Timothy Pratt

Max Bender

CS231

28 November 2022

Project8: Word Trends

Abstract:

This project once again involved using and analyzing the Reddit comments data from the previous two projects. However, this project focused on a new data structure, Heaps. We used a Heap to build a PriorityQueue, which was able to quickly return the largest item stored in the Heap. This was used to find the most common words for a given file. We also revisited the previous project's data structure, the HashSet, and used it to find the frequency of specific words. Both data structures were useful in allowing us to analyze the most common words used, along with how words trended over several years.

Results:

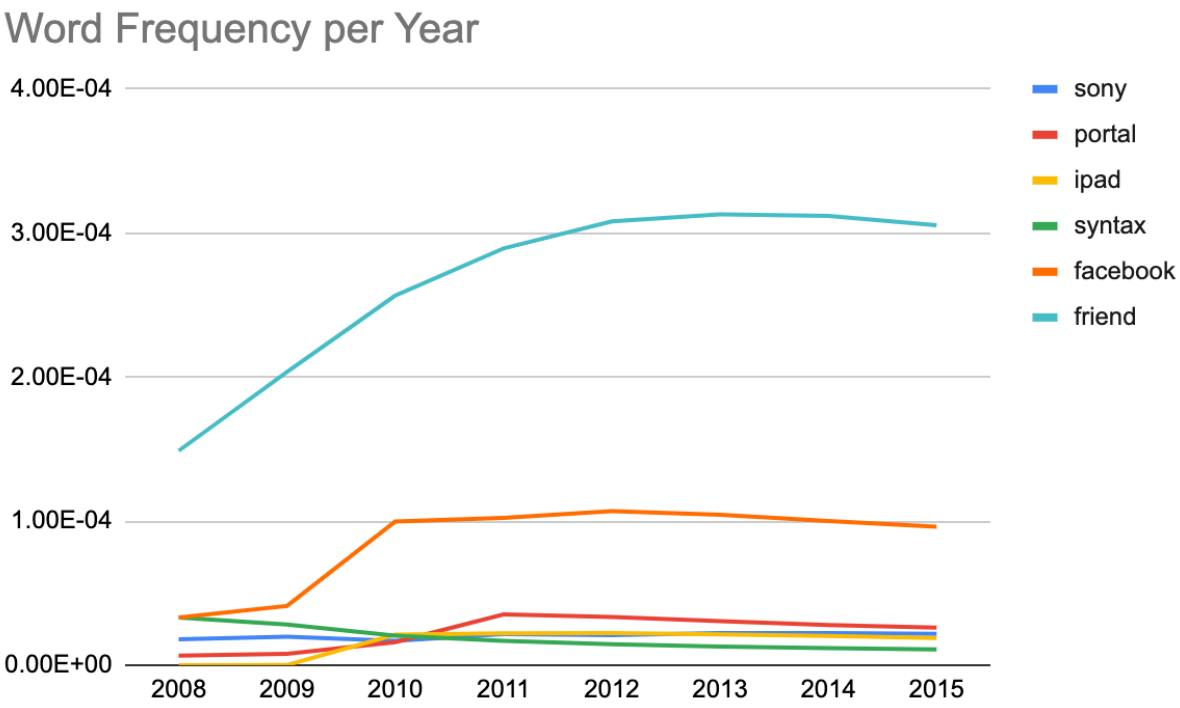
Graph 1: Most Common Words

	2008	2009	2010	2011	2012	2013	2014	2015
1	the	the	the	the	the	the	the	the
2	to	to	to	to	to	to	to	to
3	a	a	a	a	a	a	a	a
4	of	and	i	i	i	i	i	i
5	and	of	and	and	and	and	and	and
6	i	i	of	of	of	of	of	of
7	that	that	that	that	you	you	you	you

8	is	is	is	you	that	that	that	that
9	in	you	you	is	is	is	it	it
10	you	it	it	it	it	it	is	is

This graph was made from taking the ten most common words per file. The results here aren't very surprising. I expected articles and pronouns to be towards the top since regardless of what people are talking about, words like "the" and "I" will be in almost every sentence. There is some slight variation throughout the eight years, but nothing too drastic, which is to be expected.

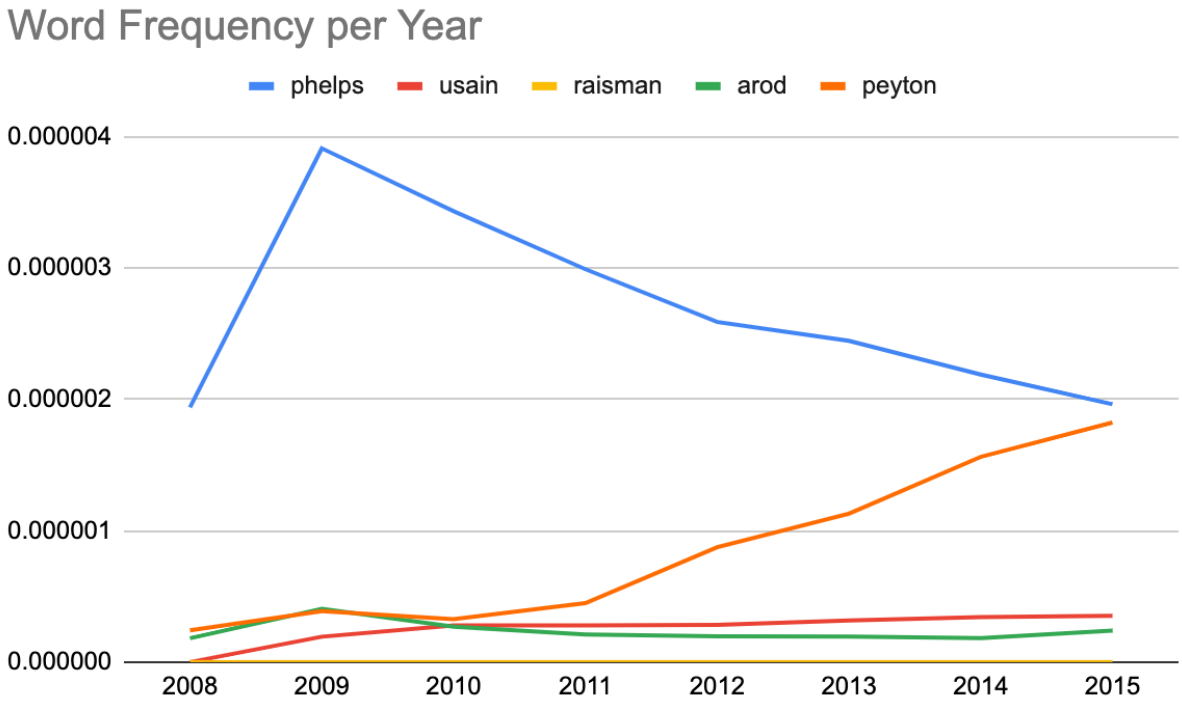
Graph 2: Trends 1



The most noticeable thing about this graph is that the word "friend" is used far more often than all of the other words on the list. While most of these words strictly relate to technology, "friend" is a much more general term that is used everywhere, not just in relation to

online friends. After that, the word “Facebook” is the next most popular word, likely because of its massive popularity and impact. Facebook was one of the first, and one of the most popular social media sites throughout most of the 2010s. Its effect on the daily lives of people is massive, so it makes sense that people are talking about it a lot. Beyond this, the word “iPad” jumps in usage around 2010, the year of its release, and “portal” spikes in 2011, the year that the video game Portal 2 was released.

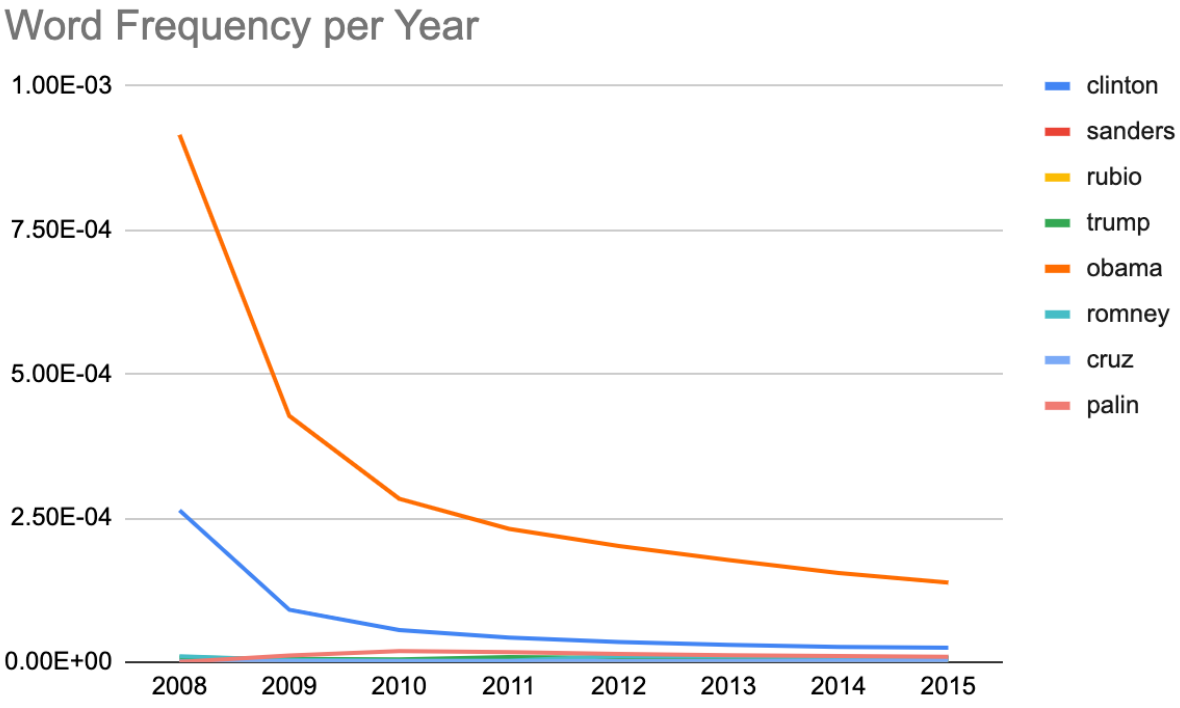
Graph 3: Trends 2



For this graph, the results were not quite as I expected. I thought that “Phelps” “Usain” and “Raisman” would have spiked during years with the Olympics since they are all Olympic athletes. However, this is not the case. Most remain relatively stable, except “Phelps” which is consistently high. “Phelps” peaks in 2009, which is the same year in which he is caught having

possession of marijuana. Beyond that, Phelps is also one of the most popular Olympic athletes and swimmers. People like to make comparisons to him frequently, and this may contribute to his high frequency. The word "Peyton" also increases in frequency up through 2015. This can most likely be attributed to quarterback Peyton Manning. The reason he was discussed so frequently during these years is likely because he was traded to the Denver Broncos in 2012 (frequency starts increasing) and retires in 2015 (frequency peaks).

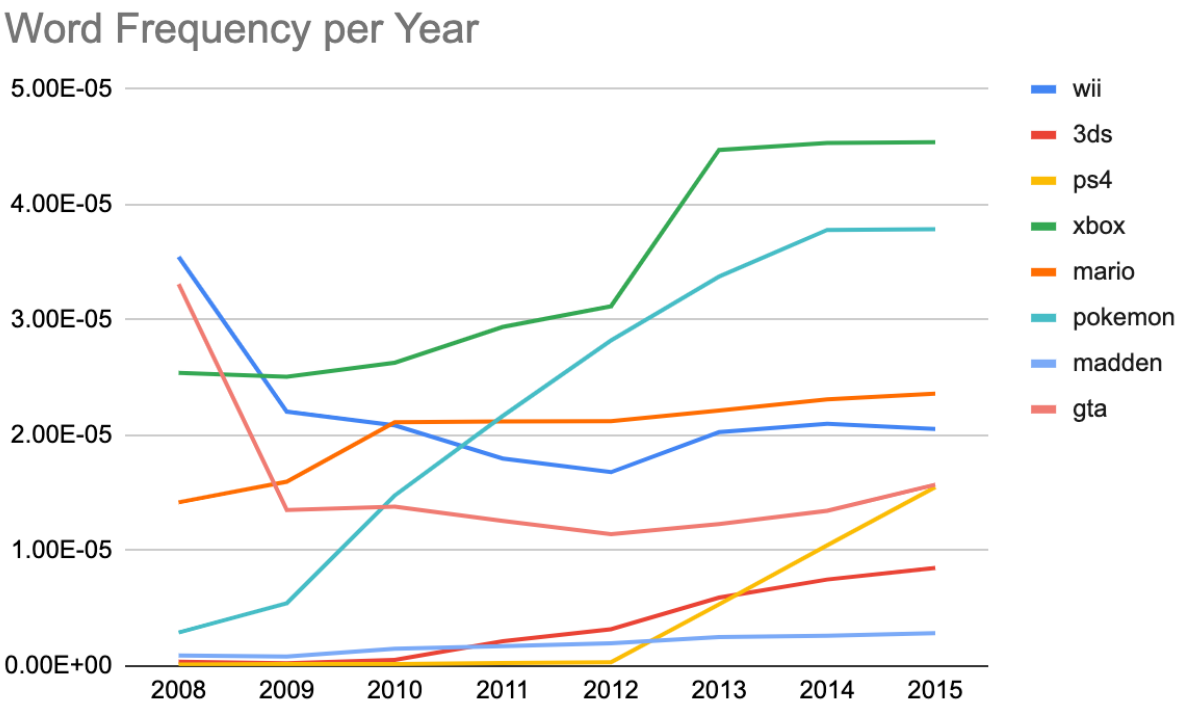
Graph 4: Trends 3



Similarly, I expected this graph to look different than it actually does. I expected many words to spike during election years. Instead, most words remain consistently low. The two exceptions are "Obama" and "Clinton", both of which peak during 2008. "Obama" has a significantly higher frequency than the rest, since he was president from 2008 to 2016. (This also

explains the word's peak in 2008). "Clinton" most likely referring to Hillary Clinton, also peaks in 2008, the first year she ran for president. While I understand these trends, I still would have expected there to be a spike in 2012, specifically for Obama and Rodney, the two main presidential candidates that year.

Graph 5: Trends 4



This final graph was made from words I chose. I looked for various video games and consoles since they are things people on the internet enjoy discussing. While many of these trends seem to be unrelated, the main consistency I found is that discussion seems to be high around "big release" times. For instance, the words "PS4" and "Xbox" shoot up in usage around the year 2013, which is when the PlayStation 4 and Xbox One were released. On the other hand, words like "Wii" and "GTA" steadily decline in frequency since their "big releases" were earlier

(The Wii in 2006 and GTA IV in 2008). “Pokemon” doesn’t spike but has a steady incline over several years, as Nintendo releases several well-received Pokemon games for the DS. Lastly, “Mario” “Madden” and “3DS” are all consistent since the former two have new games regularly, and the 3DS just wasn’t the most popular system.

Extensions:

For my extension I tried implementing the CommonWordsFinder class with an ArrayList instead of a Heap. I hoped that by doing this, I could decrease the overall runtime. Before doing this, I calculated the runtime of getting the ten most common words from each of the eight files using the regular Heap. I ended up getting a runtime of 2134.0 ms. On my first attempt, I added all KeyValuePairs to an ArrayList, then sorted that list in descending order. I then removed and returned the first ten items of the list. This gave me a much greater runtime of 5922.0 ms. However, I realized that the reason for this was that I was removing the first item of an ArrayList, which took much longer than removing the last item. So I rewrote the code to sort the list in ascending order, then return and remove the last item in the list. The runtime of this was now 2320.0 ms, much closer to the original. I believe that depending on the data size, this function could even faster. With Heaps, both adding and removing items takes O(log n) time. ArrayLists take O(1) time for both of these, but have an “expensixe” MergeSort function it has to run once. Perhaps if we wanted to get the 1000 most popular words, an ArrayList would be more efficient.

Reflections:

This project showed both the pros and cons of using Heaps, compared to BSTs and HashMaps. Heaps were best used for the CommonWordsFinder because they can quickly return

the largest (or smallest depending on the comparator) item in the heap. A HashSet or BST on the other hand would need to loop through every single item in order to find the largest item(s), making them inefficient in this regard. On the other hand, Heaps were not used in the WordTrendsFinder, because the HashSet was used instead. I chose a HashSet instead of a BST because HashSets are almost always better than BSTs, as seen in the previous project. In this situation, a HashSet is more useful than a Heap, since HashSets can quickly find specific items based on name. Heaps meanwhile are not as effective. In order to find a specific item, one must loop through the array the Heap is built around. Looping through an array takes longer than the near instantaneous lookup time of the HastSets. Ultimately Heaps are incredible at implementing priority queues, but are outclassed by HashSets in other areas.

Acknowledgments:

Professor Bender's Lecture Code: Outline of MergeSort code

reddit.com: Source of comments and data