Robert Hand
Rhand7
Forward Data Lab Week 1

Wikipedia Text Generation using Large Language Models and NLP

The confusing contradiction of Wikipedia's (and similar open source, publicly moderated online encyclopedias) public perception as an unreliable source and widespread use in scholarly and more casual research persists despite recent advances in data-to-text generation. Violations in Wikipedia's Neutral Point of View policy include language style, editor profiles/interactions, sources cited, gatekeeping, and selective coverage of topics. Linguists can decode this bias by splitting articles into atomic parts, analyzing articles phrase by phrase, creating article editors' profiles, and scoring news sources' political affiliations, as identified in a 2017 case study from the University of Hannover (Hube, C., 2017).

Using NLP to nullify biases in article summary is not a novel concept, as has been done successfully numerous times before the use of Large Language Models. In 2018, researchers from Google Brain developed one of the first language models to have success summarizing from more than one document (Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, Ł., & Shazeer, N., 2018). This model uses a decoder-only transformer, which improves past language models by focusing on only the most important parts of the text sequence during tokenization. Rather than using more computing power to process longer text strings, this model seeks to optimize language processing by reducing key-value pairs during tokenization, allowing for greater efficiency in processing larger text sections.

Similarly, the WikiTablet project (Chen, M., Wiseman, S., & Gimpel, K., 2021) sought to rewrite bias-free Wikipedia articles by pairing subsections of text with a tuple containing specific metadata about title, section, and article structure. WikiTablet first constructs this data table from page info boxes (captions), article structure, hyperlinks, and other named entities in the article using named entity recognition before using this data table to construct an article. This model uses beam search, an algorithm designed to expand the most promising nodes in a section, to emphasize important takeaways in summarization and reduce redundancy and unimportant information. Similarly, WikiTablet uses a technique called N-gram blocking, which restricts a language model from generating a repeated phrase (N-gram) during generation to eliminate unnatural language repetitions. During beam search, N-gram parsing reduces a node's chance to be visited to 0 if it is a repeated phrase. During testing, engineers behind WikiTablet found that it was virtually indistinguishable from human-written text however it struggled with hallucination and factual accuracy.

To assemble our solution, we need three components: our decoder-only language model, a component to find and parse relevant articles, and a pipeline to display the LM output in an organized way. Ideally, our language model should be "plug and play," where we can easily switch and test different language models. For the second component, we need a system to determine if an article is relevant to the topic at hand. A naïve implementation would simply recompile the same articles the original Wikipedia article uses. However, this would risk leading to the same biases as the original. A more complex model could combine the original phrase

with the language model to generate relevant subheaders and then parse a research database for the top N relevant sources. This leads to the third component, article organization, as the most straightforward way to determine subdivisions of the article would be to let the language model organize the information during generation.

The initial design uses Google's Gemini API as the language model. Using the API rather than running the LLM locally removes the requirement to have significant disk space and computational power, allowing users to receive a similar experience regardless of their device. Finding reputable, scholarly sources from which to generate the articles would be the first initial challenge, as Google Scholar disallows web scraping and frequently blocks IP addresses in suspicion of automated behavior. While first trying to circumvent these checks using a proxy address for each visit, it was much more effective in circumventing Google Scholar entirely. An API named SerpAPI was used to find sources, an open-source framework for scraping the links returned from a Google query. Because most academic papers are uploaded as pdf files, scraping for "{term} filetype:pdf", reading in each file as an in-memory binary stream, and extracting the final text using the PyPDF library, would put our reference material in a form the language model can understand. Finally, by combining our sources and prompting engineering Gemini to synthesize our sources in the provided format without using any prerequisite knowledge of its own, we can output our "Wikipedia" -like document. The source code can be found at https://github.com/RobHand27/Wiki-Generator .

Based on the initial output, some improvements can be made. For example, the current implementation only uses a "one size fits all" generation prompt. A more sophisticated model could interpret the important information involving a subject and customize the article subsections as such. Likewise, splitting the subsections into separate API calls will result in more accurate, detailed results, as LLM's tend to work more accurately when prompted to generate smaller sections of text. Further, scraping PDF files from Google Search runs the risk of citing biased or inaccurate sources. The current implementation can synthesize an average of 8-10 sources, depending on the subject matter. However, a more advanced filtration method would make this more accurate.

References

[Hube2017] Hube, C. (2017). Bias in Wikipedia. In Proceedings of the 26th International Conference Companion on World Wide Web (pp. 717-721). International World Wide Web Conferences Steering Committee.

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, Ł., & Shazeer, N. (2018). Generating Wikipedia by Summarizing Long Sequences. In International Conference on Learning Representations (ICLR 2018).

Chen, M., Wiseman, S., & Gimpel, K. (2021). WIKITABLET: A Large-Scale Data-to-Text Dataset for Generating Wikipedia Article Sections. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL, 2021). Association for Computational Linguistics.