Hacker News threads summarization

Project Report - Introduction to NLP 2020 Spring

Teemu Koivisto 014211393
Github

Project Description

In my project I pursued to create a simplistic text summarization of Hacker News https://news.ycombinator.com/ threads and their comments, which is a popular website for sharing technological and in general interesting articles. For each thread there are a number of comments, at maximum I think there can be about 300 per page. Reading such long comment chains is quite tedious so I thought maybe you could try to summarize it somehow.

To do this I create a text summarization algorithm, utilizing a very basic frequency scoring. First it fetches and parses the comments for a given thread, then removes all the stopwords and non words, turns the words into their basic form with Porter Stemmer, creates a frequency table for each word and then ranks all the sentences by the sum of their frequencies divided by the total word count of a sentence. For the end result the top 20 sentences are joined together.

Results

The results were not very good, and the summaries were quite uninformative gibberish. Using this article it produces a summary: https://news.ycombinator.com/item?id=22552632

```
"You can't. I'm guessing that isn't you. You can't sell what you don't
have. No, it wasn't secrecy that killed them. Nobody wanted it. None. I
have both a Magic Leap One and a HoloLens 1. Their problem hasn't been
timing. i don't know what your company is. That you don't see it hyped
on TV doesn't mean it is not there. I don't want a voice telling me
what to do. The goal isn't to own a product. Don't write off VR just
yet. It uses magnetic tracking and it just doesn't do well. Oh man, I
didn't know he worked there... don't you need to be able to see what's
actually going on and not be in VR land? I've got news for Magic Leap.
```

```
General Magic, GetMagic.com, and now this (not that we didn't see it
coming)... AR has no use case. > AR has no use case."
```

Just by assessing the readability of the summary it is quite obvious, that there are major problems with how the algorithm works. It over-values short sentences with popular basic words such as “guess” or “sell”. Also it doesn’t consider the context of the sentences at all, but they are picked at random from multiple different comments thus the summary lacks cohesion and structure. And while it yes counts the frequencies of the same words, it doesn’t have any inclination of the similarities of words that mean almost the same thing: good, great, awesome etc.

Improvements

So while the system worked as a simplistic first approach, one could go around many ways to enhance it. For example adding word embeddings would probably make the frequencies better, but then one should think are they even actually usable for making good summaries? Especially when picking random sentences from a large amount of comments, it’s not very satisfactory to just pick random sentences no matter how good your algorithm is.

Therefore a better approach would be an algorithm utilizing Natural Language Generation (NLG), that would create completely new text out of the corpus. This would be immensely more difficult yet I believe it would make a lot more sense when using such vast set of heterogeneous comments. Also one could use the order of the comments as a feature, as they are ranked by their popularity in some hidden manner (votes are not shown). Or the popularity of the users who have written them.

Conclusion

Text summarization is a difficult task and while seemingly simple, to produce reasonable results one must consider many aspects of ranking the content and the generation of the summary text itself. Simple frequency ranking might work with very basic data, but with any even a little more complicated data, it falls apart and the use of NLG for the summary generation should be heavily considered.

Side note

My initial project was to create a syllabication model or algorithm, and I got somewhat far with finding a dataset with 180,000 syllabized words and then two baseline algorithms, Liang's hyphenation algorithm and nltk's own syllabication algorithm. Both had quite poor accuracy with my dataset, but it was also partly because the data wasn't properly cleaned.

Nevertheless the whole algorithm creation proved to be quite difficult, although I found a research article that overlaid a solution that had promising results using N-grams with TF-IDF scores. But I didn't get to the part of implementing that one. Also I did consider first transforming the words into their phonetic IPA form and then applying syllabication, because from what I learned from Dr Nimer Abusalim and his amazing videos, that is how linguistics actually do it.

Yet to get something actually working done, I instead went with this text summarization.