MSDS 453 Natural Language Processing

Final Paper on Chatbots

Danish S. Moti

Northwestern University

The problem and the data

As a teacher for more than ten years, I have increasingly seen that students are reluctant to read and engage with the text. One trick that I have found is that a digital presentation of text that initially masks the length and the meaning is a valuable method to get students to engage with the text. I learned this when I gave students a quiz to send in their answers through text message. The following week, the students asked if we could do that "text-thing" again. In essence, they were asking me for another quiz. Learning from this experience, I want to design a chatbot for one of my teaching subjects this year, American Government. I want to encourage students to look up answers to questions they are unsure about, rather than just ignoring that gap in their knowledge.

In this paper, I outline how to use the topics covered in this class to create a chatbot to help students learn quick facts about American politics and history. The business uses go beyond my classroom. In addition, to bring a text-based question-answer bot to all fields of education, I could see this chatbot help train employees and get learning tools to spaces where internet connectivity is limited. In some regions of the world, such as South Asia, cell phones have far greater market penetration than computers. Anecdotally, I have seen that even the poorest people have access to cell phones in South Asia. In use cases like this, a chatbot could help democratize education.

There are many options from where I can draw data for this project. My initial experiments take data from Wikipedia pages and a study guide on the American government with 100 historical facts. I experiment with both because the Wikipedia source is messy and unwieldy. Wikipedia is cumbersome because of the many links, images, and irrelevant (at least for an academic course) information. The Wikipedia corpus is the text pulled from the site after querying "American Government." The study guide is more to the point and directly addresses the classes that I teach. However, the study guide is long enough and still has more information than I need. The oversize nature of the corpus is required to make use of the skills I have learned in this class. It would have been way too simple, unchallenging, and unrewarding to list ten facts where I then try to pull five facts.

Summarize algorithm and key features

The bulk of the chatbot building is done through an NLTK (Natural Language Toolkit), TF-IDF (Term Frequency Inverse Document Frequency) Vectorizer, and measuring cosine similarity scores. The first part of any data science project is to understand the data and apply preprocessing techniques. NLTK is a robust set of libraries that can process natural human language. NLTK is practically the "standard" method used for tokenization (Lane, Howard, & Hapke, 2019, p. 46). We can also use NLTK for stopword removal. However, based on my learnings from Assignment 3 and ontologies, I know stopwords are needed to help maintain relationships between the subject and object pairs.

Table 1 shows the keywords that create edges and linkages. The outputs in Table 1 show that the study guide is a historical document because most links between subject and object are established through "was." One interesting thing to note is the presence of "U" and "S." Initially, I was confused about what was happening here. But then I realized that when we split sentences based on a period (.), the term U.S. will not behave well in that command.

The TF-IDF Vectorizer converts texts into numbers so the computer can read them. The TF-IDF provides a frequency score based on how popular a word is inside a document and flips the score if the word is prevalent in all documents. The vectorizer is where the real magic happens. In the vectorization process, numbers represent the words of a corpus, but weights associated with the word show how relevant that word is in a document. This is a crucial algorithm to preprocess the text.

With the vectors, we can then compute a cosine similarity that compares the query to the corpus. The cosine similarity score plots vectors in three-dimensional space and allows one to view the distance between the vector lines. The closer the lines, the closer the query is to a particular vector. The cosine score is a popular informational retrieval tool that matches the query vector to the corpus's closest vector. This closest vector is what becomes a search result. Machine learning is applied to the vector using the SK-learn python library to output a cosine similarity matrix.

Review algorithmic results

The results for the chatbot are impressive. Immediately, I can get over 75% accuracy (shown in Table 2) in the bot's ability to recall the information and offer the correct answer. The results speak to the power of the machine learning that makes the TF-IDF vectors. Much of the work I have done here is moving towards 85% and 95% accuracy. The easiest part of the programming is to program simple greetings that the bot will respond to. In both hello and goodbye-type greetings, I have written in proper responses triggered by keyword matching.

A sample output of cosine similarity is shown in Table 3. This similarity matrix is made of the first question in the study guide and a subsequent random question. The cosine outputs vary depending on the query and corpus. One more comparison I want to run is between the five questions that will test the chatbot. These questions are as follows;

What is the thirteenth amendment?
What was the Dred Scott decision?
Who was King George?
Who was Sam Adams?
Who wrote the Declaration of Independence?

I want to check these to make sure there is variation in the questions. I believe a more significant variation in the questions will produce a better challenge for the chatbot. Table 4 shows the sparse matrix created by the count-vectorizer. One can imagine this matrix made of the entire text corpus and not just the five questions are displayed. Table 5 shows the cosine similarity between the five questions. The cosine scores show that some questions have zero correlation, while two questions show almost a 50% match. The similar questions are the two "who was" questions. The findings are generally representative of how the rest of the questions in the corpus relate to each other.

A soft cosine measure is an enhanced cosine similarity algorithm. In the soft method, different words with similar meanings have are weighted similarly. For instance, I am talking about the American government, but what if I added questions about the Canadian government? The soft cosine method with machine learning can recognize words that the human mind knows should cluster together. I reran the cosine scores for the five chatbot questions, and I also added two questions on the Canadian government to test the soft cosine. The results are presented in Table 6 and interpreted in the next section.

Interpret results

The vectorized machine learning is reasonably accurate in creating a chatbot, given that one has a suitable corpus. Much of the results are presented from the study guide corpus. The results from the Wikipedia page are less promising because of the noise that is present in that corpus. The noise includes almost a hundred links and images throughout the text. Cleaning that text is so cumbersome that it was better to focus on the study guide corpus. My experience from this paper reflects my experience working with the class corpus on movie reviews, in essence, messy data in messy data out. In my attempts to get the chatbot to 100% accuracy, I experimented with BERT and GLoVE algorithms with little success. I was mostly limited by my experience working with an advanced level of analysis and coding.

Tabe 6 shows soft cosine scores for the five questions that tested the chatbot and two more questions on the Canadian Government. The soft cosine algorithm significantly improves the weights for the five chatbot questions indicating that these questions have a lot more in common than the non-soft measures shown. I think the soft cosine measures are far more accurate in this case because, in our ground truth, we know that these questions are based on the American Government and pulled from one comprehensive study guide. Running genism and fasttext_model300 allows me to run soft cosine similarity. The fasttext word embedding is a more complex algorithm than the vectorized embeddings. The only downside of fasttext is that it requires a 1-gigabit file that was slower to download than other python libraries have been.

Ontologically speaking, we know the ground truth as it relates to the American Government. There is no debating if the chatbot gets the answer right or wrong. To create an ontological picture of what the corpus looks I have created node-entity maps using spaCy. The spaCy library in python can perform named entity recognition to make matching pairs. The ontology and pairs are shown in Table 7. The plotted ontology creates one of the more unique ontologies I have seen. I am not able to explain the giant loop. What makes perfect sense is that most entity relationships stem from the "United States." This also looks like a great tool that could help students study and make visual study aids.

Summary and findings

I am impressed by the robust tools available for natural language analysis. Much of the work that would have seemed impossible, or taken too long to consider doing, can now be done in a few minutes. The NLTK, sklearn, and TF-IDF packages in python have streamlined what can be an impossibly arduous task if it had to be done in person. Despite these advancements, technology cannot outrun the adage, garbage in garbage out. Like in previous assignments, the corpus that we start is perhaps the most crucial step in NLP. My key learning and recommendation for others are to invest the proper time getting and preparing the right corpus and then worry about applying the latest and greatest algorithms.

I am also impressed by the math and modeling that comes from cosine similarity. It makes sense that vectors are plotted in three-dimensional space, and the most similar vectors are lined up closer to each other. These scores can be enhanced with soft cosine similarity, which applies different algorithms to bundle together identical semantics. While these resources are impressive, they still are not ready to take the place of the human mind. While my chatbot correctly answered four out of five questions, there are more advanced methods out there to achieve a perfect score. Even a perfect score could not replace the intuition of a human brain.

Tables

Table 1: Top ten relations from American Government study guide (left) and Wikipedia (right)

was	35
is	20
are	5
guarantees	4
S	3
were	3
wrote	2
v	2
means	2
fired at	2

S	30
is	14
are	10
U	7
has	5
g	4
have	4
establishes	3
used	3
been	3

Table 2: Questions to test the chatbot

What is the thirteenth amendment?
ROBO: the thirteenth amendment abolished slavery.
What was the Dred Scott decision?
ROBO: dred scott v. sanford was the supreme court decision that said slaves were property and not citizens and that congress had no right to ban slavery in the territories.
Who was King George?
ROBO: "king george iii was the king of england who disbanded the colonial legislatures, taxed the colonies, and refused the olive branch petition leading to the final break with the colonies."
Who was Sam Adams?
ROBO: sam adams was a member of the sons of liberty who started the committee of correspondence to stir public support for american independence.
Who wrote the Declaration of Independence?
ROBO: "independence hall in philadelphia, pennsylvania is the site where the declaration of independence and the constitution were written."

Table 3: Cosine similarity in study guide

```
my_doc1 =  "Jamestown, the first permanent English settlement, was founded in 1607"
my_doc2 =  "The First Amendment states that ?Congress shall make no law? restricting
freedom of speech, religion, press, assembly, and petition."

documents = [my_doc1, my_doc2]
# Compute Cosine Similarity
from sklearn.metrics.pairwise import cosine_similarity
print(cosine_similarity(df, df))
[[1.          0.2556]
 [0.2556      1.      ]]
```

Table 4: Sparse matrix of chatbot questions

	adams	amendment	decision	declaration	dred	george	independence	is	king	of	sam	scott	the	thirteenth	was	what	who	wrote
What is the thirteenth amendment	0	1	0	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0
What was the Dred Scott decision?	0	0	1	0	1	0	0	0	0	0	0	1	1	0	1	1	0	0
Who was King George?	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	1	0
Who was Sam Adams?	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0
Who wrote the Declaration of Independence?	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	1	1

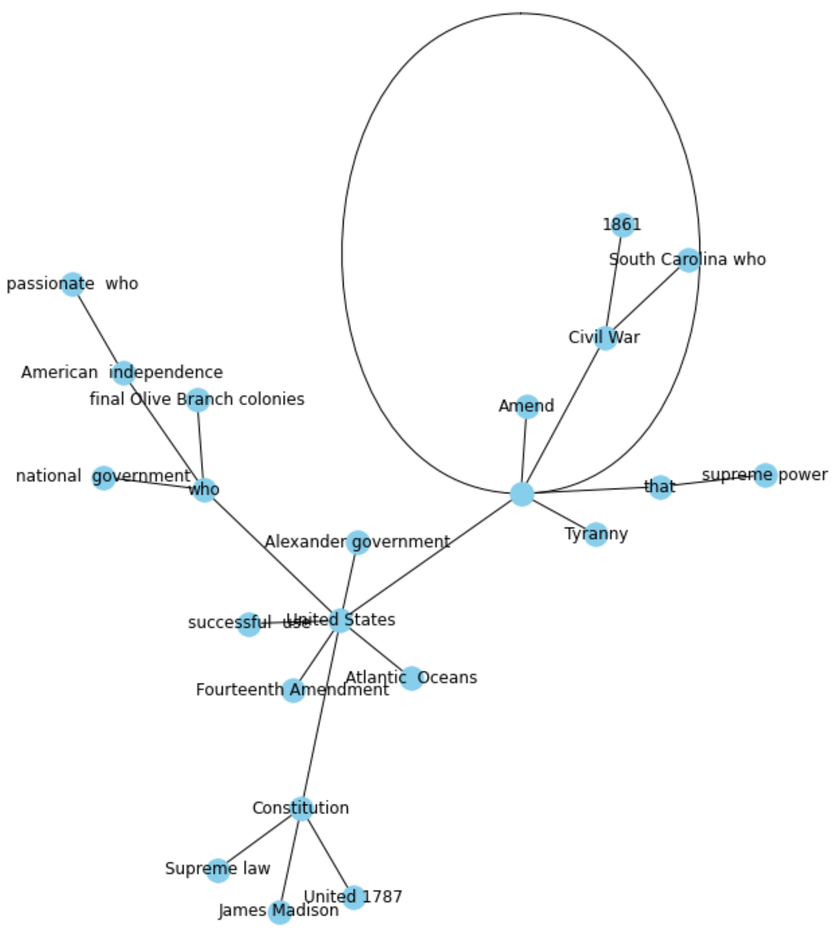
Table 5: Cosin similarity of chatbot questions

[[1.	0.3651	0	0	0.1825]
[0.3651	1	0.2041	0.2041	0.1666]
[0.	0.2041	1	0.5	0.2041]
[0.	0.2041	0.5	1	0.2041]
[0.1825	0.1666	0.2041	0.2041	1.]]

Table 6: Soft cosine similarity

	0	1	2	3	4	5	6
0	1.00	0.73	0.39	0.34	0.59	0.55	0.33
1	0.73	1.00	0.57	0.54	0.63	0.60	0.51
2	0.39	0.57	1.00	0.72	0.47	0.52	0.74
3	0.34	0.54	0.72	1.00	0.40	0.46	0.70
4	0.59	0.63	0.47	0.40	1.00	0.72	0.37
5	0.55	0.60	0.52	0.46	0.72	1.00	0.43
6	0.33	0.51	0.74	0.70	0.37	0.43	1.00

Table 7: American Government study guide ontology (zoomed in on center)



References

Lane, H., Howard, C., & Hapke, H. (2019). *Natural language processing in action*. Retrieved from https://learning.oreilly.com/library/view/-/9781617294631/?ar