NLP Module: Description for Teachers (UT Austin / Institute for Machine Learning)

Overview  This document describes an outreach module intended to introduce high school students to basic concepts from natural language processing (NLP). NLP is a discipline around building systems to solve problems involving human language input, such as question answering, text summarization, machine translation (i.e., Google Translate), and more. Specifically, we discuss language models, including BERT and GPT-3, which power many recent advances at big tech companies and have received lots of attention both in the NLP research community and the media.¹ This module starts from the basics, giving students an understanding of the underlying concepts that power these models via a coding exercise, then uses a web demo² to give them an opportunity to play around with powerful neural network models.

The main model presented that students will actually implement is a bigram language model, which models the probability distribution $P(\text{next word} \mid \text{current word})$. For instance, if you see the word to, maybe France has probability 0.1, Spain has probability 0.08, and so on and so forth. Such models can be used for tasks like predictive text on a phone.

We introduce the concepts of basic probability (including conditional probability) and estimating probabilities from data. Learning probabilities for this model does not involve complicated algorithms, but instead just an intuitive counting scheme: if we see France as the next word after to 10% of the time that to shows up in our dataset, then we set $P(\text{France} \mid \text{to}) = 0.1$. Implementing and querying this model largely uses standard math and data structures, but the data structures to store all the quantities needed (particularly counts of neighboring pairs of words) may be a little tricky for students just starting out. There are several levels of programming assignments available depending on students’ experience levels.

Goals

1. Expose students to basic concepts from natural language processing and machine learning, including principles of basic probability and learning models on training data
2. Give students a programming challenge involving querying data structures, basic loop constructs, and mathematical operations
3. Give students hands-on experience with state-of-the-art language models via a web demo
4. Challenge students to look at language data and think critically about what the pros and cons of these models are
5. Give students pointers to other resources to follow up on these topics

What this module contains

10 videos (approx. 80 minutes), 2 exercises (40+ minutes)

- Introductory content about NLP, machine learning, and high-level ideas of language modeling (3 videos, 19 minutes)
- In-depth about how to implement $n$-gram language models (3 video, 28 mins, including about 10 minutes of code walkthrough)

¹These have been covered extensively in the media; e.g., this article about GPT-2, the predecessor to GPT-3, when it came out: https://www.cnn.com/2019/02/18/tech/dangerous-ai-text-generator
²https://transformer.huggingface.co/doc/distil-gpt2
• Hands-on exercise with n-gram language models (exercise 1; 20 minutes - 2 hours depending on what version)

• Introduction to Write With Transformers (8 minute video)

• Hands-on exercise with web demo and discussion (5 minute video + exercise 2, 10-15 minute)

• Broader applications and where to go next (2 videos, 12 minutes)

Video titles

1. What is NLP?
2. Machine Learning
3. Language Modeling
4. Building n-gram LMs (exercise 1)
5. Bigram LM Code (exercise 1)
6. Querying the LM (exercise 1)
7. Hands-on: Write With Transformers (exercise 2)
8. Discussion: Write With Transformers (exercise 2)
9. Language Models in the News
10. What’s Next?

Exercises

• Exercise 1: Implementing bigram language models. Students will use basic data structures and provided framework code to query a probabilistic model of what the next word in a sentence is likely to be. Framework code reads in data and populates the needed data structures; students primarily implement loop logic to sample sentences from the probability distribution placed by the model. This exercise follows a worksheet.

• Exercise 2: Students play around with the Write With Transformer demo (see URL in footnote on page 1). The video lectures include several questions which students will answer using the web tool.

How to use this module

This module can be tailored to contain different amounts of coding and be suitable for students at different experience levels.

No coding Students can do a very small amount of coding or as little as no coding for exercise 1. See the attached worksheet; they can answer the questions with the implementation given to them.
**Level 1**  The videos are designed for this level. These exercises primarily involve calling methods and loop constructs. Students implement `sample_word` and `get_best_word` in the code.

**Level 2**  This is a more sophisticated version of the exercises that requires students to additionally deal with the complex data structures more directly. Students additionally implement `BigramLanguageModel.get_probability`. Alternatively, they can be asked to implement more of `main.py` to make it more “from scratch.”

Nearly anything in the code could in principle be deleted to force the students to implement. The main thing that would be very challenging for any student to figure out would be the exact set of data structures in `BigramLanguageModel`.

**Code and Platforms**

This module supports both Java and Python versions; the two videos that depend on code specifics have been recorded twice, once for each language. Code for both versions is available either as stand-alone downloads or as web-based projects through repl.it.