A Reverse Mad Libs Game

Natural Language Processing

Scott Bing

BSSD 6000 System Software Design

Jonathan Lee

September 5, 2020

Abstract

Typically a game of Mad Libs involves the system asking the user to fill in a list of arbitrary words that represent a different part of speech, nouns, verbs adjectives, adverbs, prepositions, etc. The system substitutes these words using the appropriate parts of speech designation into a story. The story is then shown to the user. Everyone has a laugh when the completed story is read out loud.

What if you gave the game to a computer and asked it to not only substitute the correct parts of speech for each missing word but also to come up with a suitable word that matched the context of the story. So when the final story was read, fro the most part it would make sense. This represents the theme of this project. Given an arbitrary story with a series of missing words, the Reverse Mad Libs application uses the principles of Natural Language Processing to fill in each missing word with the ‘best’ candidate word.

Introduction

Do you commonly use that autocomplete or grammar checking facilities on your favorite word processor? Have you ever used a dynamic language translator? Have you ever used a chatbot on a web page? This is a picture of a human, typically clipart, that pops up and asks you if you would like to chat about your issue. I don't need to ask whether you have ever pleaded desperately "representative" over the phone when a machine has answered. I believe just about the entire population has dealt with this situation. If you have had any of these experiences, you have been exposed to some form of Natural Language Processing.

Natural Language Processing (NLP) is a branch of Artificial Intelligence that allows electronic devices to interpret human languages. It combines the fields of Linguistics, Computer Science, and Mathematics. NLP allows these devices to communicate with people, using a human language. NLP can read and interpret text and hear and interpret human speech. The discipline has been around for nearly 50 years. We are surrounded by language. We use it to speak, read talk, and even think. We see words about us every day in signs, Signs, email, text, books, web pages, etc.

What is Natural Language Processing

Natural Langage Processing (NLP) is an attempt to meld human language with computer language. It surrounds us today. From this point on in this discussion for the sake of brevity, NLP will refer to Natural Language Processing. When you use the auto-completed facility on a phone or computer application, you are using NLP. When you converse with a non-human voice on the phone, you are using NLP. It is not a new concept. It was thought about long before the inception of the modern-day computer.

Since the early days of the electronic computer, computer scientists have tossed around the idea of a computer understanding and communicating with the human being using their local language. Computers spike in terms of '0's and '1's. A leap had to be made to translate the complexity of the human language to binary ones and zeroes and the reverse. Alan Turing wrote a paper describing such a computer model in the 1950s. Knowledge-based machine learning applications came into existence during the 1980s and 1990s. A Knowledge-based system consulted a database full of phases and contexts to simulate human language. Modern-day NLP systems use pattern matching, neural networks and mathematics to learn and utilize human language.

Rules-based vs. Mathematical Based Learning

Natuaral Language Processing uses the principles of machine learning to synthesis human language. There are two design pardigms for machine language. There is a Rules-based and Mathematical based machine language systems.The early forms of NLP were primarily rules-based. The computer would consult a 'Knowledge Base' which was a database of grammar. As the machine learned, the database had to be continually updated with new rules. While there were some significant advances in NLP achieved by rules-base machines, they were not that efficient and they did not mimic the way that humans process language. There were some rules-bases systems that were capable of rules self-regeneration.

Infants learn language by recognizing and imitating observed patterns. What if machines could do the same thing. This is where mathematical based learning had its beginnings. With a mathematical base learning, machines are trained to recognize patterns. And just as humans learn and think, patterns are evaluated and either kept for future reference or eliminated. This continual process of evaluation and dismissal is referred to a machine training. It is base upon statistical analysis. The latest NLP systems use neural network models to facilitate language learning. The ideal NLP system would combine all of these systems. As the machine acquires knowledge, that knowledge could be stored in a database for quick retrieval later, rather than repeating the learning process. The knowledge base would no longer need to be externally updated

What is Reverse Madlibs?

Typically a mad libs game is played as follows. Player One is asked to supply a series of words broken down as parts of speech. The Player One chooses arbitrary words. Player Two substitutes these into a random story. Player Two reads the resulting story. The idea is that the story and the words have no connection with each other. In a reverse mad libs game, the machine sees the context of the story and uses NLP routines to substitute the best words into the story. My project uses BERT as the NLP mechanism to substitute the words. BERT will be discussed in the next section, however Bert is an NLP tool kit that provides functions that analyze language structures and simulate human conversation.

The application is fed a series of short stories where random words are missing. The missing words are indicated with a '-.' Symbol. BERT is used to make predictions for the missing words. Pressing the Generate button. See Figure 1.1

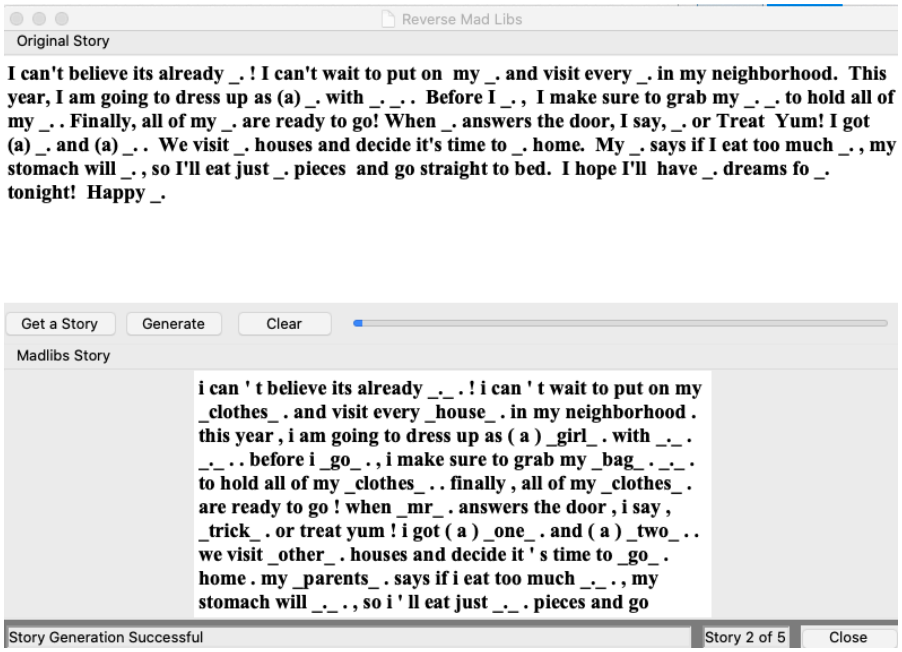

Figure 1.1 Reverse Madlibs Application

To operate the application, click the “Select a Story” button. The next story will be displayed in the upper frame. The number of the current story appears in the status bar at the bottom of the screen. Press the “Generate” button to process the story using BERT’s predictive algorithms. When the generation process is complete, the generated story will appear in the bottom frame and a confirmation message appears in the status bar at the bottom of the screen. The entire screen can be cleared off of its contents using the “Clear” button.

Project Resources

The application is written using the Python language. Python was chosen for its syntactical brevity, its eas of use, and its machine learning affinity. Python does not carry a built-in graphics capability. There is a built-in package called Tkinter. It creates simple graphic user interface screens and screen widgets, but the graphics are very limited. The Reverse Madlibs application was written using Tkinter. There some external Python packages that provide graphical routines. Since this application primarily deals with text, Tkinter is sufficient. Here is a list of some of the additional resources used to complete this project

- Python – the primary implementation programming language. Python is terse, easy to use, and lends itself to machine learning applications.
- Pytorch – a Python package that is used for data analysis functions.
- BERT – a Python package that implements NLP features
- Tkinter – a Python Package that provides a GUI wrapper for Python
  BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Pytorch holds the BERT functions and routines. Reverse Mad Libs makes use of the BERT NLP toolkit. BERT provides the NLP engine. More details on BERT will be provided later on in this discussion

Improvements to the Application

Any application is subject to improvements. Here are a few improvements to the application that would make it more robust. Rather than storing the stories as hard-coded text in the program, a file processing menu could be added to allow the user to read the stories stored in an external location. The user could simply select any story stored on a file system. This would provide more flexibility and allow for a theoretically unlimited number of stories, The application could also offer an "Edit" facility, so the user could alter the position of the substitute words. Provide a routine predicated by a 'Prepare' button that would automatically edit the file for the user by randomly placing the missing word indicators into the file. The user would simply select a text story file from the local files system and the application would automatically prepare it for predictive processing. The substitution positions in the text are difficult to read. The substitution word could be written with bold typeface, surrounded by additional spaces, or utilize different font colors to distinguish them from the remainder of the text. Some of the resulting substitutions were still erroneous. Utilizing a different trained dataset might rectify these poor substitutions. Allowing the application to train itself to the process might also correct the inadequate substitutions.

BERT

BERT (Bidirectional Encoder Representations from Transformers) is an NLP package supplied by a Google. What makes BERT so special? Why was it chosen for this project? Traditional NLP processors process text in a lateral direction as most human speakers. This is left to right or right to left. BERT combines the lateral directions to achieve a bidirectional scanning procedure. This accounts of r contextual meaning from both directions. Google engineers felt that would give a deeper analysis of the language. It may find aspects of the language not uncovered by a simple unidirectional scan. BERT does not rely on any directional scanning, it consumes the entire section of text in one access. The scanning process does not have a directional bias it evaluates the context of words based on their complete contextual surroundings. This makes BERT is ideal for classification problems, questions, and answering applications, and parts of speech recognition.

BERT is an open-source package provided by Google. The source is located at https://github.com/google-research/bert which also includes a thorough descriptive README file. BERT s also unique in the fact that it not only processes word tokens, it also processes complete sentences as a whole. First, It converts text to lowercase, if BERT's lowercase model is active. Next, it tokenizes the text into words. Words with prefixes and suffixes are broken down further into separate tokens. For example, the word calling would become 'cal; '##ing' and the word preview would become 'pre##', 'view'. The resulting words are indexed according to an internal BERT vocab file. Special BERT tags are inserted at the beginning of the text and the end of each sentence.

1. Tokenize it (i.e. "sally says hi" -> ["sally", "says", "hi"]). Break it into words

2. Tokens are similar to words, but they are not always words. Some tokens are common phrases used in the language. The tokens are kernels of information. The tokens are arranged in the order of importance within the context of the immediate language expression. The tokens are converted to vectors. They are then integrated into neural networks.

3. Break words into WordPieces (i.e. "calling" -> ["call", "##ing"]). Break at suffixes and prefixxes
4. Map our words to indexes using a vocab file that BERT provides
5. Add special "CLS" and "SEP" tokens (see the readme) CLS is a beginning of the firss sentence;.SEP is an end of sentence marker
6. Append "index" and "segment" tokens to each input (see the BERT paper)

How Bert Works

https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

https://www.lyrn.ai/2018/11/07/explained-bert-state-of-the-art-language-model-for-nlp/

BERT – State of the Art Language Model for NLP

BERT uses a Google-developed facility called a transformer. to learn the context between words. The transformer is composed of encoders and decoders. The encoder reads the txt and breaks it down into tokens. The decoder makes predictions about the contextual relationship of those words. The transformer is not limited to the reading test left to right or right to left. It consumes an entire block of text. It considers the textual context from all directions.

Here are the steps by which BERT processes language:

7. Convert text to lowercase (if we're using a BERT lowercase model)
8. Tokenize it (i.e. "sally says hi" -> ["sally", "says", "hi"]). Break it into words
9. Tokens are similar to words, but they are not always words. Some tokens are common phrases used in the language. The tokens are kernels of information. The tokens are arranged in the order of importance within the context of the immediate language expression. The tokens are converted to vectors. They are then integrated into neural networks.

10. Break words into WordPieces (i.e. "calling" -> ["call", "##ing"]). Break at suffixes and prefixxes
11. Map our words to indexes using a vocab file that BERT provides
12. Add special "CLS" and "SEP" tokens (see the readme) CLS is a beginning of the firss sentence;.SEP is an end of sentence marker
13. Append "index" and "segment" tokens to each input (see the BERT paper)

BERT Takeaways

Unlike other NLP tool kits that ar limited to unidirectional analysis, BERT is bidenrctional in nature, BERT is easy to use and it is an open-source product.

Advantages and Disadvantaes of BERT

Bert is no panacea for NLP. It does have it's drawbacks. It is very processing intense compared with the directional algorithm; however, this deficiency is offset with increased accurate contextual predictions.

Project Resources

These are the resources used to complete this project

Python – the primary implementation programming language. Python is terse, easy to use, and lends itself to machine learning applications.

Pytorch – a Python package that is used for data analysis functions.

BERT – a Python package that implements NLP features

Tkinter – a Python Package that provides a GUI wrapper for Python

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Why Bert

There are numerous free NLP tool kits available for anyone to use. Many of them provide ample text parsers, text analysis utilities, and text predictors. These are the samae features that I demonstrate with my project. BERT provides a unique bidirectional scanning feature tha the other packages do not provide.

BERT

BERT (Bidirectional Encoder Representations from Transformers) is an NLP package supplied by a Google. Traditional NLP processes processed text in a left to the right direction as most human speakers. BERTuses both left to right and a right to left scanning procedure. Google engineers felt that would give a deeper analysis of the language. It may find aspects of the language not uncovered by a simple left to right scan. BERT does not rely on any directional scanning, it consumes the entire section of text in one access. The scanning process does not have a directional bias it evaluates the context of words based on their complete contextual surroundings.

This is accomplished by using a Google developed facility called a transformer

Do you commonly use that autocomplete facility on your favorite word processor? Have you ever used a chatbot on a web page? This is a picture of a human, typically clipart, that pops up and asks you if you would like to chat about your issue. I don't need to ask whether you have ever pleaded desperately "representative" over the phone when a machine has answered. I believe just about the entire population has dealt with this situation. If you have had any of these experiences, you have been exposed to some form of Natural Language Processing.

BERT Explained: State of the art language model for NLP
https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

A bibliography includes items you may have read or looked at, but did not quote in your paper. (select this text box to delete it)

bold) (start on a new page)

BOOK WITH ONE AUTHOR
McGhee, Robert. *The Last Imaginary Place: A Human History of the Arctic World*. Oxford: Oxford University Press, 2005.

A second work by the same author: use em dash: insert -> symbol -> more symbols -> the 3rd dash is em dash. (select this text box to delete it)

——. *Beluga hunters: an archaeological reconstruction of the history and culture of the Mackenzie Delta Kittegaryumiut*. [St. John's]: Institute of Social and Economic Research, Memorial University of Newfoundland, 1974.

BOOK WITH TWO OR MORE AUTHORS
Williams, Philip F. and Yenna Wu. *The Great Wall of Confinement: The Chinese Prison Camp Through Contemporary Fiction and Reportage.* Berkeley: University of California Press, 2004.

WORK WITH AUTHOR'S NAME IN THE TITLE
Darwin, Charles. *Charles Darwin's Letters: A Selection, 1825-1859.* Edited by RFrederick Burkhardt. Cambridge: Cambridge University Press, 1996.

"works cited" is a list of the works you actually made reference to in your footnotes in the body or any images. (select this text box to delete it)

Works Cited (this is bold) (start on a new page)

EDITED WORK WITHOUT AN AUTHOR
Salih, M. Mohamed Salih, ed. *African Parliament: Between Governments and Governance.* New York: Palgrave Macmillan, 2005.

TRANSLATED WORK
Bingying, Xie. *A Woman Soldier's Own Story*. Translated by Barry Brissman and Lily Chia Brissman. New York: Columbia University Press, 2001.

MULTIVOLUME WORK
Kinder, Hermann and Werner Hilgemann. *The Penguin Atlas of World History.* Vol. 1, *From Prehistory to the Eve of the French Revolution.* Rev. ed. New York: Penguin Books, 2004.

CHAPTER IN AN EDITED WORK(ANTHOLOGY)

Hamilton, Bernard. "The Impact of the Crusades of Western Geographical Knowledge." In *Eastward Bound: Travel and Travellers, 1050-1550,* edited by Rosamund Allen. Manchester: Manchester University Press, 2004.

EDITION OTHER THAN THE FIRST

Chafe, William H. *The Unfinished Journey: America since World War II*, 5th ed. New York: Oxford University Press, 2002.

WORK IN A SERIES

Flehinger, Brett. *The 1912 Election and the Power of Progressivism: A Brief History with Documents*. Bedford Series in History and Culture. Boston: Bedford/St. Martin's, 2003.

ARTICLE IN A JOURNAL PAGINATED BY VOLUME

Lucero, Lisa. "The Collapse of the Classic Maya: A Case for the Role of Water Control." *American Anthropologist* 104 (2002): 814-6.

ARTICLE IN A JOURNAL PAGINATED BY ISSUE

Wynn, Rhoda. "Saints and Sinners: Women and the Practice of Medicine throughout the Ages." *Journal of the American Medical Association* 283, no. 5 (2000): 668.

ARTICLE IN A POPULAR MAGAZINE

Thomas, Evan. "The Day That Changed America." *Newsweek Special Double Issue*, December 2001-January 2002, 45-46.

NEWSPAPER ARTICLE

Harris, Hamil. R. and Darryl Fears. "Thousands Pay Respects to King." *Washington Post*, February 5, 2006, sec. A, Maryland edition.

BOOK REVIEW

Cooper, Ilene. Review of *Nat Turner's Slave Rebellion in American History*, by Judith Edwards. *Booklist* 96 (2000): 1093.

SOUND RECORDING

Holst, Gustav. *The Planets*. Royal Philharmonic Orchestra. Andre Previn. Telarc compact disc 80133.

FILM, VIDEOCASSETTE, OR DVD

The Civil War. Produced and directed by Ken Burns. 11 hours. PBS Video, 1990. 9 videocassettes.

REFERENCE WORKS

Well-known reference works, such as encyclopedias, are generally included in footnotes/endnotes but not in the bibliography. Check with your instructor to see if he/she would like you to include them in your bibliography, in which case you would follow one of the examples for a book.

(select this text box to delete it)

WHOLE WEBSITE WITH A KNOWN AUTHOR
Knox, E. L. Skip. “The Crusades.” http://crusades.boisestate.edu.

WHOLE WEBSITE WITHOUT A KNOWN AUTHOR
The Ohio State Department of History. “The Scopes Trial.” http://history.osu.edu/Projects/Clash/Scopes/scopes-page1.htm.

SELECTION FROM A WEBSITE
Linder, Douglas. “An Account of Events in Salem.” *Famous Trials*. www.law.umkc.edu/faculty/projects/ftrials/salem/sal_acct.htm.

ONLINE BOOK
Mather, Cotton. *Memorable Providences, Relating to Witchcrafts and Possessions*. Boston: 1689. At Douglas Linder. *Famous Trials*. www.law.umkc.edu/faculty/projects/ftrials/salem/asa_math.htm.

ARTICLE IN AN ONLINE JOURNAL
Friedman, Shamma. “A Good Story Deserves Retelling--The Unfolding of the Akiva Legend.” *Jewish Studies: An Internet Journal* 3 (2004):55-93. www.biu.ac.il/JS/JSIJ/3-2004/Friedman.pdf.

ARTICLE ACCESSED THROUGH AN ELECTRONIC DATABASE
Toplin, Robert Brent. “The Filmmaker as Historian.” *American Historical Review* 93 (1988): 1210-27. *JSTOR*.www.jstor.org.

ONLINE NEWSPAPER ARTICLE
Linzer, Dafna. “Strong Leads and Dead Ends in Nuclear Case Against Iran.” *WashingtonPost.com*. February 8, 2006. www.washingtonpost.com/wp-dyn/content/article/2006/02/07/AR206020702126.html (accessed February 9, 2006).