Measuring Complexity on the Web

Erik Nadel[1], Kartik Shetty[2], and Han Bao[3]

[1]Worcester Polytechnic Institute, einadel@wpi.edu
[2]Worcester Polytechnic Institute, kshetty@wpi.edu
[3]Worcester Polytechnic Institute, hbao@wpi.edu

Abstract— **The internet has changed since its first inception. Content being delivered online has become more interactive and dynamic. Every company wants to keep you hooked on the page and provide ease of use for every browser and user. In a previous work, authors Michael Butkiewicz, Harsha V. Madhyastha, and Vyas Sekar identify a set of metrics to characterize the complexity of websites both at a content-level and service-level[1].**

For this paper, we use the set of metrics established in the previous work to compare the complexity of the web in 2016 as compared to the results found in 2011. We found that the web had become more complex according to their metrics, but some trends had also stayed the same. Some categories are still more complex than others and that rank is not a strong indicator of complexity. However, we were not able to conclude that number of objects was related to the load time of a web page given our data.

I. INTRODUCTION

In the original paper by Michael Butkiewics, Harsha V. Madhyastha, and Vyas Sekar, they presented a comprehensive measure-driven study of the complexity of web pages in 2011 and its impact on performance. They measured roughly 1700 websites from four geographically distributed locations over a 7 week period. In the analysis, they focused on the landing pages of the websites chosen and chose the websites across a variety of ranks and categories.

To quantify websites, they used HTTP Archives(HAR), abbreviated as HAR. Browsers can export HAR files to show a log of a browser's activity when loading a web page. This includes the requests, responses, headers, content, and other useful metrics. This format is based on JSON and allows for easy analysis of web pages.

To analyze the web pages, they first quantified the complexity of a web page based on content fetched. For the complexity of content, this included number of objects fetched, the sizes of the objects fetched, and the types of content. Originally, they found no distinct difference in features when comparing these metrics across rank. However, they found a difference when comparing website categories. News websites, for example, loaded a significantly higher number of objects than others and Kids/Teen websites loaded a larger fraction of Flash content.

In addition to content complexity, they inspected where the content originated from. This involved looking at how much content came from non-origin servers and how significant it contributed to the number of bytes fetched.

After analyzing based on content complexity, they observed service complexity. Service complexity involves metrics that affect the time to download and render a web page. They found that the the number of objects fetched was the most dominant factor of client-perceived load times rather than the total number of bytes fetched to render a web site.

II. PROJECT BACKGROUND

For our experiment, we verified the key findings of the previous researchers. Since the experiment was performed in 2011, we would be comparing the differences in the key content and service complexity metrics. Using the same methods of data gathering, we compared the following content complexity results: median web page requests, contribution of various content types, number of objects loaded by category, fraction of objects with respect to bytes, and flash content. For service complexity results, we compared: the number of servers loaded, number of non-origin servers, and fraction of object types served from non-origin.

In verifying these results, we were able to follow the setup of the original paper with only a difference in some of the applications used. After collecting all of the data, we replicated several of the charts found in the original paper and used the charts to make conclusions based on the differences.

III. METHODOLOGY AND SETUP

A. Original Setup

In the original paper, the authors collected the data using a clean Firefox instance with no UX addons. To collect the HAR data, they used the firebug extension which supports the Firestarter extension for the actual export of HAR data. In summary, whenever Firefox visited a web page, it automatically created an HAR file for the request. Next, the authors ran the Firefox instance on three Amazon EC2(Elastic Computing) nodes and one home computer. One instance was in Europe, one in Asia, and one in US-EAST.

At each instance, they ran a measurement agent that periodically (60 seconds) selected a random web-page from the list of 2000 URLs. Each run would generate an HAR file that would be a request log of the landing page of the given URL. These measurements were repeated for 9 weeks. To filter bad requests, if a HAR file had a bad request, 0 bytes received, or loaded instantly, it was removed from the data set. In the end, the authors were left with around 1700 sites from the original list of 2000.

B. *Our Methodology and Setup*

In replicating the experiment, we were able to achieve the exact set up with a few differences. For obtaining the Amazon EC2 nodes, we created three separate accounts to take advantage of the free-tier and have all of the nodes running. For the home computer, we used a digital ocean server that we had already acquired. We did not use Firefox for the HAR file retrieval, however. Instead, we used a docker image that had a pre-installed browser with an HAR exporter. When we tried using Firefox by itself, our node kept having issues and locking up. The docker image allowed us to easily deploy the measurement agents across the different nodes.

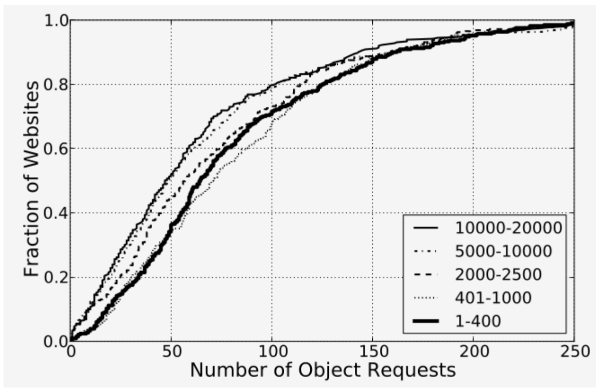
After collecting data for 9 weeks, we filtered out all of the junk HAR files using the same process as the original paper and ended up with around 1800 sites. Once we filtered out the data, we parsed the information from the HAR file that was needed into a spreadsheet and then plotted to recreate the key diagrams from the original paper. With the graphs recreated, we compared the results and made our analysis from there.

IV. Results and Comparative analysis

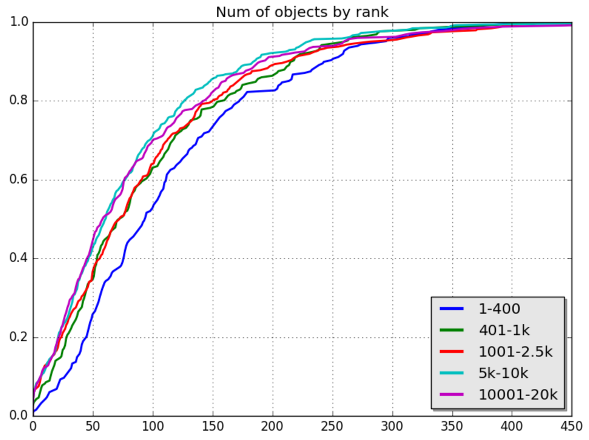
We consider content level metrics and service level metrics to understand the complexity of websites. Content level metrics deals with the number of objects requested for each MIME type. Service level metrics deals with the sources used to request and render the objects for a website. For example: www.abc.com would use internal domains like images.abc.com to get images and then use xyz.com to deliver third party content like javascript source files. In each case we present a breakdown of websites across rank (eg: top 1-400 vs 10000-20000) and also across different categories (eg: news,entertainment etc.) and then compare it to the results obtained in the original implementation. We begin by analyzing the metrics and comparing it to the results of the original implementation. And then we end by analyzing the performance (load time).

A. *Content complexity*

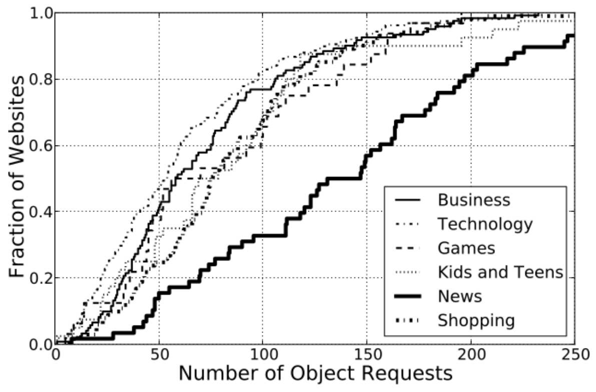
Number of objects: We start by analyzing the Figure 1 which shows the total number of objects i.e the number of HTTP GET requests made to render a web page. In Figure 1b, we observe that the top 1-400 ranked websites require more number of objects to render a web page. We observe that the distribution across ranks is similar to the one in original implementation (Figure 1a). We observe that the top 1-400 require more objects, but across other ranked bins the distribution is qualitatively and quantitatively similar. Across the categories, we note that the adult category stands out, as it requires more number of objects. It is followed by the news and the entertainment category. It is difficult to make a comparison with the original work as they consider different set of categories to analyze their results, but we observe similar trend with respect to news category.



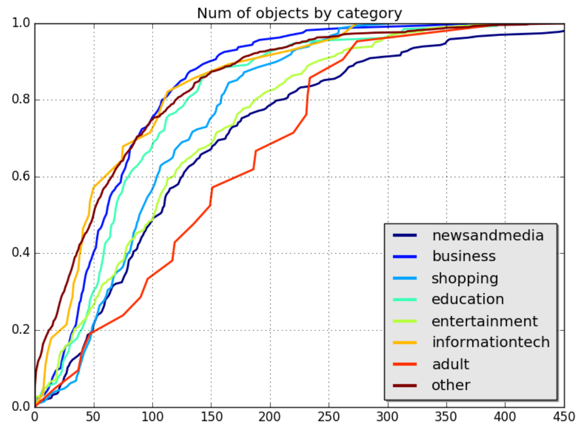
(a) Original by rank



(b) Our finding by rank

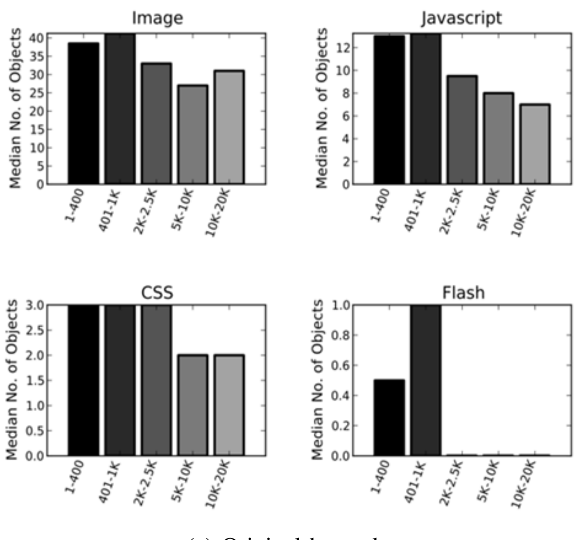


(c) Original by category

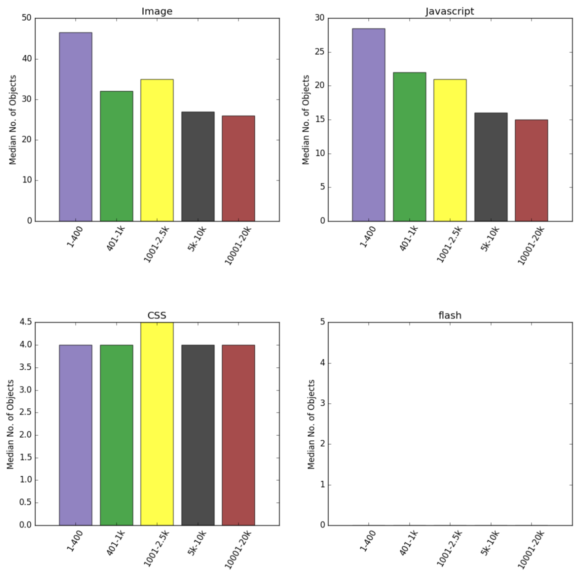


(d) Our finding by category

Fig. 1: Total number of objects loaded on the base web page of websites across rank ranges and categories.

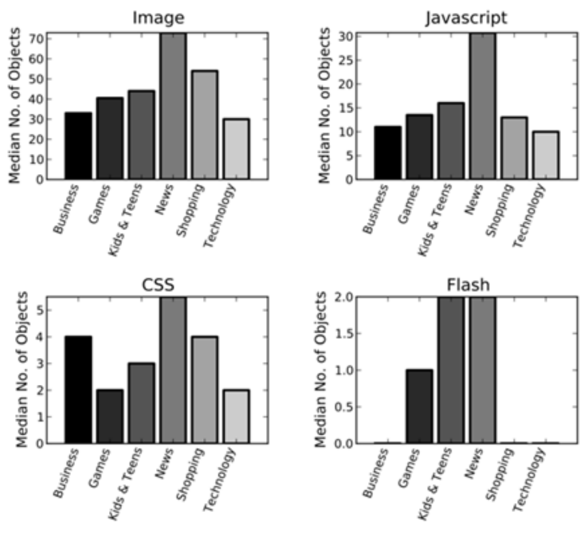


(a) Original by rank

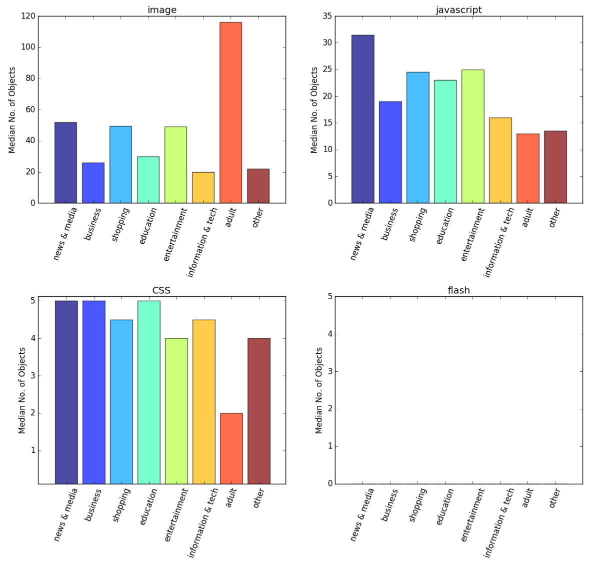


(b) Our finding by rank

Fig. 2: Median number of requests for objects of different MIME-types across different rank ranges.



(a) Original by category



(b) Our finding by category

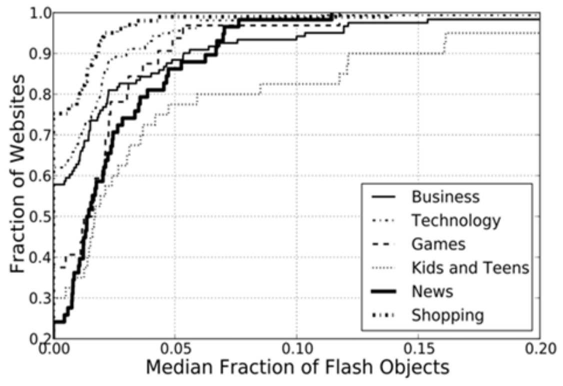
Fig. 3: Median number of requests for objects of different MIME-types for different categories.

Type of objects: After analyzing the total number of object requests, we next consider their breakdown by content MIME type. As is shown in Figure 2b, in comparison with the result in the original paper Figure 2a, we found that the number of image objects is around the same level as before. But image objects in rank 1-400 were more than any other rank ranges, which is different from the original paper. As for javascript objects, we can clearly see in the graph that as the rank range goes down, the number of javascript objects decrease. One more important thing is that the number of javascript almost doubled compared to the old number in the paper. The number of css objects did not change much, while flash objects basically disappeared. In the original paper, we can see that there are about half of the websites in the first two rank range have one flash. But according to our data, only 43 of 1791 are still using flash.

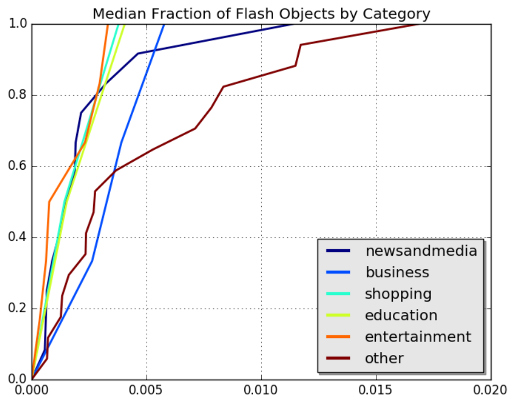
Figure 3b shows the number of objects requested of various content types across different categories. Adult category is probably not included in the original paper, but in our finding, the median number of image objects adult websites have is over 100, which is much more than other categories. In addition, we can see in any category that the median numbers of flash objects are all 0, which does not surprise us though.

Then we analyzed the websites which are still using flash, as Figure 4b shows. We can see that though flash objects still exists, the fraction of flash objects has decreased significantly (less than 2%), which was about 1/10 of 5 years ago when the original paper was written.

Bytes downloaded: Figure 5b, shows the byte contribu-

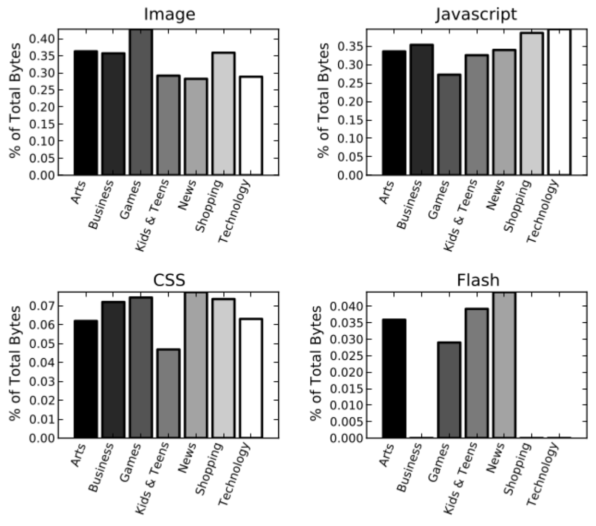


(a) Original

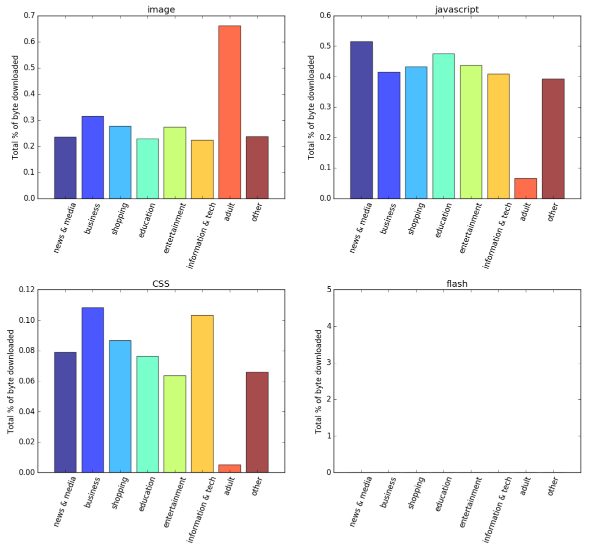


(b) Our finding

Fig. 4: Fraction of objects accounted for by Flash objects, normalized per category.



(a) Original



(b) Our finding

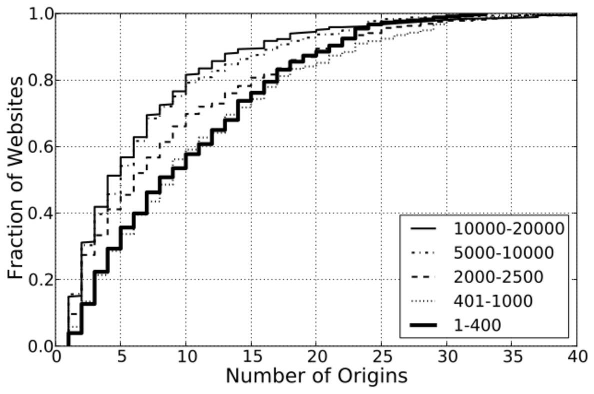
Fig. 5: Median normalized contribution of different MIME types to total bytes downloaded.

tion for different MIME types, which gives us a different perspective than just considering the number of object requests. We have considered median values across different categories. We observe that in the Figure 5b, the sum of image and java script byte contribution dominant with respect to total byte contribution. It accounts for more than 60% across all the categories. Also, images contribute to majority of the bytes for the adult category (around 70%). By comparing the results of the original work(Figure 5a) with ours, we observe that the byte contribution of flash has disappeared across all the categories. Also, javascript contributes around 35% across all the categories compared to 25% in the original work.

B. Service complexity

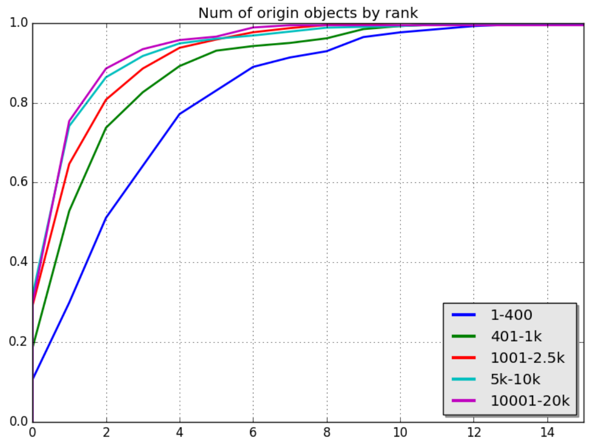
A web page generates HTTP GET requests to more than one servers to render its page. We divide the sources into origin and non-origin based on the domain name of the source server. A origin server or source is under the same domain as the website. For example: images.foo.com would be considered as a non-origin source for foo.com. A non-origin server or source is a third party server which does not come under the domain of the website. For example:xyz.com would be considered as an non-origin with respect to foo.com. We now introduce the service complexities which deals with the origin and non-origin sources for a website.

Number of non-origin services: Non-origin sources are used by websites to deliver third party content like static content, to track user activity (google-analytics) or use advertisement services to generate revenue. We have used the approach used by the authors of the original work to distinguish between an origin and a non-origin source. We would first identify the name servers of the website under consideration. In the next step we would determine the name servers for each of the source and compare it to set obtained from the above step. A source would be considered as an origin if it contains a name server in common with the website and would be considered as an non-origin otherwise. Looking at Figure 6a, we notice that there is a similar trend with respect to number of origins across ranks from the original work(Figure 7a). We see that top 1-400 websites use a higher number of origins to render content than other ranks. Looking across categories (Figure 6b) we notice that the news category does not prominently dominate with respect to other categories, which was not the case in the original work (Figure 7a).



(a) By rank

(a) Original



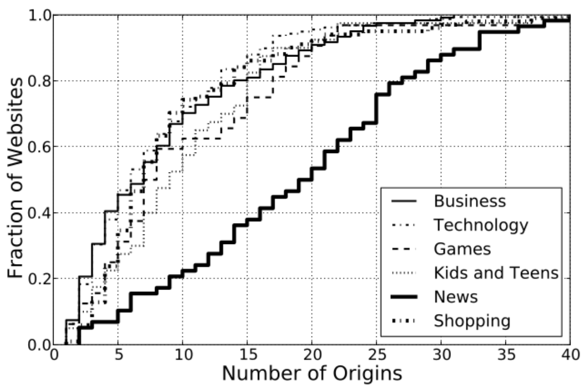
(b) Our finding

Fig. 6: Total number of objects loaded on the base web page of websites across rank.

Number of distinct servers: As is shown in Figure 8b, we can see that popular websites (in higher rank range) tend to use more distinct servers, which is the similar result to the result in the original paper (Figure 8a). In Figure 9b, we also found that News websites normally have more server, but not so distinguished from other categories as is shown in the original paper (Figure 9a). And adult websites are using less servers than others.

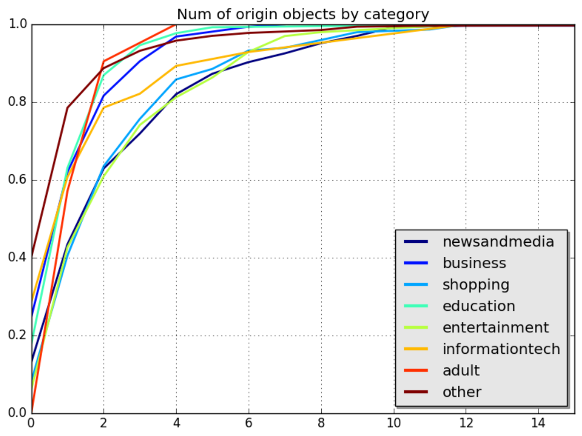
C. What do non-origins offer?

What are the differences between the object the websites put on their own server and non-origin servers? That is what we are going to find out. We check the URL of every object to find out which kind of server they are in. And we first calculate the median of each MIME type, then select the



(b) By category

(a) Original



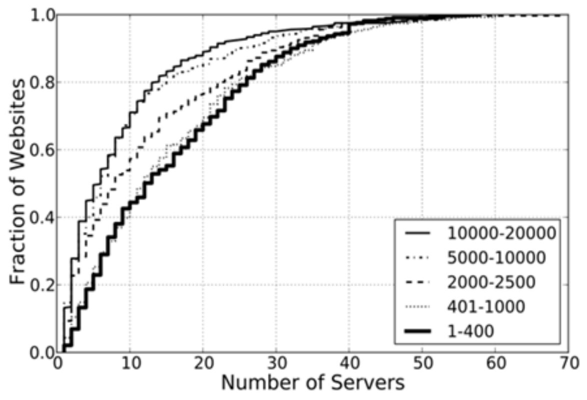
(b) Our finding

Fig. 7: Total number of objects loaded on the base web page of websites across categories.

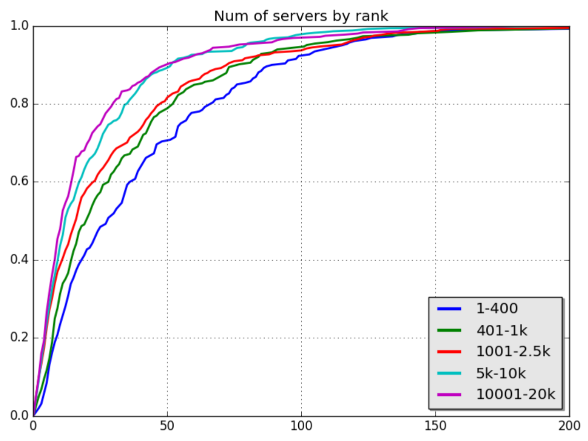
website for which the median value of each MIME type is close to the median value calculated previously.

Content type breakdown: Figure 10b shows the breakdown of the different content types served by non-origins,both in terms of the number of objects and their size in bytes of our median website. We can see that the fraction of image objects in byte contribution is smaller than the object contribution, which means image objects are actually smaller than javascript and css. And this is odd because typically images are larger than others. The result is same as is in the original paper. The author shows the figures of fraction by objects and bytes (Figure 11a and Figure 12a), and explains that many small gifs are fetched from non-origins.

However, our data is quite different. In Figure 11b and Figure 12b, we can see that though images are still the most objects, the number of javascript objects is very close to the image. And in Figure 12b, we can see that the average size of javascript is larger than image, which is different from the result in the original paper. In addition, we test it by counting and recording all types of images (shown in the table) and found that gifs are actually less than 1/3 of all



(a) Original by rank



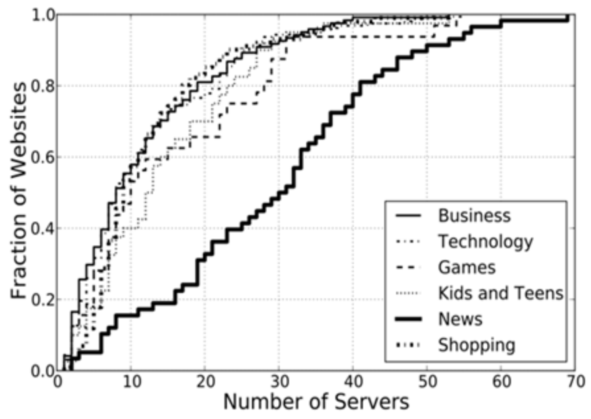
(b) Our finding by rank

Fig. 8: Number of distinct servers by rank range

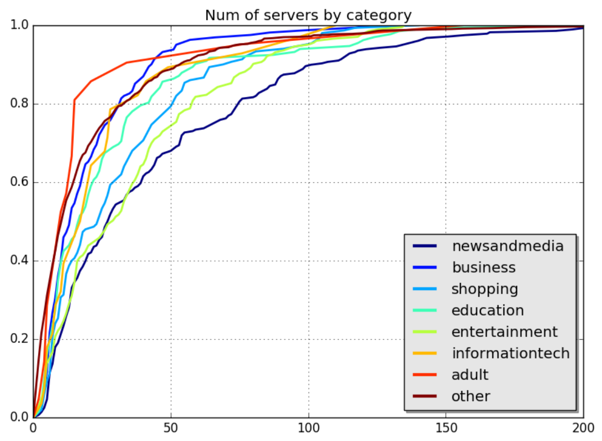
images. Therefore, we believe that gifs are not the reason for small contribution of bytes for today's websites.

Image Type	Number of Objects
image/jpeg	224181
image/gif	156116
image/png	152544
image/svg+xml	10057
image/jpg	1096
image/pjpeg	288
image/bmp	85
image/x-icon	35
image/x-png	10
image/webp	2
image/x-ms-bmp	1

Origin vs Non-origin: We understand the contribution of both the origin and the non-origin sources to a website by considering the respective percentage contribution of each MIME type to the median website. By looking at the Figures 13a & 13b, we notice that the percentage of java script objects from the origin has increased approximately 15% in comparison to the original work. But we observe a similar trend in comparison to the original work for the non-origin sources.

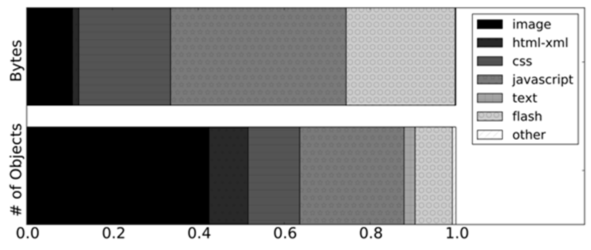


(a) Original by category

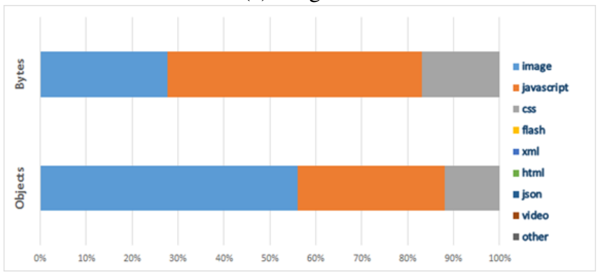


(b) Our finding by category

Fig. 9: Number of distinct servers by category



(a) Original

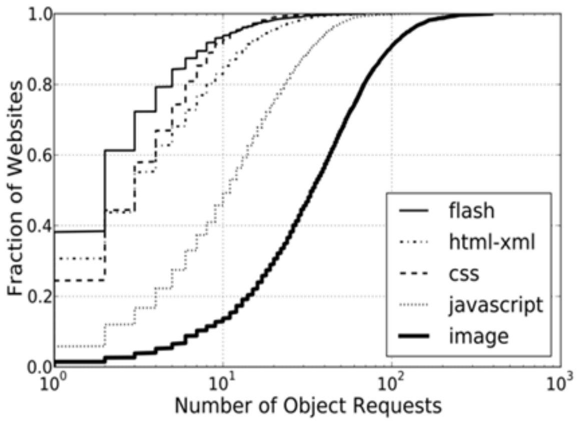


(b) Our finding

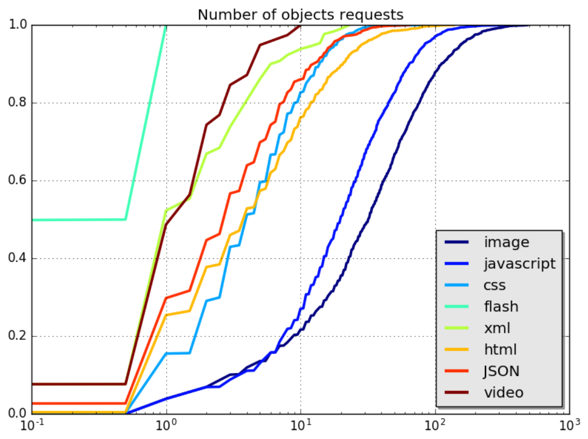
Fig. 10: Normalized contribution of objects from non-origin services in the median case

D. Load time

Load time for a webpage is an indicator of the performance of a website. Load time is determined by looking up at

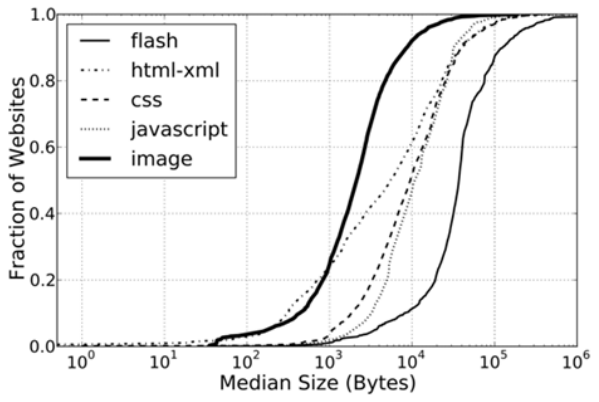


(a) Original

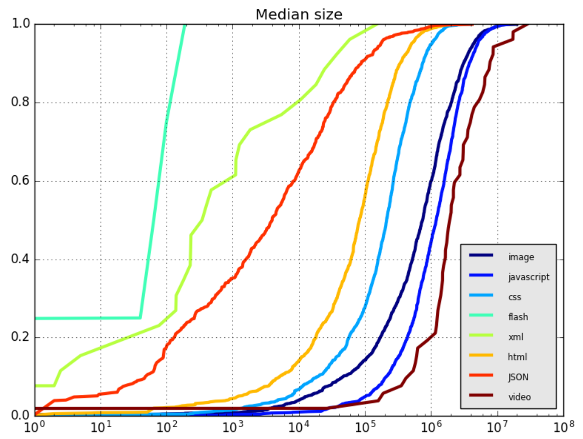


(b) Our finding

Fig. 11: Distribution across content types of the number of objects from non-origins

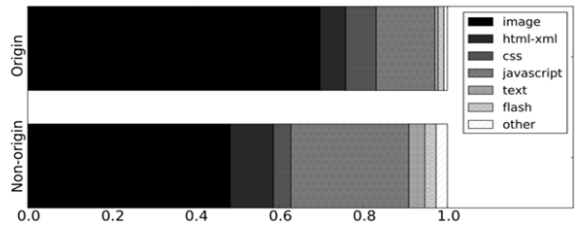


(a) Original

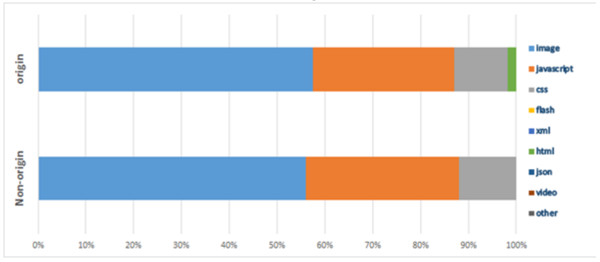


(b) Our finding

Fig. 12: Distribution across content types of the median object size from non-origins



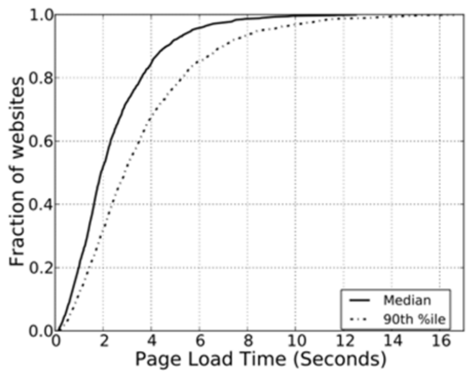
(a) Original



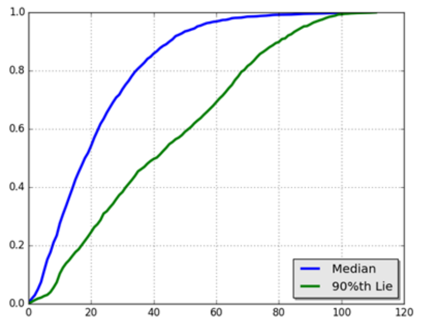
(b) Our finding

Fig. 13: Comparison of types of content served from origin and non-origin servers on the median website.

the RenderEnd parameter in the HAR file. We used the approach of the authors of the original work to determine relation between the load time and the complexity metrics that we defined above. The authors had established a correlation between the load time and the number of objects requested for a website and we wanted to confirm whether this relationship still existed. Looking at the Figure 16 we observe that the load time does not increase steadily as the number of objects increased, but on the contrary we found the load time to fluctuate as the number of objects increased. Looking back at Figure 1d, we observed that the news, adult and education categories sites load larger number of objects than the other sites. But in the Figure 15, we notice that the business and adult category sites took more time to load than the other sites. This could imply that there might not be a correlation between the load time and the number of objects today. Figure 14 shows a similar trend with the original work for the distribution of the median and 90th percentile value's of RenderEnd across websites. Also in the Figure 14b we observe that about 60% of the websites have a median load time of higher than 20 seconds. Although we observed unusually high load time for the websites, we are more concerned with establishing or confirming relationships between parameters.

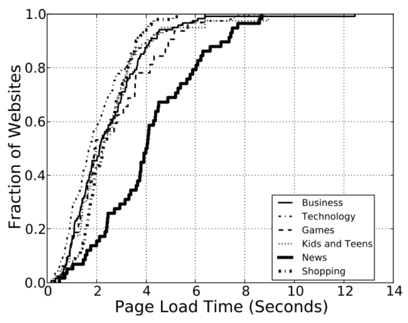


(a) Original

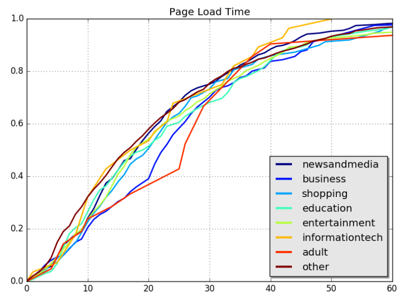


(b) Our finding

Fig. 14: Distribution of RenderEnd times across all websites.



(a) Original



(b) Our finding

Fig. 15: Page load times for websites of different categories

V. CONCLUSIONS

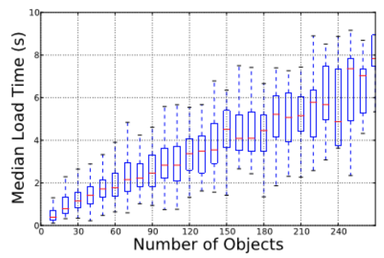
Overall, our project was successful in gathering data in the way that the original paper did, and comparing our data with theirs. Through our data, we can have a better understanding of the complexity on today's websites, and by comparison, we can see the changes happened these years and also trends in the future. However, we were not able to conclude that number of objects was related to the load time of a web page given our data.

By comparing our results with those in the original paper, we found that there is a big increase in the request number. The median of request number in the samples is about twice as the median in the original paper. In terms of different types of objects, median number of JavaScript objects also doubled in the past few years. And a JavaScript object normally contributes more bytes than other type of objects. Image objects still dominate fraction of objects, but not in bytes. In other words, image objects are actually smaller in size. As to flash objects, we found that less than 3% of websites in our sample are still using flash. And even though they use flash, the fraction of flash objects is quite small, less than 2%. In summary, what has changed in recent years is that JavaScript has doubled, while flash is disappearing.

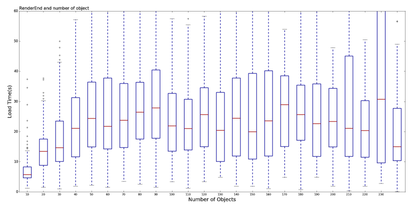
And by analyzing our result by categories, we found that news websites are still loading more objects than other categories, except for adult websites which loads large number of images. And news websites also get their contents from more servers than others. In the meanwhile, by analyzing our result by rank, we found that popular websites tend to have more objects and connect more servers especially their own servers.

VI. REFERENCES

1. Butkiewicz, M., Madhyastha, H. V., & Sekar, V. (2011, November). Understanding website complexity: measurements, metrics, and implications. In Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference (pp. 313-328). ACM.



(a) Original



(b) Our finding

Fig. 16: Box-and-whiskers plot between RenderEnd and number of objects.