rate, on a scale from 1-10, how sarcastic the person is being (1 = not at all, 10 = extremely sarcastic). Also give an explanation {text} Rating:
classification	This is a list of tweets and the sentiment categories they fall into. Tweet: {tweet_content1} Sentiment: {sentiment1} Tweet: {tweet_content2} Sentiment: {sentiment2}
classification	{java code} What language is the code above written in?
classification	You are a very serious professor, and you check papers to see if they contain missing citations. Given the text, say whether it is missing an important citation (YES/NO) and which sentence(s) require citing. {text of paper}
extract	Extract all course titles from the table below: | Title | Lecturer | Room | | Calculus 101 | Smith | Hall B | | Art History | Paz | Hall A |
extract	Extract all place names from the article below: {news article}
extract	Given the following list of movie titles, write down any names of cities in the titles. {movie titles}
generation	Write a creative ad for the following product to run on Facebook aimed at parents: Product: {product description}

Continued on next page

- a total of 12,000 prompt-completions were paid for. (alot!)

- these were used to "fine-tune" GPT-3 originally
  - which means just running the "predict-the-next-word" training that you learned in assignment 3
  - with some extra cleverness to select the exact model (using step 2 as a judge, for example).

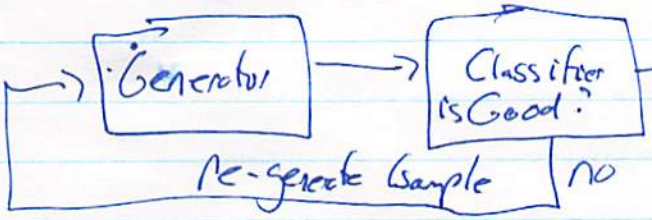
→ This model is called Supervised Fine Tuning (SFT) & is the basis for next steps.
- perspective: 12,000 - alot of work → too much work, need to automate
⇒ Automated Fine Tuning.

Step 2: Train a separate classifier model that takes Prompt + Completion as input and produces a rating of the quality (of the Completion given the Prompt) as output

Ask: Who is doing generation?

~~Extent~~ Rating from 1-7. (Likert: strongly disagree ↔ strongly agree)

- this might be relevant to all generation projects in this course; ~~getting enough~~
→ such a classifier, if you had one could be used for:
① Measure how good the generator is! (as I've mentioned)
② To act as a filter on outputs, i.e.:



③ To create labelled data that could be used to help train the generator model in a different way than fine tuning → a different loss function!

→ the classifier was built/trained from
(i) A "smaller" GPT-3 (6B parameter)
(ii) Another human-labelled dataset
- collected prompt + completions (good and bad!)
- asked humans to label 1→7.

⇒ trained (i) → as a classifier with two completions for every prompt

⇒ used the difference in human labels to train output to know which completion would be preferred by human.

this detail may not matter to overall idea.

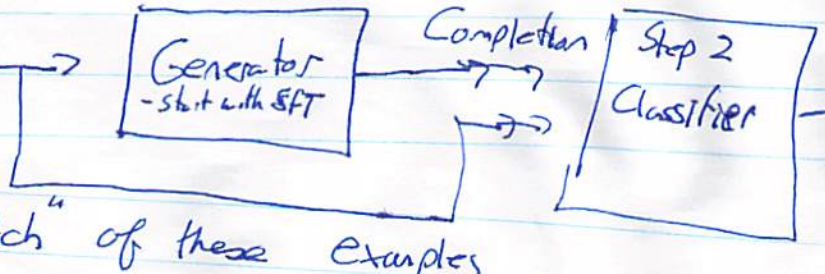
→ model output is a number → higher means better

→ Table 6 shows size of training dataset: ≈ 33,000 prompts
- that's alot of work! each produces multiple (≈14) completions that are human labelled

Step 3: Use the classifier together with the generator to iteratively generate example prompt + completions, label them, & then train the generator to be better
- using the labels to compute loss function. ← what to set

→ they frame this as a reinforcement learning problem but it is probably easier to think of it this way:

3a: Create more prompt + completions & label them:



- generate a "batch" of these examples

3b: for each example compute a loss function and, for the batch, back-propagate that loss through the Generator.

- keep iterating [→ 3a → 3b] until done (happy enough with loss?).

One key aspect of the loss function

loss = f(prompt, completion, label, model outputs, model outputs of SFT)

→ loss increases when the label indicates bad

→ loss increases when the model outputs change too much compared to the SFT model outputs.

→ this is an interesting & important subtlety; without it, the model loses some of its capabilities as measured on standard datasets such as SQUAD, DROP, HellaSwag & WMT 2015 (translation)

Using this version of GPT-3 is both exciting but perhaps difficult → don't have direct access to the model, like we do with GPT-2

However, OpenAI provides 'API' access to the model that let's you do specific things you might have done with GPT-2 +!.
See: beta.openai.com/docs/introduction.

- many project-related useful things here.

① Prompt Design - docs/guides/completion/prompt-design
- show/discuss - e.g. show & tell.; classification
- new insertion & editing

② Fine Tuning docs/guides/fine-tuning
/api-reference/fine-tunes
- lots of case studies
- analyzing
Advanced. - validation data → run.
- Hyperparameters → #epochs, batch size, learning rate multiplier,
- 3¢/1000 tokens training. } on best model then
- 12¢/ " " inference }

See me if $ an issue. (≤ $50)

③ Embeddings: Text in Embedding out
- might be good quality
- look at the use cases
- recommendations, search, clustering

④ Moderation docs/guides/moderation/overview
- classifier that checks text for compliance with OpenAI content policy.

Other items

① Project Progress Report due Nov 21.
- see next page for specification that it should follow

② S/W Developed using Git / Github.com ("Source code control")
- each team now has Github repository.
- you're required to use it.
- both team members should commit & push code.
- unfamiliar with this - see lecture slides posted under lecture 9 on Quercus.
  → this is how all modern software is made when 2 or more people collaborate. (well even just 1)!

Progress Report Description

The project progress report is a check-in to show that you are on track to complete your project. By the project progress date, you should have made good progress on:

- Collecting almost all the data
- Producing a baseline model
- Producing at least one result, including one qualitative or quantitative comparison
- Reflected upon the feedback given at proposal time

The report document demonstrates your progress. The document has a word limit of maximum of 1000 words.

Some of the sections are similar to your project proposal. You may find that when you look at your previous writing a second time, that you find ways of expressing your ideas more concisely.

The word limit is hard: There is a 1% penalty for every word in excess of the 1000 limit. Please count the words in your document, compute the penalty, and put it on the front page. These are not included in the word count, nor are pictures or references.

There is a penalty-free grace period of one hour past the deadline. Any work that is submitted between 1 hour and 24 hours past the deadline will receive a 20% grade deduction. No other late work is accepted.

The progress report should have the following sections:

Introduction

- Give a clear (re)statement of the goal of your project, making use of the feedback you received on the proposal, and adjustments since the proposal.

Data Processing

- Describe the data that you have collected and cleaned to date. Be clear and specific when describing what you've done, so that a classmate can reproduce your work. If at all possible, show some statistics about your cleaned data (e.g. number of examples in each class), and at least one example of a cleaned training data. Since no plan ever survives first contact with reality, this section will probably be different from what you wrote in your proposal.

Baseline Model

- Briefly describe (again) your baseline model that you created to compare with your neural network. This may have evolved from your proposal, so indicate what has changed if anything.

Architecture

- Give a description of the best model architecture that you have built so far, and how you got to this point. This description should be more detailed than in your initial proposal. In particular, you should provide a rough idea of how complex your model is (e.g. number of layers, number of parameters), and what someone will have to do to reproduce a model similar to yours.

Result

- At least one result or comparison between your working model and your baseline. You are not measured on how well your model is performing at this point. For some problems, you will need a qualitative measure of the baseline rather than a quantitative one. Quantitative measures are preferred, but if you can make a case for a qualitative comparison, that's okay too.

Discussion

- Discuss your results, including at least one set of training curves if applicable, or otherwise use some other metrics. Do you think your model is performing well? Base your discussion on both the results that you have shown, and the interpretation of your training curve. What issues, particular to your project, will you have to overcome?

Team Work and Progress

- Describe how well your team is working together. Take a look at the divided tasks and deadlines you set earlier. How is each person doing? What has each person accomplished?