Visualizing Reused Text Within Wikipedia

Matheus Avellar de Barros[0000-0003-2810-8710]

Universidade Federal do Rio de Janeiro, Rio de Janeiro RJ 21941-909, Brasil
matheusavellar@dcc.ufrj.br

Abstract. Text reuse within Wikipedia, although widespread, goes largely unnoticed by editors and readers alike, given that there are no readily available or builtin tools with the purpose of detecting such occurrences. In this work, we present a proof of concept web application aiming to facilitate the visualization of such reused text, built upon a previous work's dataset of reused text within Wikipedia. We find that the implementation of such an application is a feasible task: by accessing Wikipedia's web API and choosing a quick comparison algorithm – namely Myer's *diff* algorithm – we can compute comparisons directly on the client side, allowing for serverless infrastructures to implement this application. Henceforth, future works can adapt and improve upon this solution to create a more stable, reliable and accurate comparison system and its visualizations.

Keywords: Wikipedia, text reuse, text analysis, data visualization.

1 Introduction

Wikipedia is possibly the most well-known community-driven educational service of our time, receiving roughly 870 million unique visitors every month.[1] Having over 250 billion articles across all languages, it is to be expected that some repetition of text will occur, be it coincidental or done on purpose. Situations where this might arise are, for example, in articles relating to similar topics: both Mangaratiba and Itaguaí are municipalities of the state of Rio de Janeiro. It is therefore unsurprising that they have instances of reused text, more specifically the sentence "*is a municipality of the Brazilian state of Rio de Janeiro*".

Large-scale studies of such text reuse within Wikipedia articles are scarce. Because of this, the availability of tools to support the maintenance of reuse occurrences is also remarkably low. Using the previous example as an illustration of this issue, if it were to happen that the Brazilian state of Rio de Janeiro was renamed or annexed as part of another state, both the Mangaratiba and the Itaguaí articles would need to be updated separately. However, there would be no indication for editors if only one of them was rewritten, and the other was forgotten, even though both should always have identical – and up-to-date – sentences. It stands to reason that such a tool would benefit not only Wikipedia editors, but also its readers, who many times rely on the information present being accurate.

[1] https://stats.wikimedia.org/

Hence, we propose the concept of an application to aid in the detection and visualization of reused text modification, and present a proof-of-concept implementation of it.

2 Related Work

Text reuse on the broader web has been widely researched upon, usually focusing on combating plagiarism and tracking information flow. With a society that is increasingly more connected to the Internet, even in public spaces [1], the so-called *clickbait* titles and fake news stories have become ever more profitable for publishers online, considering the broad use of advertisements in such websites [2]. Disregarding real world impacts of manipulating media, websites may knowingly publish and republish deceitful content with the sole intent of earning advertisement revenue from unsuspecting readers. Therefore, the detection of deceptive information and its spread has been studied extensively, to an ever increasing accuracy [3]. However, generating fake news stories requires a level of human labor; to get around this and still earn advertisement revenue, many websites instead opt to programatically rehash and repeat content from other sources – be them legitimate or not –, causing issues such as users giving credit to a story to the wrong publisher, diverting revenue from legitimate writes. To detect this and other cases, in the last couple of decades, new text reuse detection algorithms have been created and improvements to existing ones have been made [4, 5].

With regards to text reuse specifically in Wikipedia, previous works have analysed near-duplicate clusters of sentences within articles in English [6], and direct copies of Wikipedia content in external websites, focusing on violations of Wikipedia's terms which prohibit profiting off replicated content [7, 8]. Subsequent improvements merged these two approaches and introduced a *text alignment* step in the processing pipeline, which benefited the detection of reused text both within Wikipedia articles, as well as outside Wikipedia [9]. With this extra step, there is greater fine-grained control over the reuse of text passages within sentences, making for a more accurate resulting corpus of replicated content.

3 Materials and Methods

In this work, we seek to build a user-friendly application focusing on the visualization of reused text in Wikipedia, which could aid in detecting eventual modifications to it. We propose the use of a faster algorithm for article comparisons, such as Myers' *diff* algorithm [10]. By doing so, we can take advantage of previous works' results without spending as much computing time to reach outputs, thus allowing the execution of the comparison on the client side, without putting a heavy load on the user's machine.

Development was compartmentalized so as to create a separation of concerns. At the furthest level away from the end user, there's the dataset used. We utilized the "Within Wikipedia Text Reuse" dataset produced by Alshomary et al. [9], made publicly available by them for further academic research.[2] That, in and of itself, after extraction and decompression, already added up to over 80 GB of JSONL[3] files. Due to limitations in computing power and otherwise time restraints, we decided it would be more appropriate to try and reduce the dataset for usage in our proof-of-concept implementation.

Thus, the next "layer" of the application is a filter layer, in which a Python script randomly picks only 1,000 lines from one of the over 40 GB JSONL reuse cases files. Before settling on a candidate line, it first checks if some words are present in the article body. The words it looks for are the following: "village", "town", "city", "census", "district", "municipality", "enzyme", "species". This semi-filter was put in place because of the huge and unexpected amount of pages listed with very similar subjects. That is, the proportion of Wikipedia articles relating to, to pick an example, cities (or towns, villages, municipalities, and districts) with reused content was much greater than pages that did *not* relate to those subjects. That also applies to animal and plant species, of which there are a ridiculous amount, and censuses, the articles of which contain the most amount of reused content, as they only ever need to have updated values and percentages. A posteriori, this makes sense: there are a very large number of, for example, places in the world with Wikipedia articles written about them, and it isn't to be expected that each and every one of them will have a uniquely written introduction. As such, it is very common to see copied pages with modified place names, as it makes for much easier writing and much quicker article creation. It is also to be noted that the authors of this work do not condemn these practices; it is how Wikipedia works, and it is very much within the goals of the Wiki project as a whole to reuse text that relates to the same, or very similar, things.

After the filtering done by the script, a manual selection stage was used with the goal of picking a handful – in the final project, 10 – of example articles that would be the most interesting for demonstration purposes. It is important to make the distinction between the infamous act of "cherry picking" data, which has the primary intent of skewing research results, and this manual selection stage, which merely aims to create a more visually interesting demonstration of the application. This manual stage is what led to the implementation of filters regarding cities, censuses and such in the above layer. Without the filters and the manual selection, there was a high rate of uninteresting reuse cases, which although also important in their own merit, was not the primary goal of demonstrating this work.

Following the manual stage, a regular JSON[4] file is produced, which includes information about: the ID of both Wikipedia articles that contain text reuse; the first and last 25 characters of the reused section, to mark the start and the end of the part of the article body we're interested in; a label to describe shortly what the article pair describes. It is relevant to note that the 25 characters were not always perfectly formatted. For some reason, the dataset used would have empty spaces where there should not have been. For example, it turned some single quotes, ′, into spaces. Therefore, the manual stage also became a "repairing" stage where these were fixed for the final JSON.

[2] https://webis.de/data/webis-wikipedia-text-reuse-18.html
[3] https://jsonlines.org/
[4] https://json.org/

One layer closer to the user is located a web server, programmed in Python, that simply serves the website – an HTML structure file, a CSS styling file, Google's "diff-match-patch"[5] JavaScript library file, and the output JSON file from the previous layer.

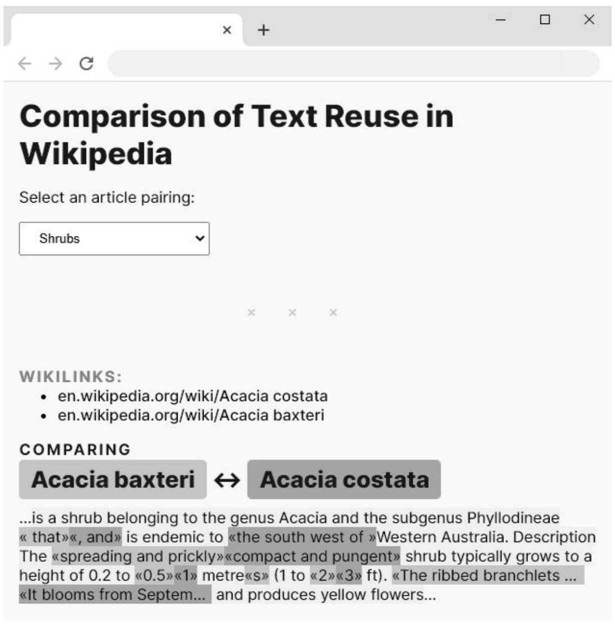
The final layer, the closest to the user, is the website application itself. By interacting with the website served by the previous layer, the user makes requests to the Wikipedia's API[6] which, given an article's ID, can be made to return that article's title and text body. After some cleaning up and filtering, the text goes through a trimming process, where the previously saved 25-character pairs describing the start and end of the text reuse section are located inside the complete article body, and the reused section is extracted. Finally, Google's optimized Myers' *diff* algorithm is applied to compare the extracted reused excerpts from both articles selected. The application then displays the difference between the articles, and highlights their similarities.

4 Results

At the top, a selection box allows the user to choose between labels describing pairs of articles from the filtered JSON mentioned in the previous section. When the user chooses a label, the website automatically sends an HTTP request to Wikipedia's API endpoint, and updates the text accordingly.

[5] https://github.com/google/diff-match-patch

[6] https://en.wikipedia.org/w/api.php



Comparison of Text Reuse in Wikipedia

Select an article pairing:

Shrubs

WIKILINKS:

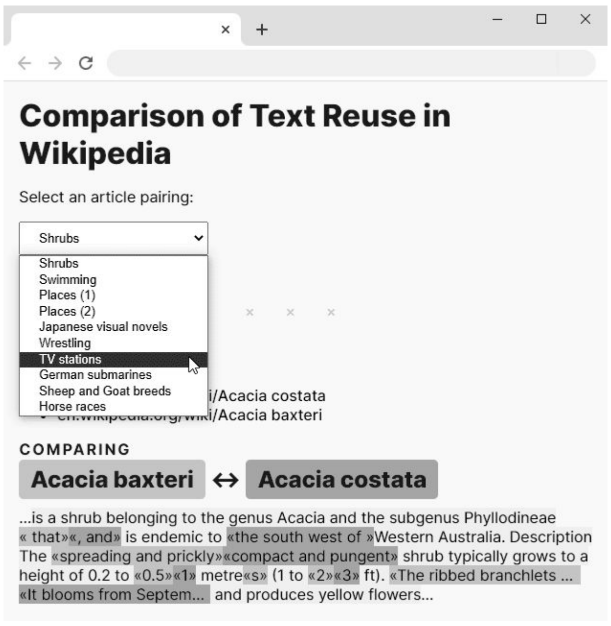
- en.wikipedia.org/wiki/Acacia costata
- en.wikipedia.org/wiki/Acacia baxteri

COMPARING

Acacia baxteri ↔ **Acacia costata**

...is a shrub belonging to the genus Acacia and the subgenus Phyllodineae « that»«, and» is endemic to «the south west of »Western Australia. Description The «spreading and prickly»«compact and pungent» shrub typically grows to a height of 0.2 to «0.5»«1» metre«s» (1 to «2»«3» ft). «The ribbed branchlets ... «It blooms from Septem... and produces yellow flowers...

Fig. 1. An example state of the website, comparing two articles with text reuse. The website style was inspired by Tailwind UI's "Feature Sections" templates.[7]



Comparison of Text Reuse in Wikipedia

Select an article pairing:

Shrubs

- Shrubs
- Swimming
- Places (1)
- Places (2)
- Japanese visual novels
- Wrestling
- TV stations
- German submarines
- Sheep and Goat breeds
- Horse races

i/Acacia costata
i/Acacia baxteri

COMPARING

Acacia baxteri ↔ **Acacia costata**

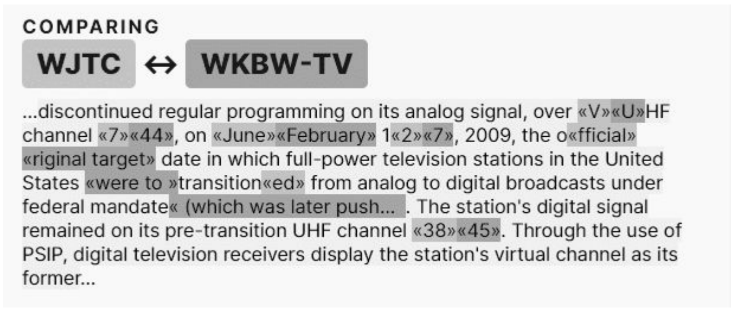
...is a shrub belonging to the genus Acacia and the subgenus Phyllodineae « that»«, and» is endemic to «the south west of »Western Australia. Description The «spreading and prickly»«compact and pungent» shrub typically grows to a height of 0.2 to «0.5»«1» metre«s» (1 to «2»«3» ft). «The ribbed branchlets ... «It blooms from Septem... and produces yellow flowers...

Fig. 2. An example state of the website showing the user selecting an article pair.

[7] https://tailwindui.com/components/marketing/sections/feature-sections

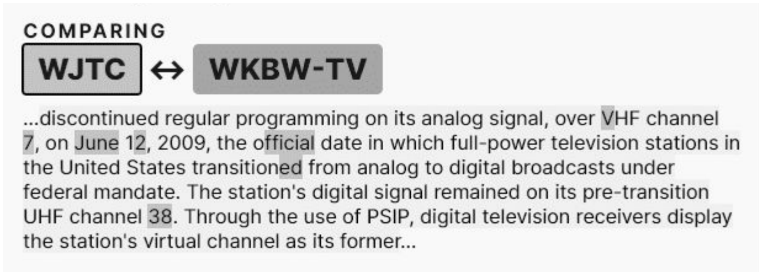
4.1 Success Cases

One of the objectives of the manual filtering stage was to find cases of text reuse where the faster, yet more generalized, *diff* algorithm would succeed in returning a human-readable comparison of the articles in question. We were able to – although in a condition of very restrictive time constraints – produce a list of a few articles which were deemed "success cases", that were subsequently added to the final JSON. The following is an example of one of those such cases.



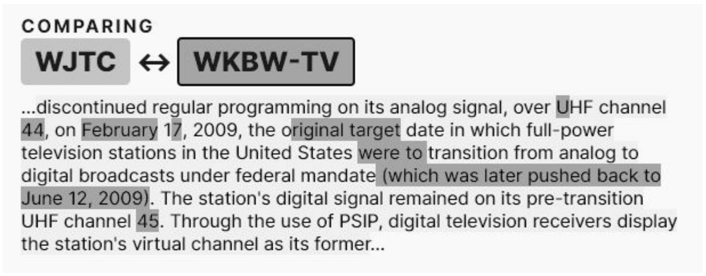
...discontinued regular programming on its analog signal, over «V»«U»HF channel «7»«44», on «June»«February» 1«2»«7», 2009, the o«fficial» «riginal target» date in which full-power television stations in the United States «were to »transition«ed» from analog to digital broadcasts under federal mandate« (which was later push... . The station's digital signal remained on its pre-transition UHF channel «38»«45». Through the use of PSIP, digital television receivers display the station's virtual channel as its former...

Fig. 3. Success case example: comparing two articles, "WJTC" and "WKBW-TV". Both relate to discontinued television channels, and contain a fair amount of reused text. However, some innate differences, such as the actual channel number, make it so the reused portions are not identical. In this example, the *diff* algorithm works very well by highlighting the modified words within the reused sentences.



...discontinued regular programming on its analog signal, over VHF channel 7, on June 12, 2009, the official date in which full-power television stations in the United States transitioned from analog to digital broadcasts under federal mandate. The station's digital signal remained on its pre-transition UHF channel 38. Through the use of PSIP, digital television receivers display the station's virtual channel as its former...

Fig. 4. Success case example: displaying the first article's retrieved text.



...discontinued regular programming on its analog signal, over UHF channel 44, on February 17, 2009, the original target date in which full-power television stations in the United States were to transition from analog to digital broadcasts under federal mandate (which was later pushed back to June 12, 2009). The station's digital signal remained on its pre-transition UHF channel 45. Through the use of PSIP, digital television receivers display the station's virtual channel as its former...

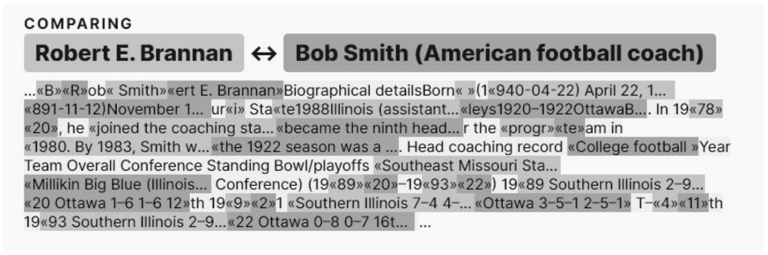
Fig. 5. Success case example: displaying the second article's retrieved text.

In figures 4 and 5, individual differences are shown for each article. The application colors text that is not reused in the paired article differently than it colors reused text. This way, differences between both articles become easily recognizable by the user, which was one of our main goals in developing the client-side.

It is interesting to note that, in the examples presented in figures 4 and 5, while the first article body states that the channel it relates to was discontinued on "June 12, 2009", the second article's text tells a more complete story: "February 17, 2009, the original target date [...] later pushed back to June 12, 2009". Immediately, our application's usefulness becomes apparent: it would likely be relevant, in an informational sense, to update the first article's content to deliver as much detail as the second one.

4.2 Failure Cases

However, not all is fun and games. As expected, the overgeneralization of the *diff* algorithm also causes failure cases, in which the result isn't ideal for human readability, or in which it fails to detect an instance of reused text. Furthermore, constant and frequent changes to Wikipedia also directly affect the application: article bodies are fetched in real time using Wikipedia's API, so any changes to the start and end characters will inevitably break the article trimming process. Similarly, if the reused text itself is removed, rewritten, or even moved to another section, the less robust *diff* algorithm will fail to detect it. In the *§ Discussion* section below, we evaluate alternatives, as well as solutions to the failure cases.

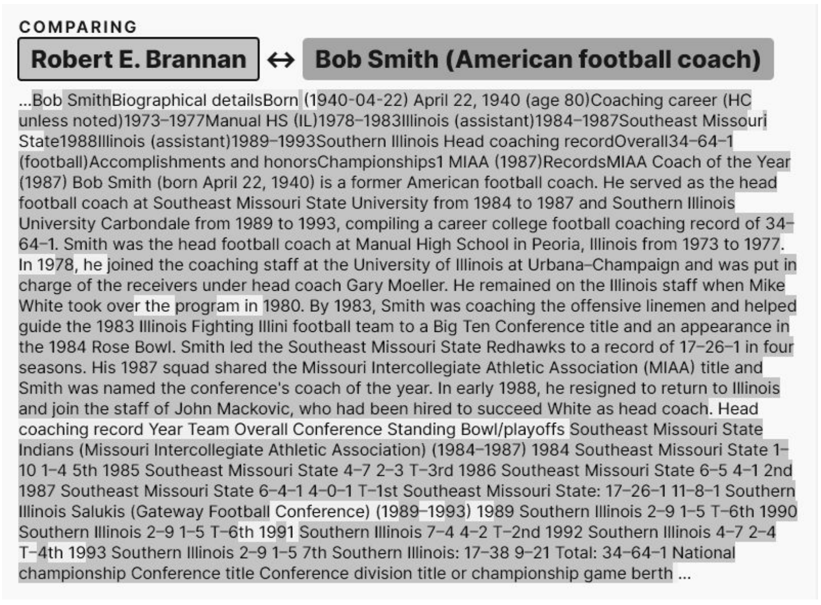


COMPARING

Robert E. Brannan ↔ Bob Smith (American football coach)

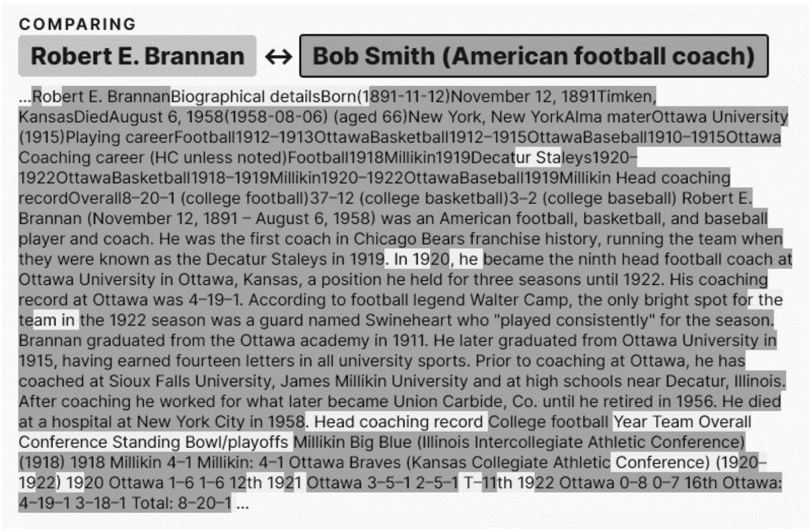
...«B»«R»ob« Smith»«ert E. Brannan»Biographical detailsBorn« »(1«940-04-22) April 22, 1...
«891-11-12)November 1... ur«i» Sta«te1988Illinois (assistant...«leys1920–1922OttawaB.... In 19«78»
«20», he «joined the coaching sta... «became the ninth head... r the «progr»«te»am in
«1980. By 1983, Smith w... «the 1922 season was a .... Head coaching record «College football »Year
Team Overall Conference Standing Bowl/playoffs «Southeast Missouri Sta...
«Millikin Big Blue (Illinois... Conference) (19«89»«20»–19«93»«22») 19«89 Southern Illinois 2–9...
«20 Ottawa 1–6 1–6 12»th 19«9»«2»1 «Southern Illinois 7–4 4–... «Ottawa 3–5–1 2–5–1» T–«4»«11»th
19«93 Southern Illinois 2–9...«22 Ottawa 0–8 0–7 16t... ...

Fig. 6. Failure case example: comparing two articles, "Robert E. Brannan" and "Bob Smith". Both relate to football coaches. However, reuse limiters were modified since the generation of the dataset, making the extraction method fail to only return the reused section. Because of this, both articles are being compared in their entirety, which, without the text-alignment stage of the previous work's pipeline, makes the *diff* algorithm only find scattered reuse cases, consisting of only words and letters.



...Bob SmithBiographical detailsBorn (1940-04-22) April 22, 1940 (age 80)Coaching career (HC unless noted)1973–1977Manual HS (IL)1978–1983Illinois (assistant)1984–1987Southeast Missouri State1988Illinois (assistant)1989–1993Southern Illinois Head coaching recordOverall34–64–1 (football)Accomplishments and honorsChampionships1 MIAA (1987)RecordsMIAA Coach of the Year (1987) Bob Smith (born April 22, 1940) is a former American football coach. He served as the head football coach at Southeast Missouri State University from 1984 to 1987 and Southern Illinois University Carbondale from 1989 to 1993, compiling a career college football coaching record of 34–64–1. Smith was the head football coach at Manual High School in Peoria, Illinois from 1973 to 1977. In 1978, he joined the coaching staff at the University of Illinois at Urbana–Champaign and was put in charge of the receivers under head coach Gary Moeller. He remained on the Illinois staff when Mike White took over the program in 1980. By 1983, Smith was coaching the offensive linemen and helped guide the 1983 Illinois Fighting Illini football team to a Big Ten Conference title and an appearance in the 1984 Rose Bowl. Smith led the Southeast Missouri State Redhawks to a record of 17–26–1 in four seasons. His 1987 squad shared the Missouri Intercollegiate Athletic Association (MIAA) title and Smith was named the conference's coach of the year. In early 1988, he resigned to return to Illinois and join the staff of John Mackovic, who had been hired to succeed White as head coach. Head coaching record Year Team Overall Conference Standing Bowl/playoffs Southeast Missouri State Indians (Missouri Intercollegiate Athletic Association) (1984–1987) 1984 Southeast Missouri State 1–10 1–4 5th 1985 Southeast Missouri State 4–7 2–3 T–3rd 1986 Southeast Missouri State 6–5 4–1 2nd 1987 Southeast Missouri State 6–4–1 4–0–1 T–1st Southeast Missouri State: 17–26–1 11–8–1 Southern Illinois Salukis (Gateway Football Conference) (1989–1993) 1989 Southern Illinois 2–9 1–5 T–6th 1990 Southern Illinois 2–9 1–5 T–6th 1991 Southern Illinois 7–4 4–2 T–2nd 1992 Southern Illinois 4–7 2–4 T–4th 1993 Southern Illinois 2–9 1–5 7th Southern Illinois: 17–38 9–21 Total: 34–64–1 National championship Conference title Conference division title or championship game berth ...

Fig. 7. Failure case example: displaying the first article's entire retrieved text.



...Robert E. BrannanBiographical detailsBorn(1891-11-12)November 12, 1891Timken, KansasDiedAugust 6, 1958(1958-08-06) (aged 66)New York, New YorkAlma materOttawa University (1915)Playing careerFootball1912–1913OttawaBasketball1912–1915OttawaBaseball1910–1915Ottawa Coaching career (HC unless noted)Football1918Millikin1919Decatur Staleys1920–1922OttawaBasketball1918–1919Millikin1920–1922OttawaBaseball1919Millikin Head coaching recordOverall8–20–1 (college football)37–12 (college basketball)3–2 (college baseball) Robert E. Brannan (November 12, 1891 – August 6, 1958) was an American football, basketball, and baseball player and coach. He was the first coach in Chicago Bears franchise history, running the team when they were known as the Decatur Staleys in 1919. In 1920, he became the ninth head football coach at Ottawa University in Ottawa, Kansas, a position he held for three seasons until 1922. His coaching record at Ottawa was 4–19–1. According to football legend Walter Camp, the only bright spot for the team in the 1922 season was a guard named Swineheart who "played consistently" for the season. Brannan graduated from the Ottawa academy in 1911. He later graduated from Ottawa University in 1915, having earned fourteen letters in all university sports. Prior to coaching at Ottawa, he has coached at Sioux Falls University, James Millikin University and at high schools near Decatur, Illinois. After coaching he worked for what later became Union Carbide, Co. until he retired in 1956. He died at a hospital at New York City in 1958. Head coaching record College football Year Team Overall Conference Standing Bowl/playoffs Millikin Big Blue (Illinois Intercollegiate Athletic Conference) (1918) 1918 Millikin 4–1 Millikin: 4–1 Ottawa Braves (Kansas Collegiate Athletic Conference) (1920–1922) 1920 Ottawa 1–6 1–6 12th 1921 Ottawa 3–5–1 2–5–1 T–11th 1922 Ottawa 0–8 0–7 16th Ottawa: 4–19–1 3–18–1 Total: 8–20–1 ...

Fig. 8. Failure case example: displaying the second article's entire retrieved text.

5 Discussion

Due to unavoidable limitations in development time and the computing power available, this work was intended to be simply a proof of concept / minimum viable product. Because of this, there will inherently be flaws and shortcomings in its performance. Nevertheless, we deem to have reached our goal of demonstrating that such an application is feasible, and hope to see more advances in this area in the future.

6 Conclusion

6.1 Expectations

This work presents a proof of concept implementation for an application to aid in visualizing reused text within Wikipedia articles, with potential to allow users to detect modification of such reused sections. Although not perfect, it serves as an example of what could be more robustly built given more time and computational power. We expose the project's shortcomings and failure cases, and propose below possible solutions to those issues, as well as some suggestions for future improvements.

We are nevertheless pleased with the results, as they fall within our – admittedly, somewhat low – expectations for this project.

6.2 Future Work

Some of the issues present in this project could be potentially resolved by adapting previous works' methods of reuse extraction, to become more lightweight and performatic, and thus more suitable for client side execution. A simpler implementation of the pipeline developed by Alshomary et al. [9], with a more fine-grained control over input and output of each stage – specifically to allow for individual article pairs to be fed into the text alignment stage – would likely be more effective and more suitable overall than our approach with the more general *diff* algorithm.

Additionally, in a fully implemented environment, the manual stage would require automatization. Some form of filtering would likely still be required or, at the very least, the reimplementation of the same or a similar form of the preprocessing used in the original pipeline for Wikipedia article extraction.

Lastly, an issue still to be thought about is regarding the searchability of indexed articles in the client side of the application. In our work, a subset comprising only a mere 10 articles (out of the JSONL dataset of over 80 GB) was delivered. Because of this, a simple selection box was more than enough to fulfill our goals. However, in an eventual full-sized implementation, different solutions would need to be thought of in order to provide the user with the ability to pick all – or at the very least, a sizable portion of – the reuse cases, but without overwhelming them with endless scrollable lists.

Acknowledgments. We would like to thank Dr. Giseli Lopes for the advice and feedback given for the duration of this work, as well as for offering the opportunity to develop this project as a part of the Information Retrieval class. We would also like to extend our thanks to the Federal University of Rio de Janeiro (UFRJ) for providing both the physical and the academic infrastructure for the Computer Science course.

References

1. Forlano, L.: WiFi Geographies: When Code Meets Place. The Information Society: An International Journal. 25:5, 344–352 (2009). doi:10.1080/01972240903213076
2. Akers, J., Bansal, G., Cadamuro, G., Chen, C., Chen, Q., et al.: Technology-Enabled Disinformation: Summary, Lessons, and Recommendations. arXiv preprint:1812.09383 (2018)
3. Aldwairi, M., Alwahedi, A.: Detecting Fake News in Social Media Networks. In: Procedia Computer Science. **141**, 215–222 (2018). doi:10.1016/j.procs.2018.10.171
4. Seo, J., Croft, W.: Local Text Reuse Detection. In: Proceedings of SIGIR, 571–578 (2008).
5. Chiu, S., Uysal, I., Croft, W.: Evaluating Text Reuse Discovery on the Web. In: Proceedings of the third symposium on Information interaction in context, IIiX '10. Association for Computing Machinery, New York, 299–304 August 2010. doi:10.1145/1840784.1840829
6. Weissman, S., Ayhan, S., Bradley, J., Lin, J.: Identifying Duplicate and Contradictory Information in Wikipedia. arXiv preprint:1406.1143 (2014)
7. Ardi, C., Heidemann, J.: Web-scale Content Reuse Detection (extended). (2014)
8. Stein, B., Eissen, S., Potthast, M.: Strategies for Retrieving Plagiarized Documents. In: Proceedings of SIGIR, 825–826 July 2007. doi:10.1145/1277741.1277928
9. Alshomary, M., Völske, M., Licht, T., Wachsmuth, H., Stein, B., Hagen, M., Potthast, M.: Wikipedia Text Reuse: Within and Without. In: ECIR 2019. arXiv preprint:1812.09221 (2018)
10. Myers, E.: An O(ND) Difference Algorithm and Its Variations. Algorithmica. **1**, 251–266 (1986). doi:10.1007/BF01840446