ECE 1786 Lecture #9

Last Day: Proposal Presentations → Project Launch

Work-in-Flight: Assignment 4 → Project → Progress Report
due tonight

Today:
1. How GPT 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 davinci-02) is!
- it might even do everything you wanted done in your project, with just a single zero-shot prompt!
- you should be free to try that, not for classifier and general.
- if it turns out to work perfectly, that is both good 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 smarter prompt; or break into pieces & 'pipeline'
  2. Try to get close to this with smaller/cheaper/faster models → fine tune GPT-2
     - fine-tune a smaller GPT-3
- compare among several.
  (GPT-3 is painfully big & cost $ to use)

- show Richard this paper. 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 page) 9-2 a-d
- the other big models don't seem to get the goal as clearly & cleanly as GPT-3 (all the 175B 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 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.
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 1,2,3 in more detail because is interesting
BLOOM is an open-access, large language model (LLM), trained to continue text from
languages that is fairly distinguishable from text written by humans. BLOOM can also
be instructed to perform text tasks it hasn’t been explicitly trained for, by editing them
as such, it is able to output coherent text in 47 languages and 7 domains.

A model on vast amounts of text data, using industrial-scale computational resources,
BDI Becomes Large Open-Source Multilingual Language Model

Model Details

Total seen tokens: 3.66B
Current Checkpoint: Training Iteration 9500

Version 1.3 / 6 July 2022

BLOOM 1.3B Parameters, 5 Languages, Open-Source

Highking Face is way more fun with friends and colleagues!

See more options...
Like the results? Support Alpha development by starring Alpha on GitHub.

Sure, there’s a lot of data on Reddit. I think you’re going to need a lot of data for that.

- Sentiment classification of Reddit comments
- Machine translation of Reddit comments
- Sentiment analysis of Reddit comments
- Sentiment analysis of Reddit comments

Suggestions for project ideas in a course called "Creative Applications of Natural Language Processing"
Suggestions for project ideas in a course on applied 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.

Our current output in 2023 is a few models trained on various datasets.

The features submitted to our models may be used to train new models.

Use good judgment when sharing.

Keep in mind:

- You can control whether the model's output contains the context of what you provided.
- The model is not trained to produce comprehensive answers to multiple questions.
- Enter an instruction or select a prompt.

Get Started
Step 1: hire humans to create good examples of inputs & outputs.
- recall that these are "zero-shot" English specificities or "few-shot" examples.
- terminology: call these "prompts" call the desired outputs "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.

Labeled 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 InstructGPT 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.
### Use Case | Example
--- | ---
brainstorming | What are 4 questions a user might have after reading the instruction manual for a trash compactor?

{user manual}

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
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}

<table>
<thead>
<tr>
<th>Rating:</th>
</tr>
</thead>
</table>
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 | 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:

<table>
<thead>
<tr>
<th>Title</th>
<th>Lecturer</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculus 101</td>
<td>Smith</td>
<td>Hall B</td>
</tr>
<tr>
<td>Art History</td>
<td>Paz</td>
<td>Hall A</td>
</tr>
</tbody>
</table>

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. (Cool!)
- these were used to "fine-tune" GPT-3 original
- 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)

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. (Strongly disagree)

- 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.

```
Generator \[\rightarrow\] Classifier [isGood?]
\[\rightarrow\] produce output
\[\rightarrow\] re-generate sample
```

- To create labelled data that could be used to help train the model in a different way than fine-tuning (by a different loss function)
- the classifier was built/trained from
  (i) A "smaller" GPJ-3 (68 parametered)
  (ii) Another human-labelled dataset
      - collected prompt + completions (good and bad)
      - asked humans to label 1-77.

⇒ trained (ii) with two completions for every prompt
⇒ used the difference in human labels to train output to know which completion would be preferred by human.

⇒ model output is a number ⇒ higher means better
⇒ Table 6 shows size of training dataset: 33,000 prompts
  - that's a lot of work! Each produces multiple 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.
⇒ 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:

Select Prompt Randomly
From set of 33,000

- 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 \rightarrow 3b \rightarrow 367 until done (happily crash with loss?).

One key aspect of the loss function

\[
\text{loss} = f(\text{prompt}, \text{completion}, \text{label}, \text{model output}, \text{model output of SF7})
\]

loss increases when the label indicates bad.

loss increases when the model output changes too much compared to the SF7 model outputs.

this is an interesting and important subtlety: without it, the model loses some of its capabilities as measured on standard datasets such as SQUAD, DROP, ItellusABC, \& 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 lets 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  
   - should discuss — e.g., show & tell; classification
   - new insertion & editing

2. Fine Tuning  
   - docs/guides/fine-tuning
   - api-reference/fine-tuning
   - lots of case studies
   - analyzing
   - advanced: validation data — run
   - hyperparameters — *epochs, batch size, learning rate multiplier, 3 * 1000 tokens training, 12 * 1000 inference
   - see me if there are issues

3. Embeddings: Text in Embedding out
   - might be good quality
   - look at the use cases
   - recommendations, search, clustering
4) Moderation - uses a classifier that checks text for compliance with OpenAI content policy.

Other items

   - See next page for specification that it should follow.

2) S/W Development 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.
   - If 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?