Study of Internet Metrics on Document Collaboration Applications

Kuhu Halder (kh761) and Katherine Lee (ksl103)

May 6, 2023

Abstract

Document collaboration has become an essential feature in word processors and similar applications. Users expect to be able to share their documents with friends and colleagues in order to be able to view, edit, and collaborate in real-time. While document collaboration is vital to these tools, especially in the post-pandemic era as people continue to work remotely, there is very little research on how well they perform. This study seeks to understand how connectivity and network affect the performance of some of the most popular collaboration tools: Google Docs, Microsoft Word, and Notion, and how performance can be improved by studying a small custom document collaboration app.

1 Introduction

Document collaboration tools have become essential in helping users to work collaboratively at work, in school, and even in personal settings. These tools allow users, even across the globe, to work on a single document in real-time, fostering collaboration and innovation. In the past few years, especially in the midst of a global pandemic, these tools have allowed people to stay connected and continue working seamlessly as they always would have. And yet, while these tools play a critical role in connecting users across the globe, there is little research on how effectively these tools perform and how individual tasks such as real-time editing, change propagation, etc. respond to disruption and other internet and connectivity issues. Here, we take a first step toward this understanding, by analyzing internet benchmarks of several popular document collaboration tools as well as our own custom document collaboration application.

2 Related Work

There is some existing work in the area of measuring internet benchmarks for real-time, production collaboration, and streaming applications.

Michel et al. took measurements and provided an in-depth analysis of the popular video-conferencing application Zoom. Through controlled experiments, they were able to identify specific metrics like latency, media bit rates, frame size, and jitter on their network. [1] In a similar vein, MacMillan et al. were able to study and compare the performance of several popular video-conferencing applications, Zoom, Google Meet, and Microsoft Teams. Similar to Michel et al., they were able to measure "application-layer performance metrics (e.g., resolution or frames per second) under different network capacities and scenarios." [2] MacMillan et al. primarily used popular API WebRTC (Web Real-Time Communication) to capture peer-to-peer communication on these applications. Zhang and Liu conducted a related study on the interactive live-streaming application, Twitch. In their research, Zhang and Liu measure three latencies of interest: live messaging latency, broadcast latency, and switching latency, and attempt to measure network impact for viewing users. [3]

There are also some existing programs and tools that measure some of the benchmarks that we plan to study. TwitchTest and Twitch Inspector are tools, exclusive to the live streaming application Twitch, that are able to measure internet connection, bandwidth, and broadcasting metrics. [4, 5] As previously mentioned, WebRTC is a free, open-sourced project that is able to provide real-time statistics for Peer-to-Peer (P2P) audio and video applications. [6] Google Lighthouse and Google Web Vitals are open-source automated tools that provide internet metrics

with the express mission to improve the quality of web pages. [7] They are able to run on any web page and provide audit information for performance, accessibility, etc. [8]

3 Method

What all of the previously mentioned related work has in common is an attempt to measure internet benchmarks on proprietary technology, without having exclusive access to back-end servers, internal systems, etc. There is no research that we could locate that attempts to do this same type of study on document collaboration systems. In order to do this same type of study with document collaboration, we plan to collect and analyze internet benchmark data for several document collaboration tools as well as our own lightweight custom document collaboration application.

3.1 Applications

The proprietary applications that we studied were Google Docs, Notion, and Microsoft Word. Google Docs and Microsoft Word are primarily word processors, and part of larger suites of document editors Google Drive and Microsoft Office Suite, respectively. Though these two applications are explicitly word processors, they have functionality that allows multiple users to collaborate on a document at once. Notion is a productivity and note-taking web application, rather than a word processor, and its functionality is more focused on task management, project tracking, to-do lists, etc. For the purposes of this study, each application will simply be treated as a collaborative document editor, where multiple users are able to access, view, and edit a document at the same time.

Our custom document collaboration application is built using React.js, and SocketIO, with a MongoDB backend. It is built with reference to a React.js tutorial for building collaborative whiteboards.[9, 10] The custom application has the same limited functionality of allowing several users to collaborate through typing on a single document that we are testing with Google Docs, Microsoft Word, and Notion.

3.2 Benchmark Gathering

In order to do benchmark gathering, we are using two primary APIs. The first is NPM Auto-Cannon which is an HTTP benchmarking tool written in Node.js on top of wrk2. AutoCannon, similar to WebRTC is able to provide some important statistics like latency, throughput, and data rate, but is not exclusive to audio and video P2P applications. [11] The second tool that we utilized was Chrome DevTools. DevTools is an extension built directly into the Google Chrome web browser. In particular, we are using the suite of network tools to provide network activity insights, record network requests, emulate offline and slow network connections, throttle web socket connections, etc. [12, 13]

We tested three primary benchmarks under a variety of plausible conditions that could affect internet metrics. The benchmarks that we observed are:

- Latency
- Request/Second (Throughput)
- Bytes/Second (Data Rate)

4 Experiments & Analysis

4.1 Scenarios

As previously mentioned, the goal of this study was to test each of these individual applications under a series of plausible conditions that might affect their metrics. The scenarios that we tested were:

- What happens when only a single user is actively working on the document and all other users are passively viewing

- What happens when multiple users are actively working on the document
- How does uploading images affect internet metrics
- What happens when the application must compete with other traffic (throttling)
- What happens when network connectivity is disrupted

Each of these scenarios was deemed to be plausible in a normal co-working environment.

4.1.1 Custom Document Collaboration Application

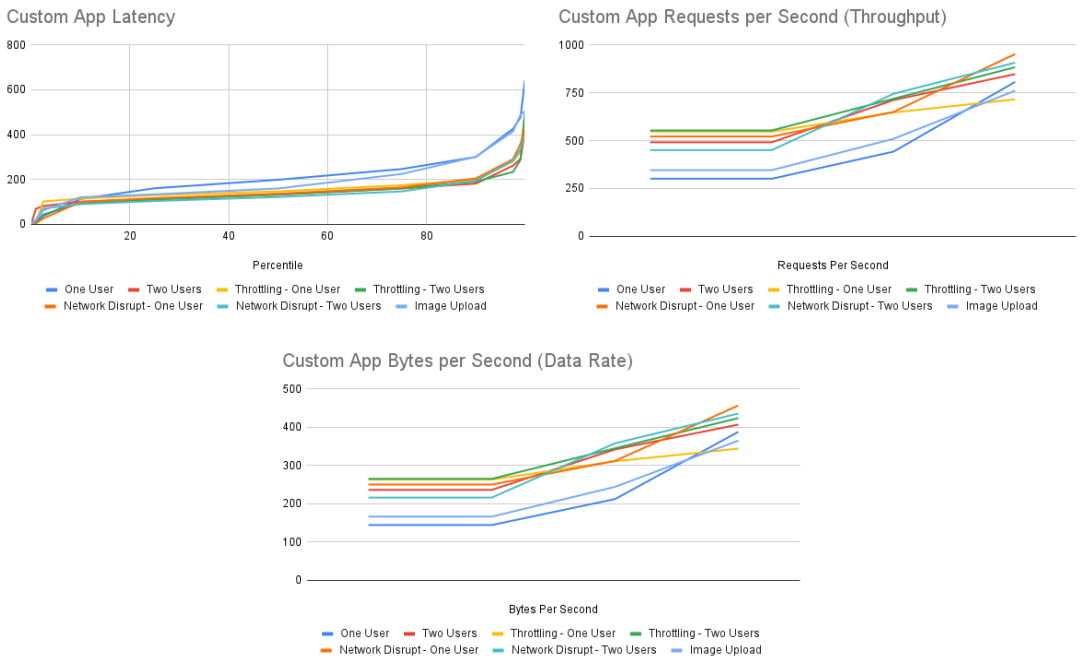

Figure 1: Internet Metrics for Custom Document Collaboration Application

The latency for our custom document collaboration application is fairly consistent across experiments. We can see from the top left image of Figure 1 that the highest latency for image upload, however, in general, latency remains low across experiments. Though the custom application is run in the browser, there is likely a less pronounced effect of throttling and network disruption because the application is running locally on the machine.

We also see when looking at throughput, in the top right of Figure 1, in conjunction with latency that though our experiments with one user typing and uploading an image have the lowest number of requests per second, they also have the highest latency. This demonstrates that these two experiments seem to put the highest strain on the system. By contrast, though all the other experiments (two users typing, throttling, network disrupt) all require the system to deal with more requests per second, the latency is lower.

The data rate is the speed of data transmission over some specified time period over the network application, in this case, bytes per second. This follows from our discussion of the latency and throughput of our application. Experiments with one user and image upload seem to have the lowest data rate, in comparison to all other experiments, meaning that data in these experiments is transmitted more slowly.

4.1.2 Google Docs

In looking at the latency of Google Docs, we see that from Figure 2 that we experience the highest latency when two users are typing simultaneously and also dealing with other competing traffic or network slowdown (through throttling). We also see a latency spike when one user is typing and experiences some network disruption. Both of these are cases where we would expect to see some slowdown. Interestingly, however, we do not see the same type of network spike when

one user is typing and experiencing network slowdown, or two users are typing and the network connection is disrupted.

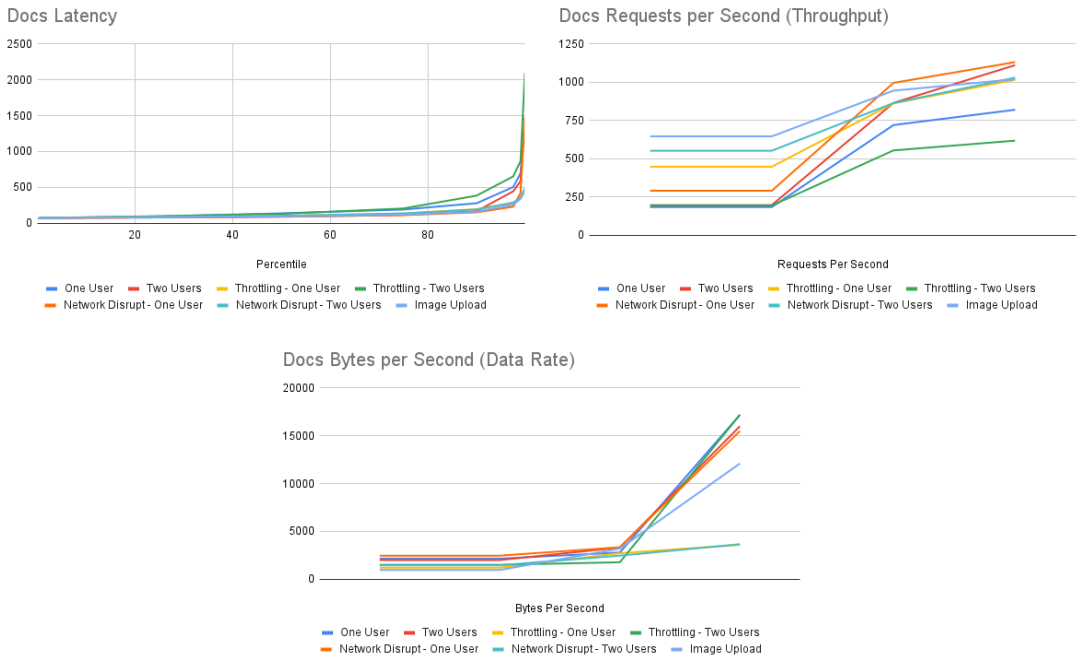

Figure 2: Internet Metrics for Google Docs

Our throughput experiment shows us that with Google Docs, we see the highest number of requests per second when the network is disrupted for a single user, then followed by when an image is uploaded. Docs does experience higher latency when a single user is typing on a document, showing that this is putting some strain on the application, however, it is still able to deal well with the higher number of requests per second.

We see the lowest throughput when Docs is competing with external traffic and multiple users are collaborating on a single document, which is consistent with where we see the highest latency in Google Docs. This shows that having multiple users on a single document and network slowdown puts the highest amount of strain on the application.

Though we see the highest latency with a single user and multiple users experiencing network slowdown, these two experiments also have the highest data rate, meaning the rate at which these experiments processed the data was the fastest. It shows the strain that these two experiments put on this application that even though data is processed the fastest, they still experience the highest latency. By contrast, when multiple users are collaborating on a single document and not experiencing network slowdown, we see that data rate remains high and throughput remains high, but the experiment's latency is low, which is ideal for this system.

4.1.3 Notion

Notion seems to experience the highest latency overall when an image is being uploaded, followed closely by when the application experiences network throttling. This may be due to that Notion is not a text editor, but rather a productivity application that is able to also function as a text editor with some key differences. Notion expects either a command denoted by a "/" or text at each line. Notion also makes suggestions depending on what is typed. Thus, when an image is uploaded, the system must first identify that a user is attempting to upload an image with the /image command and then insert a box where a user can upload their image and then process the image, adding latency to the process. When the application experiences slowdown or throttling, it still must process all of these additional commands. By contrast, our custom application as well as Microsoft Word and Google Docs do not expect commands within the text editor and in fact, cannot process commands in the text editor, so do not experience this additional latency.

Though Notion experiences the highest latency with image uploads, this does not seem

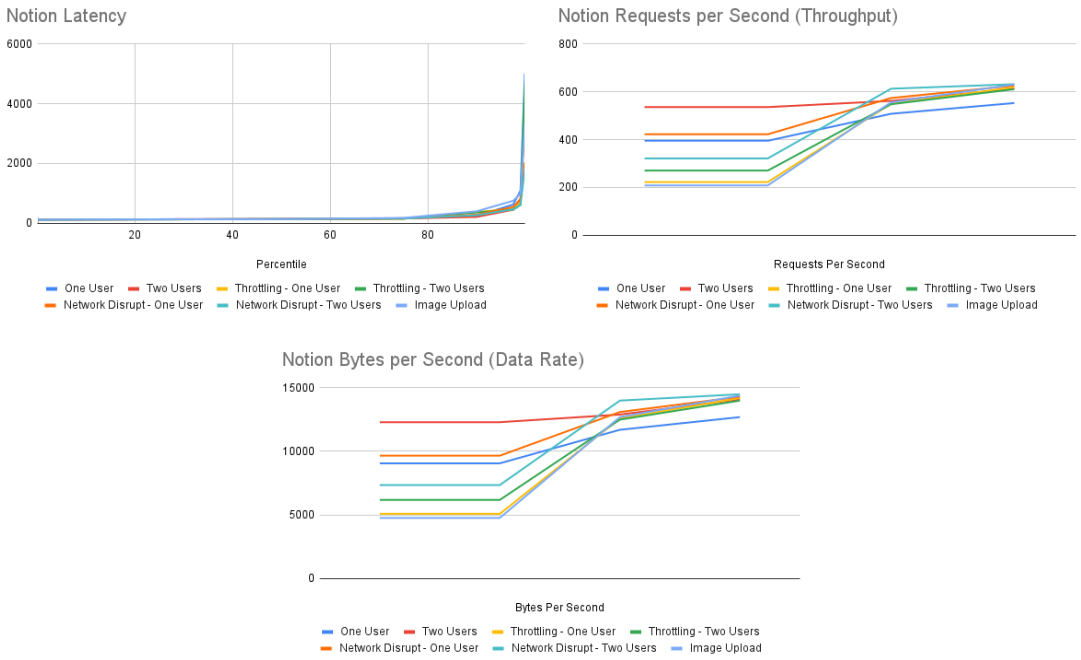

Figure 3: Internet Metrics for Notion

to significantly affect the rate of transmission. Rather, throughput is fairly consistent across all experiments with the lowest throughput experienced when a single user is working on a document.

The system also experiences the lowest data rate when there is a single user working on a Notion document. This low data rate could be part of why Notion's throughput is the lowest for when a single person is working on a document, but because we do not experience additional latency, this may not be reflective of worse performance. We expect that when a single user is working on a document, the application would not need to process as much data as if there were an image being uploaded or if multiple people were collaborating on the document at once. Since less data needs to be processed, the data rate and requests per second can be lower, without there being an increase in latency or a drop in performance.

4.1.4 Microsoft Word

It is important to note that in order to test Microsoft Word fairly against the other web applications, we tested Microsoft Word Online which is available in the browser. The recommended use of Microsoft Word is through a downloaded software application, rather than in the browser, however, the downloaded application does not allow real-time collaboration.

Microsoft Word Online experiences the highest latency when two users are collaborating on a document. This is a relatively new feature for Microsoft Word. Previous additions to the application do not have this feature and this feature is only available on the Web. While we would expect to see consistency with throughput and higher requests per second when two users are collaborating on a document leading to higher latency, this does not seem to be the case. Thus, we do not think network congestion is necessarily causing the increase in latency. Rather, it may be server response time or synchronization time which are causing this increase in latency.

Interestingly, we also see an increase in throughput and in the data rate when the network is disrupted. This could suggest that there was a bottleneck somewhere that the network disruption removed. If this is the case, packets were likely dropped and rerouted through less congested links, which would not necessarily mean improved performance. The data rate for all other experiments is consistent.

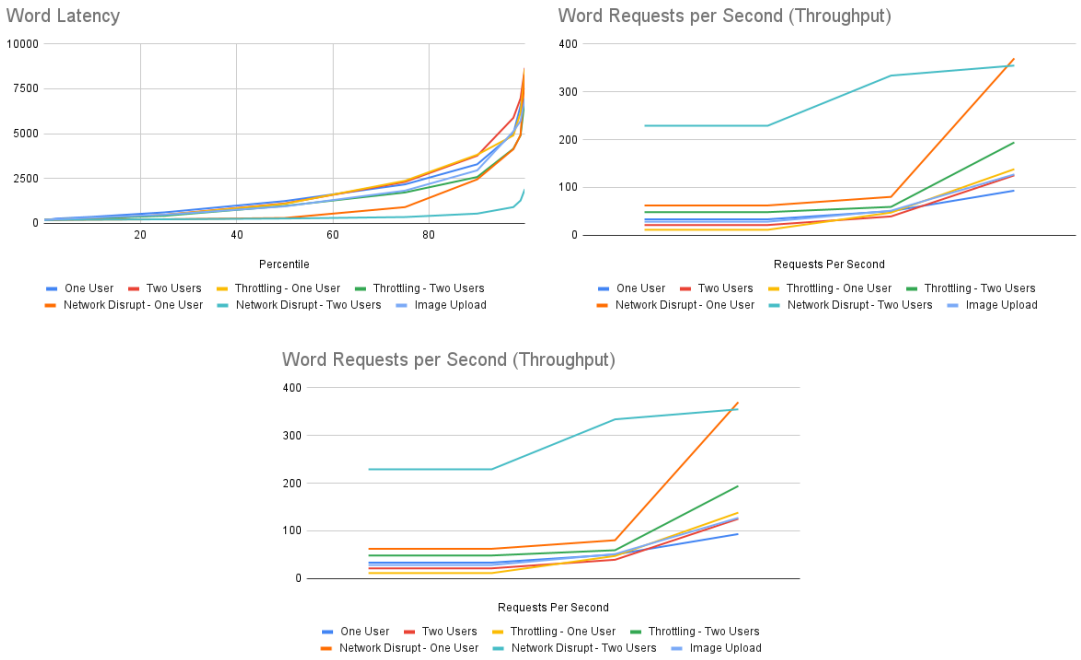

Figure 4: Internet Metrics for Microsoft Word Online

4.2 Analysis

Overall, we see the best performance with our own custom application, however, this could have a number of explanations that are not necessarily related to internet performance. There are several likely causes of this unrelated to network or software. First, the custom application has a significantly smaller scope of functionality. It is meant to be a lightweight text editor that can be accessed by several users at once. Second, because the software is running locally on the same machine as the user and as the database, there is little to no time spent moving between machines, as would be necessary on an enterprise-scale, production application. That said, we still think it is interesting to observe how third-party production applications behave in relation to our application.

We see that when only a single user is working on a document (no real-time collaboration), Word's latency is significantly higher than all of the other applications, then followed by Notion and Docs as in Figure 5. In terms of latency, Docs is only slightly slower than our custom application. This, combined with Word's having the lowest throughput and data rate makes it the bottom performer in terms of a single user working on a document. We can see that in terms of throughput, Docs and Notion perform similarly at the 50th percentile, however, at the 99th percentile, Docs processes significantly more requests per second, rivaling our custom application. And when we look at the data rate, though Notion seems to process more bytes per second up until the 90th percentile, Docs does at that point have a faster data rate.

In terms of latency, having two users collaborating on a document does not change much in terms of performance for our four test applications as seen in Figure 6. Word still has significantly higher latency than all other applications. Also similar to our previous experiment, Word has the lowest throughput and data rate (data rate is tied with custom application). In terms of requests per second, Notion stays largely consistent across percentile. While Docs's performance in terms of throughput seems to stay consistent regardless of whether a single person is working on a document or two people are working on a document. In contrast to performance with a single person typing, Notion's data rate is consistently higher than Doc's until the 99th percentile.

Notion having this higher data rate might suggest that Notion is more optimized for several users working collaboratively on a document, despite having higher latency. However, this could also be attributed to Notion making more requests per second for every line typed in a document because that is how commands are given versus in Docs, where the document itself is only a text box and commands are not built into the text box itself.

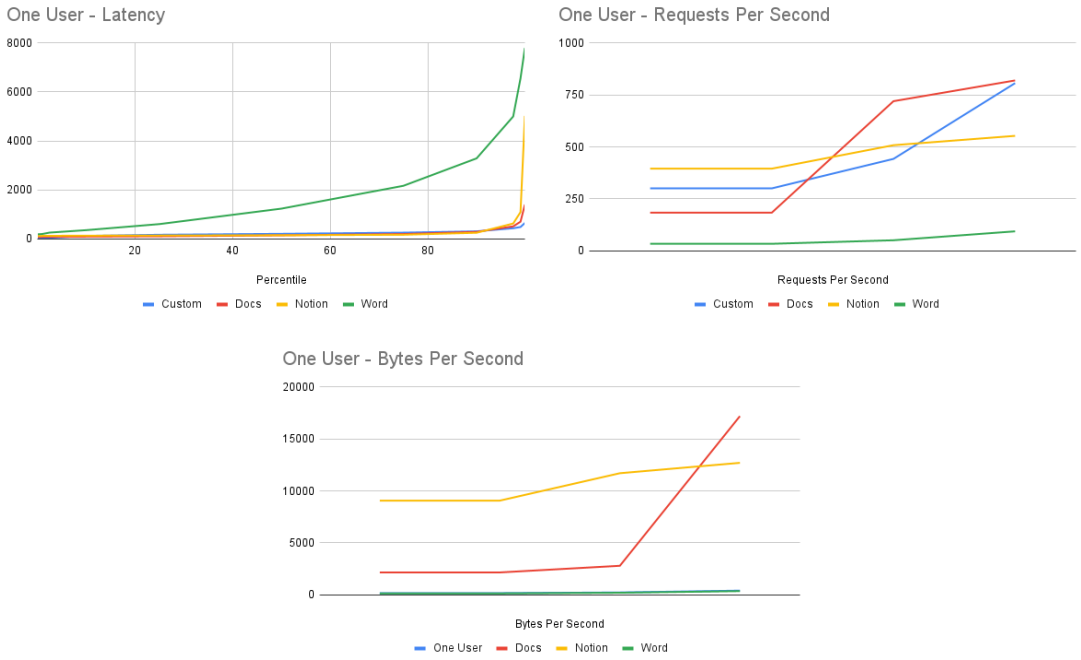

Figure 5: Application Metrics for Single User Typing

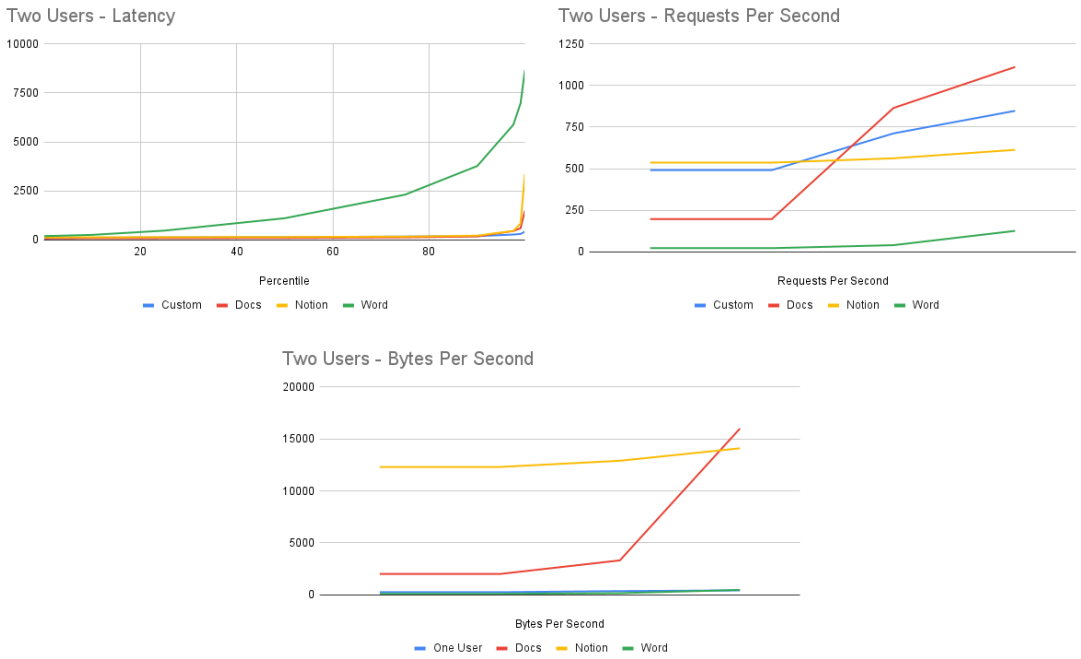

Figure 6: Application Metrics for Two Users Typing

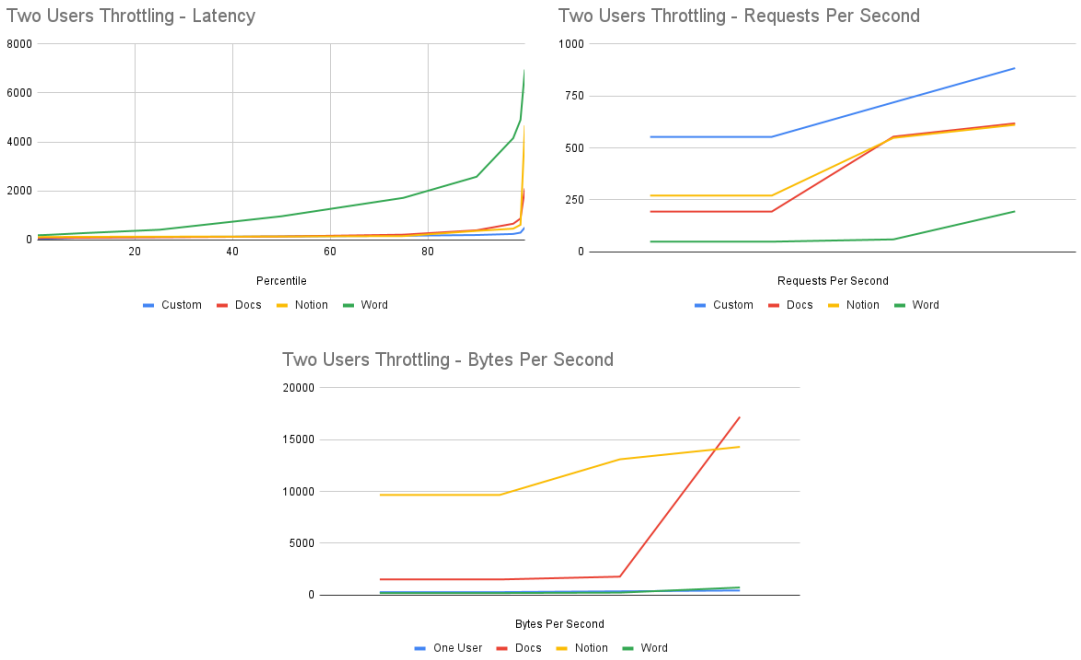

Figure 7: Application Metrics for Two Users Typing with Competing Traffic

We see in Figure 7 that introducing competing network traffic does seem to change performance for these applications. Latency does increase for all the production applications and Word's latency remains the highest, but where Docs and Notion might have had comparable latency previously, Notion's latency is significantly higher when there is competing traffic on the network. In terms of throughput, when throttling is introduced, Notion and Docs have very similar performance overall. The data rate, by contrast, is similar to just having two users, however, both Notion and Docs are not able to process as many bytes per second. The data rate slows with throttling.

Interestingly, Docs seems to perform better, in terms of latency, than even our custom application when an image is uploaded as in Figure 8. Notion's latency is not as high as it is when throttling is introduced, but Notion's latency does seem to increase with image uploads. Word, as has been consistent throughout our experiments still has the highest latency. Docs also performs the best in terms of throughput when images are uploaded. It is able to process the most requests per second, outperforming both Notion and our custom application (and Word).

5 General Observations

In general, our observations suggest that between the three production applications, Google Docs provides the best option and Microsoft Word provides the worst in terms of internet metrics for document collaboration. Microsoft Word has consistently high latency and low throughput and data rate, which results in slower performance that can actually be observed while working on Microsoft Word in the browser. That we are working on Microsoft Word Online may be an important factor here. We must consider that Microsoft Word was not built to be worked collaboratively in this way and is most popularly used as a software application where users may collaborate, but not in real-time. This may still be the way that most users interact with Microsoft Word suggesting that perhaps real-time collaboration is not a key priority for Word.

Notion's performance is generally good, though does not rival Google Docs. There could be many reasons for this. We must remember that Notion is not a text editor, nor a document collaboration application, though it can be used in this way. Notion seems to make many more requests per second, which may have little to do with a user or users typing on the document. Notion is looking for specific characters in order to interpret them into specific commands that the user may want. This is not the case with Docs or Word, where the text editor is only a space for users to type or upload images. Notion encourages users to make use of these commands

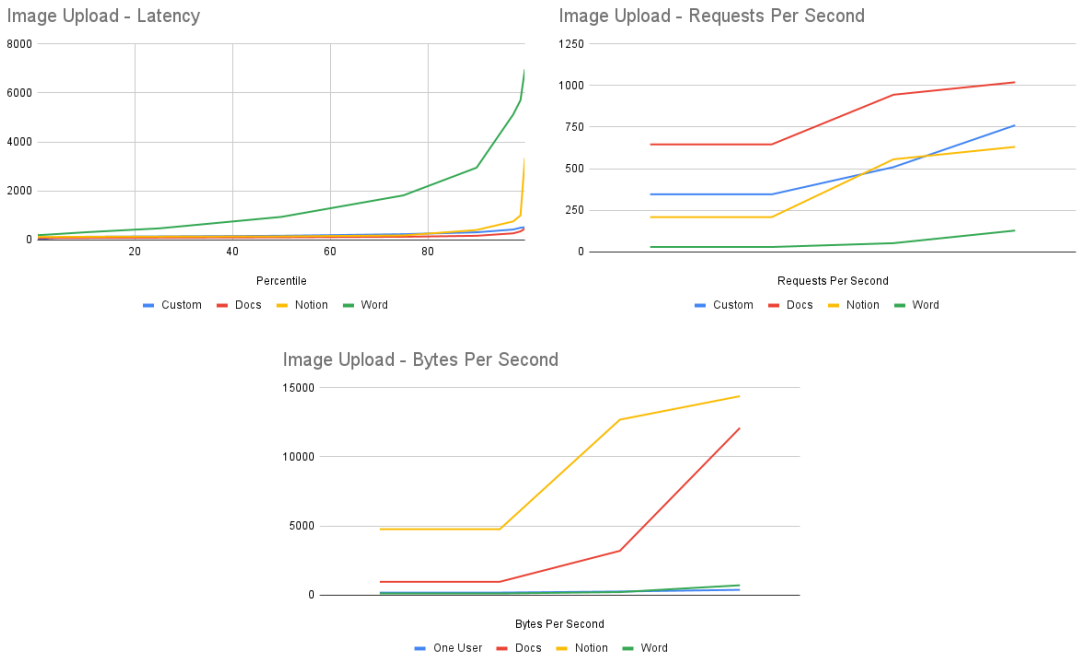

Figure 8: Application Metrics for Uploading an Image

in order to organize their pages, whereas at their core Docs and Word are giving users a space to write. This additional functionality seems to be where reduced performance is introduced when thinking about Notion only in terms of internet benchmarks and collaborative document editing.

Google Docs seems to have the best, most consistent performance out of the three applications when thinking about latency, throughput, and data rate. It often rivals our custom app, which is impressive given that requests to Docs still have to travel from machine to machine because it is a production application and our custom application is not. That said, there are points at which Notion is able to still outperform Docs suggesting that it is not *always* better to use Docs over Notion especially when we consider other facets of Notion's functionality. Still, when we consider only internet benchmarks, it seems clear that Docs has the best, most consistent performance across these three applications.

6 Future Work

We have mentioned many times previously that some of Microsoft's lacking performance is likely due to the fact that we are using Microsoft Word Online. It would be interesting to consider whether or not Microsoft Word software has these same internet performance issues. That said, these same experiments may not be able to be carried out in the same way because Microsoft Word does not have the same real-time collaborative functionality as its in-browser counterpart.

We tested these applications primarily as text editors, but they do have other functionalities, and both Google Docs and Microsoft Word are part of larger text-editing suites. It would be interesting to consider other functionality, for example, graph and chart generation, spreadsheets, etc. to see if those other functionalities face the same issues, or if that functionality is somehow optimized because decisions have been made that those are more important etc. It would also be interesting to test other applications in the suite, Google Sheets and Microsoft Excel, for example, to see if we get similar results.

7 Code Repository

All code for this project can be found here: https://github.com/kuhuhalder/document-collaboration-app

8 Conclusion

Overall, we were able to successfully observe the performance of important internet benchmarks of several popular document collaboration applications and understand and reason their advantages and disadvantages. We suspect that document collaboration latency, throughput, and data rate are under-studied with respect to their video conferencing and video streaming counterparts because the problems one may face such as latency, such as buffering are much less pronounced than they might be on, for example, a video stream. However, these applications are still widely used for users' work and personal matters, and hugely important as tools and it is vital to understand the advantages and disadvantages faced with each of them, especially as they continue to improve over time.

References

[1] Michel, O., Sengupta, S., Kim, H., Netravali, R. & Rexford, J. Enabling Passive Measurement of Zoom Performance in Production Networks. *Proceedings Of The 22nd ACM Internet Measurement Conference*. pp. 244-260 (2022), https://doi.org/10.1145/3517745.3561414

[2] MacMillan, K., Mangla, T., Saxon, J. & Feamster, N. Measuring the Performance and Network Utilization of Popular Video Conferencing Applications. *Proceedings Of The 21st ACM Internet Measurement Conference*. pp. 229-244 (2021), https://doi.org/10.1145/3487552.3487842

[3] Zhang, C. & Liu, J. On Crowdsourced Interactive Live Streaming: A Twitch.TV-Based Measurement Study. (2015)

[4] Twitch. Twitch Inspector. https://inspector.twitch.tv/#/

[5] Richard Stanway. TwitchTest. https://r1ch.net/projects/twitchtest

[6] WebRTC. Real-time communication for the web. https://webrtc.org/

[7] Google Chrome Web Store. Web Vitals. https://chrome.google.com/webstore/detail/web-vitals/ahfhijdlegdabablpippeagghigmibma?hl=en

[8] Chrome Developers. Lighthouse Overview. https://developer.chrome.com/docs/lighthouse/overview/

[9] Kumar, N. How to Build a Google Docs Clone with React, Material UI, and Firebase. (2022)

[10] Web Dev Simplified. How To Build A Google Docs Clone With React, Socket.io, and MongoDB https://www.youtube.com/watch?v=iRaelG7v0OU

[11] NPM. Autocannon. https://www.npmjs.com/package/autocannon

[12] Chrome Developers. DevTools. https://developer.chrome.com/docs/devtools/

[13] Chrome Developers. Network Features Reference. https://developer.chrome.com/docs/devtools/network/reference/