TL;DR

Anna Nixon, Elizabeth Powell, Amelia Rosenbaum, Nicolas Barra

Basic Project Information

TL;DR uses crowdsourcing to generate a summary of long and verbose text. News articles, especially ones from The New Yorker, are notoriously long and winding. In addition, dense, lengthy language makes it hard for people to keep up with the news when they don't have a lot of time. By using the crowd to filter out superfluous sentences or highlighting important ones, a summary is generated that includes only information most people would find important. It also reduces s the complexity and length of the article thus allowing users to understand its content without having to read the entire thing. Further, we hope that this same technique could be extended in the future to be able to provide summaries for additional kinds of text, such as terms and conditions documents, privacy statements, academic papers, research studies, books, and any other long-form text.

There are many news summary services that provide short, concise summaries of news articles every day. TheSkimm, is one such website that sends daily news article summaries to subscribers. They also have an app that provides the same service for mobile devices. Beyond those specific to news, there are numerous online summary generators that allow users to put in text and the service automatically generates a summary of that text. We are using one of these websites, to compare our summaries to in order to assess their quality. Another online text summarizer takes in a URL or the text that needs to be summarized and outputs a summary containing the number of sentences the user requests. This allows us to compare the quality of the summaries we generate with the quality of summaries existing tools create.

TL;DR is a business idea which uses crowdsourcing. It falls somewhere between engineering a complex system and conducting in-depth data analysis. First, we gather a news articles from The New Yorker. (Note that if we were to implement our project in its most complete form as a business idea there, would be a website that offers users the ability to create human-generated summaries of text of their choice. We would not be gathering the articles ourselves; rather users would input the URL to the text that they want summarized and we would pull the text from there.) Then, we break each document into sections of 3 paragraphs each, and post a round of Human Intelligence Tasks (HITs) on Mechanical Turk where Turkers must select the sentences that are important in these sections and another round of HITs where Turkers must delete the sentences that are not important in these sections. Next, we aggregate the data and generate two summaries, one from the HITs where Turkers selected important sentences and one from the HITs where Turkers deleted non-important sentences. Finally, we will post another round of HITs where Turkers are presented with the two Turker-generated summaries and the summary created by the online text summarizer. They are tasked with choosing the most coherent and relevant summary of the three. Then we tally the number of votes each summary gets and determine how the quality of our summaries compared to that of summaries generated by existing tools.

Video Link: https://vimeo.com/334922005
Password: NETS213

The Crowd

The members of our crowd will be Mechanical Turkers. Because this task requires substantial reading in English, we will give this task to Turkers in the United States. We used a real crowd for our final project. We posted HITs on Mechanical Turk and offered a reward for completing each HIT in which Turkers select or delete sentences. 10 unique Turkers looked at each section of each article, and approximately 50 Turkers in total gave us input.

Incentives

To incentivize the crowd to participate, we offer $0.03 for completing each HIT that contains a section of an article from which Turkers must select or delete sentences. This is an effective incentive since reading these articles doesn’t require any specialized skills and the task is not especially time consuming. Turkers will complete the HIT to get the reasonable amount of money that is rewarded to complete the tasks.

To incentivize the crowd to complete HITs where they need to select the best summary, we offer a $0.05 reward for each HIT. This is a high reward considering it requires no technical skills and takes very little time. We decided on this amount for reward since the completion of these HITs is crucial to determining the quality of the summaries we generated.

What the Crowd Gives Us

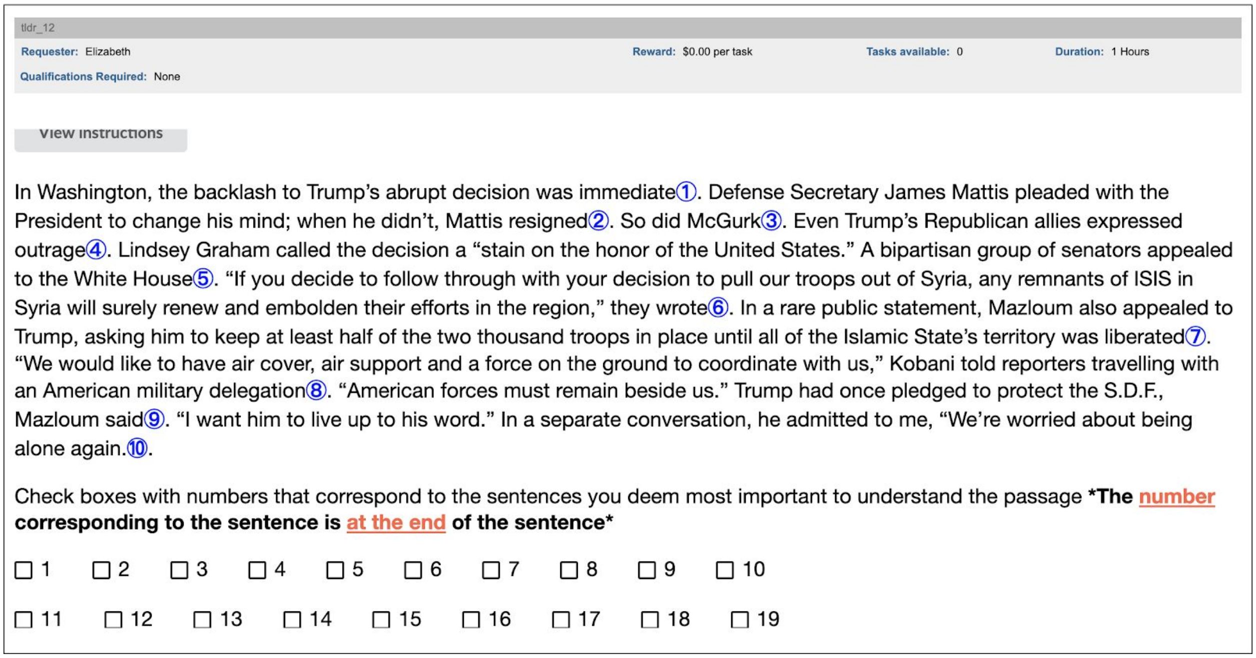
The crowd provides input on which sentences within a passage are most important or least important. In one HIT, crowd workers click on the sentences within a section of text which are most relevant to the main idea of the passage. The workers identify which sentences are most important for someone to read if they want to understand the central argument but do not have time to read the entire article. In another HIT, crowd workers click on the sentences within a section of text which are least relevant to the main idea of the passage. The workers identify which sentences are least important to understanding the central argument and can be skipped if someone does not have time to read the entire article. In addition, the crowd also helps us evaluate the quality of the summaries by voting for the best summary.

Automatic summarizers do exist, and we wanted to test whether a crowdsourced summary is better than an automated summary. We presented crowdworkers with three summaries, one generated by an auto-summarization tool and two generated using our crowdsourced method. In 73 % of occurrences, Turkers preferred a crowdsourced summary to the automated summary.

It is possible that at some point in the future, auto-summarizers will be just as good as or better than the crowd at creating a summary. However, for now, crowdsourcing outperformed the automated process. Additionally, any summarization algorithm would likely have to be trained with crowdsourced data. We did not train a machine learning component, but our data could certainly be a good input for training a machine learning algorithm.

We did create a user interface for crowd workers. We created HITs on Amazon Mechanical Turk for users to either select the most important sentences or delete the least important sentences within passages of text from articles. Originally, users selected a checkbox corresponding to a number printed at the end of each sentence. However, we updated the HIT so workers could directly click on sentences which they identified as important or unimportant, depending on the HIT. (Unfortunately, all of our HITs have expired so we cannot include a URL to the HIT, but all of the code used to format our HITs is on the TL;DR github repository at https://github.com/nicolasbarra/tldr/tree/master/src/hit_design).

Our HITs on Mechanical Turk evolved significantly from the start of our project to the end. Originally, we only had one type of HIT, where users identified the most important sentences in a section of text. We now have two types of HIT, one where workers select important sentences to keep, and one where workers select unimportant sentences to delete. Within each HIT, workers are only asked to select important OR unimportant sentences, not both.



tldr_12

Requester: Elizabeth **Reward:** $0.00 per task **Tasks available:** 0 **Duration:** 1 Hours

Qualifications Required: None

In Washington, the backlash to Trump's abrupt decision was immediate①. Defense Secretary James Mattis pleaded with the President to change his mind; when he didn't, Mattis resigned②. So did McGurk③. Even Trump's Republican allies expressed outrage④. Lindsey Graham called the decision a "stain on the honor of the United States." A bipartisan group of senators appealed to the White House⑤. "If you decide to follow through with your decision to pull our troops out of Syria, any remnants of ISIS in Syria will surely renew and embolden their efforts in the region," they wrote⑥. In a rare public statement, Mazloum also appealed to Trump, asking him to keep at least half of the two thousand troops in place until all of the Islamic State's territory was liberated⑦. "We would like to have air cover, air support and a force on the ground to coordinate with us," Kobani told reporters travelling with an American military delegation⑧. "American forces must remain beside us." Trump had once pledged to protect the S.D.F., Mazloum said⑨. "I want him to live up to his word." In a separate conversation, he admitted to me, "We're worried about being alone again.⑩.

Check boxes with numbers that correspond to the sentences you deem most important to understand the passage ***The number corresponding to the sentence is at the end of the sentence***

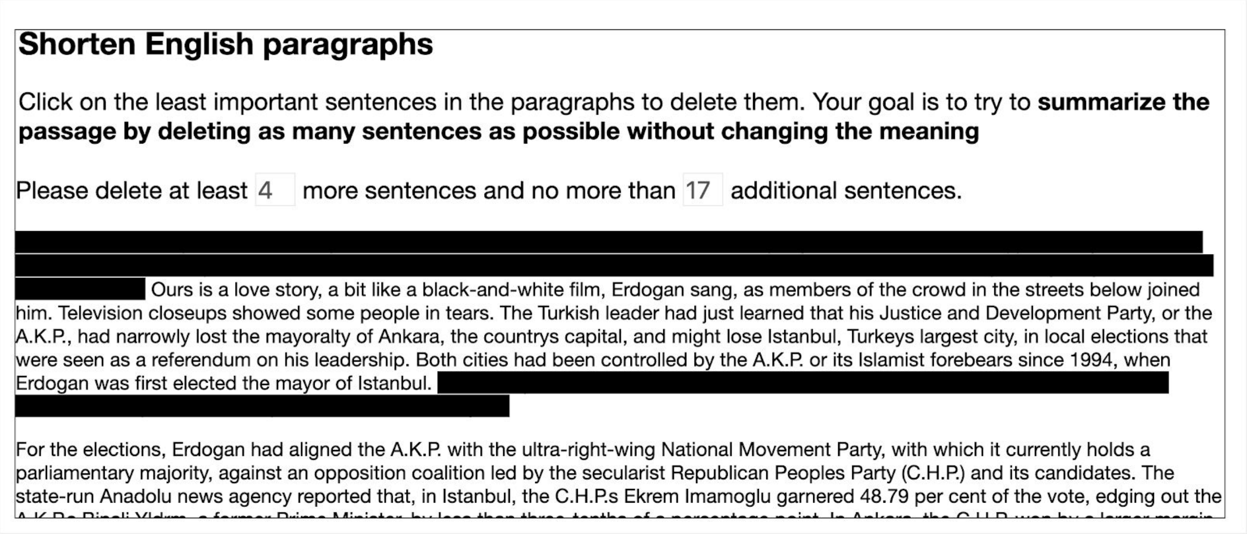
□ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □ 8 □ 9 □ 10

□ 11 □ 12 □ 13 □ 14 □ 15 □ 16 □ 17 □ 18 □ 19

Caption: Original HIT design https://github.com/nicolasbarra/tldr/blob/master/docs/originalhit.png

At the beginning of the project, we inserted circled numbers, for example, ①, at the end of each sentence in the section of text presented to the Turker. At the bottom of the page, there were checkboxes where the worker had to click if they wanted to mark a sentence as important or unimportant, depending on the HIT. All of the HITs had 28 checkboxes at the end of the page, but not all of the sections of text had 28 sentences, and feedback from classmates helped us realize this was confusing.

As a result, we modified our HIT so workers could directly click on sentences to select them. In the HIT for selecting important sentences, when a worker clicked a sentence, it became highlighted in green. In the HIT for deleting unimportant sentences, when a worker clicked a sentence, it became highlighted in black, essentially redacted or deleted. If the worker changed their mind or made a mistake, they could click the sentence again to un-select it. Clicking on any part of the sentence highlighted or un-highlighted the whole sentence.



Shorten English paragraphs

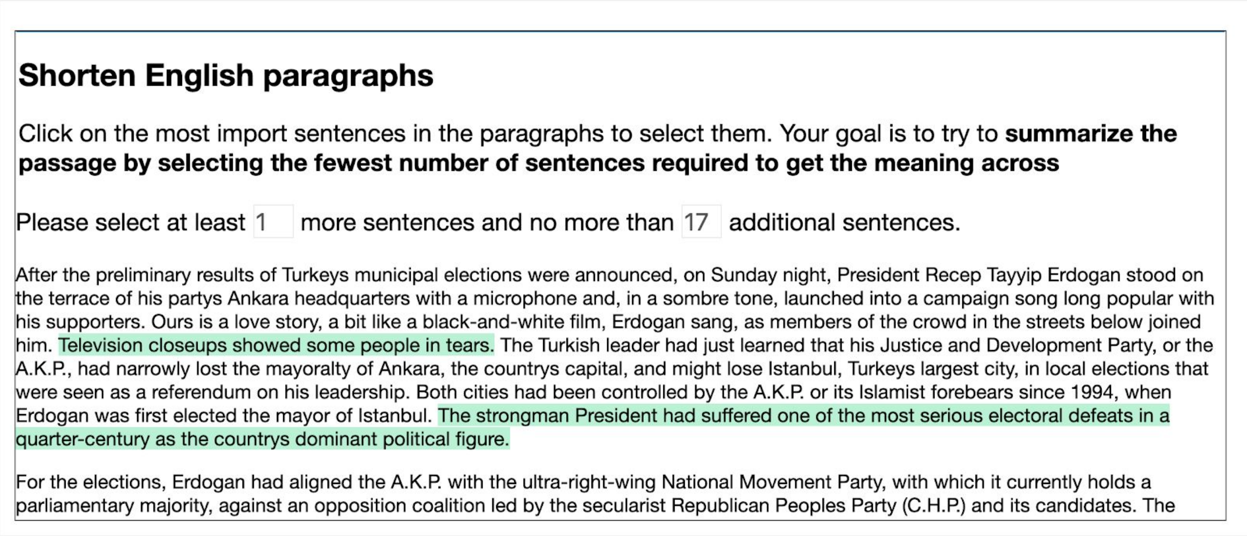
Click on the least important sentences in the paragraphs to delete them. Your goal is to try to **summarize the passage by deleting as many sentences as possible without changing the meaning**

Please delete at least 4 more sentences and no more than 17 additional sentences.

[redacted] Ours is a love story, a bit like a black-and-white film, Erdogan sang, as members of the crowd in the streets below joined him. Television closeups showed some people in tears. The Turkish leader had just learned that his Justice and Development Party, or the A.K.P., had narrowly lost the mayoralty of Ankara, the countrys capital, and might lose Istanbul, Turkeys largest city, in local elections that were seen as a referendum on his leadership. Both cities had been controlled by the A.K.P. or its Islamist forebears since 1994, when Erdogan was first elected the mayor of Istanbul. [redacted]

For the elections, Erdogan had aligned the A.K.P. with the ultra-right-wing National Movement Party, with which it currently holds a parliamentary majority, against an opposition coalition led by the secularist Republican Peoples Party (C.H.P.) and its candidates. The state-run Anadolu news agency reported that, in Istanbul, the C.H.P.s Ekrem Imamoglu garnered 48.79 per cent of the vote, edging out the

Caption: HIT where workers deleted unimportant sentences
https://github.com/nicolasbarra/tldr/blob/master/docs/deletionhit.png



Shorten English paragraphs

Click on the most import sentences in the paragraphs to select them. Your goal is to try to **summarize the passage by selecting the fewest number of sentences required to get the meaning across**

Please select at least 1 more sentences and no more than 17 additional sentences.

After the preliminary results of Turkeys municipal elections were announced, on Sunday night, President Recep Tayyip Erdogan stood on the terrace of his partys Ankara headquarters with a microphone and, in a sombre tone, launched into a campaign song long popular with his supporters. Ours is a love story, a bit like a black-and-white film, Erdogan sang, as members of the crowd in the streets below joined him. Television closeups showed some people in tears. The Turkish leader had just learned that his Justice and Development Party, or the A.K.P., had narrowly lost the mayoralty of Ankara, the countrys capital, and might lose Istanbul, Turkeys largest city, in local elections that were seen as a referendum on his leadership. Both cities had been controlled by the A.K.P. or its Islamist forebears since 1994, when Erdogan was first elected the mayor of Istanbul. The strongman President had suffered one of the most serious electoral defeats in a quarter-century as the countrys dominant political figure.

For the elections, Erdogan had aligned the A.K.P. with the ultra-right-wing National Movement Party, with which it currently holds a parliamentary majority, against an opposition coalition led by the secularist Republican Peoples Party (C.H.P.) and its candidates. The

Caption: HIT where workers selected important sentences
https://github.com/nicolasbarra/tldr/blob/master/docs/selectionhit.png

At the top of the page, there were instructions covering how to complete the HIT, as well as realtime instructions for how many more sentences the worker had to select at a minimum, and how many more sentences he/she was allowed to select at a maximum. These numbers were a function of the length of the article. The worker always had to select at least 10% of the sentences in the article, and at most 75% of the sentences in the article (rounded to the nearest integer). For example, in a passage with 20 sentences, the instructions initially said the worker had to select at least 2 more sentences, and at most 15 more. After the worker selected 1 sentence, the instructions updated to say that the worker had to select at least 1 more sentence and at most 14 more sentences.

Skills

Our crowd workers must be literate in English at a high school level at the least, as the passages we are asking them to read often contain complex concepts and difficult vocabulary words. We intentionally selected articles which are long and difficult to comprehend, because generating a high quality summary of a long, dense article is far more relevant than generating a summary of a short, simple article. We chose to summarize articles from the *New Yorker*, which is known to have verbose content. Here is one of the articles we asked the workers to summarize.

The English literacy skills of workers on Amazon Mechanical Turk vary widely, as many Turkers live in countries where English is not the primary language. For this reason, we only presented the task to Turkers in the United States. Even within the United States, literacy is not consistent. We considered only presenting the task to Turkers who have a high school degree, but we decided not to add this constraint as it made the project significantly more expensive, and we hypothesized that Turkers might lie about their education level anyway to become eligible for more tasks.

Education level, language spoken at home, and country of origin can impact English literacy skills. Even among people who were raised in the United States and speak English at home, disparities within the American education system can lead to people with similar backgrounds and education levels to having drastically different literacy levels. Additionally, even with strong English literacy skills, if a Turker was absent-minded, distracted, multitasking, or simply not paying close attention while completing our HIT, he/she might not have successfully chosen the most important sentences. He/she might have instead opted for long sentences, introductory sentences, or concluding sentences, and missed a short but vital sentence sandwiched in the middle of a dense paragraph.

We considered first giving Turkers a reading comprehension test, and then presenting our summarization HITs to workers who performed well on the test. However, our initial analysis of the results from the Turkers indicated that a reading comprehension test was unnecessary because the summary we generated from the input of the Turkers was just as good as the summary we generated from the input of our classmates, who are obviously highly literate.

Quality Control

We were concerned with the quality of the responses of the crowd because it was vital that the Turkers actually select important sentences which were not tangential to the main idea, or overly specific. However, because the passage of text presented was short, and the users simply had to select sentences rather than giving any creative input, we were hopeful that crowd workers would not find it too difficult to provide high quality responses.

We used a majority vote method for ensuring quality of the responses from the Turkers. We presented each passage of text to 10 Turkers, so any sentence in the article could have anywhere from 0 to 10 votes (indicating it was important or unimportant). The quality control algorithm took in the results from the HIT and then counted the number of votes every unique sentence received, as well as the maximum number of times a single sentence was voted for in an article. The maximum number of times a single sentence was voted for represented the maximum agreement on that article and the votes for each sentence in that article were scaled based on that max value. These values were outputted to be used by the aggregation module.

We also decided, after our data had been collected, to post a HIT to gauge how well the Turkers pay attention to the task of selecting important sentences. We created a HIT identical to our selection HIT, but added a sentence in the middle which said, “select this sentence if you are paying attention.” Interestingly, only half of Turkers selected the sentence. The fact that the worker-generated summaries were preferred to the online-generated summaries, even though roughly half of users were not paying close attention, is an indication of the power of our quality control mechanism of aggregating votes. In the future, we could likely create an even better summary by only using those people who we knew were paying attention while selecting sentences. While this would likely produce an even better summary, it would also likely require more HITs to achieve the same number of quality responses and, thus, cost more money

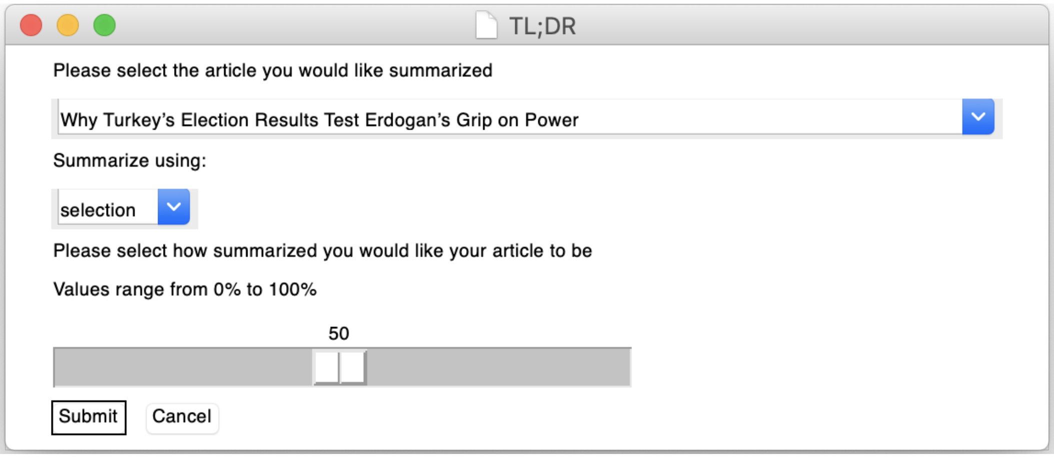
Since our quality control module and our aggregation module were closely coupled we analyzed their success together, which is explained below in the section entitled “Quality Control and Aggregation Analysis”.

Aggregation

Our aggregation methodology was based around the idea that increasing the threshold of agreement for what was an ‘important’ sentence would result in more concise summaries when looking at data where Turkers selected sentences. Conversely, by decreasing the threshold of agreement for what was a ‘useless’ sentence would result in more concise summaries. If multiple workers selected a sentence as important it would remain in the output summary even as the threshold was raised, while a sentence that only one person selected as important would be cut if the user wanted shorter summaries but kept if the user wanted more information.

Although we did not do extensive analysis on the quality of individual summaries compared to the aggregated summaries, we did sample a couple of individual responses and found that the summaries either missed many important pieces in the case of the selection HIT or Turkers did not delete enough content for the result to truly be a summary in the case of the deletion HIT.

The aggregation module allows users to select an article title, a summarization method (either selection or deletion), and an agreement threshold. The program then filters for only sentences that have at least that percentage of users agreeing that the sentence is important or at least that many users who do not declare it as unimportant (depending on the summarization method selected). The sentences are ordered using a dictionary that maps sentences to their original order in the article and that result is returned to the user.



TL;DR

Please select the article you would like summarized

Why Turkey's Election Results Test Erdogan's Grip on Power

Summarize using:

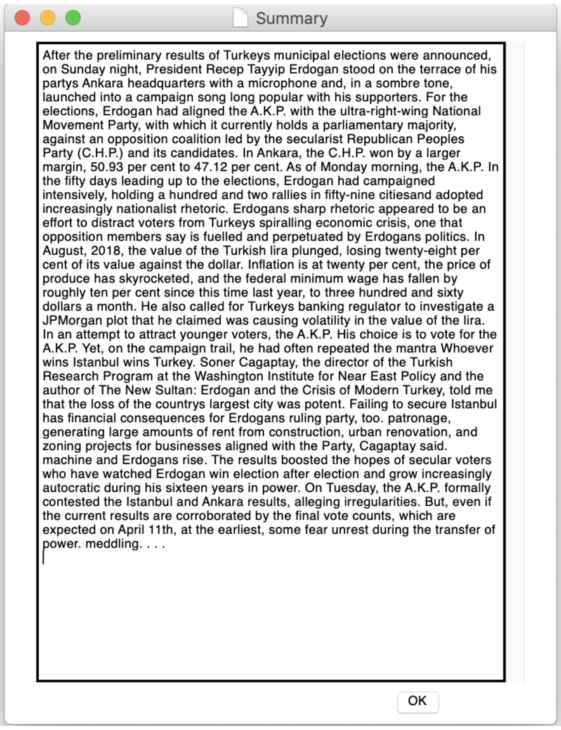
selection

Please select how summarized you would like your article to be

Values range from 0% to 100%

50

Image 1: The figure above shows the user interface containing drop downs for article and summarization type as well as a slider for agreement threshold https://github.com/nicolasbarra/tldr/blob/master/docs/gui.png

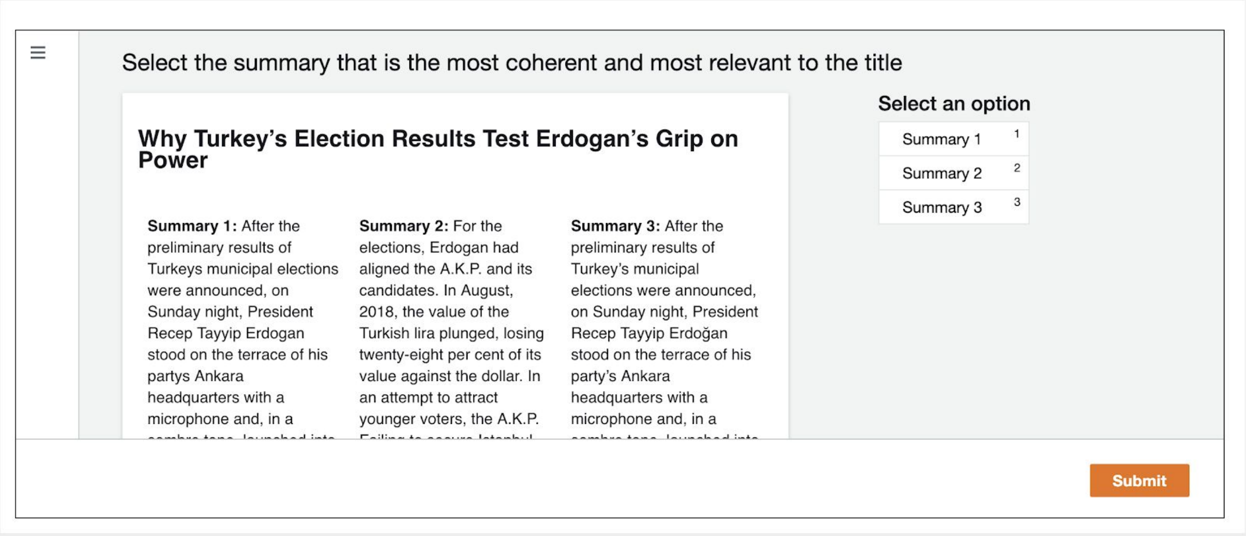


After the preliminary results of Turkeys municipal elections were announced, on Sunday night, President Recep Tayyip Erdogan stood on the terrace of his partys Ankara headquarters with a microphone and, in a sombre tone, launched into a campaign song long popular with his supporters. For the elections, Erdogan had aligned the A.K.P. with the ultra-right-wing National Movement Party, with which it currently holds a parliamentary majority, against an opposition coalition led by the secularist Republican Peoples Party (C.H.P.) and its candidates. In Ankara, the C.H.P. won by a larger margin, 50.93 per cent to 47.12 per cent. As of Monday morning, the A.K.P. In the fifty days leading up to the elections, Erdogan had campaigned intensively, holding a hundred and two rallies in fifty-nine citiesand adopted increasingly nationalist rhetoric. Erdogans sharp rhetoric appeared to be an effort to distract voters from Turkeys spiralling economic crisis, one that opposition members say is fuelled and perpetuated by Erdogans politics. In August, 2018, the value of the Turkish lira plunged, losing twenty-eight per cent of its value against the dollar. Inflation is at twenty per cent, the price of produce has skyrocketed, and the federal minimum wage has fallen by roughly ten per cent since this time last year, to three hundred and sixty dollars a month. He also called for Turkeys banking regulator to investigate a JPMorgan plot that he claimed was causing volatility in the value of the lira. In an attempt to attract younger voters, the A.K.P. His choice is to vote for the A.K.P. Yet, on the campaign trail, he had often repeated the mantra Whoever wins Istanbul wins Turkey. Soner Cagaptay, the director of the Turkish Research Program at the Washington Institute for Near East Policy and the author of The New Sultan: Erdogan and the Crisis of Modern Turkey, told me that the loss of the countrys largest city was potent. Failing to secure Istanbul has financial consequences for Erdogans ruling party, too. patronage, generating large amounts of rent from construction, urban renovation, and zoning projects for businesses aligned with the Party, Cagaptay said. machine and Erdogans rise. The results boosted the hopes of secular voters who have watched Erdogan win election after election and grow increasingly autocratic during his sixteen years in power. On Tuesday, the A.K.P. formally contested the Istanbul and Ankara results, alleging irregularities. But, even if the current results are corroborated by the final vote counts, which are expected on April 11th, at the earliest, some fear unrest during the transfer of power. meddling. . . .

Image 2: The figure above shows how the resulting summary is displayed to users in a scrollable window. https://github.com/nicolasbarra/tldr/blob/master/docs/output.png

Quality Control and Aggregation Analysis

We subjectively compared the quality of the summary generated from the selection HIT, the deletion HIT, and the auto-summarizer by reading each summary after reading the article they summarized. We also created a HIT on Mechanical Turk which asked workers to select the best summary out of three summaries, one generated using each of the above strategies.



Select the summary that is the most coherent and most relevant to the title

Why Turkey's Election Results Test Erdogan's Grip on Power

Summary 1: After the preliminary results of Turkeys municipal elections were announced, on Sunday night, President Recep Tayyip Erdogan stood on the terrace of his partys Ankara headquarters with a microphone and, in a sombre tone, launched into

Summary 2: For the elections, Erdogan had aligned the A.K.P. and its candidates. In August, 2018, the value of the Turkish lira plunged, losing twenty-eight per cent of its value against the dollar. In an attempt to attract younger voters, the A.K.P. Failing to secure Istanbul

Summary 3: After the preliminary results of Turkey's municipal elections were announced, on Sunday night, President Recep Tayyip Erdoğan stood on the terrace of his party's Ankara headquarters with a microphone and, in a sombre tone, launched into

Caption: HIT created for Turkers to choose their preferred summary
https://github.com/nicolasbarra/tldr/blob/master/docs/analysishit.png

Because we are all literate in English and had read the original article, we were able to do an initial subjective evaluation of the quality of each summary. We concluded that the deletion summary was the best summary. Although it was the longest by far, as setting an extremely high threshold still yielded a long summary, it was also the most fluid and informative. Relative to the length and complexity of the initial article, we thought the deletion article was the best compromise between brevity and relevance.

Crowdsourcing the analysis led to similar but not identical results. In the HIT on Mechanical Turk, workers were presented with three summaries of the same length for the same article. One summary was generated from the HIT where Turkers selected important sentences, one summary was generated from the HIT where Turkers deleted unimportant sentences, and one summary was generated by an auto-summarizer. We presented the summaries in a vertical column order and asked each Turker to select the most relevant, cohesive summary. Although they had not read the original article, the purpose of a summary is to prevent someone from having to read the entirety of a long text, so we thought this type of quality analysis was relevant to the real-world application of our project. We then counted the number of votes each summary received to evaluate the relative quality of each.

We investigated whether asking Turkers to select important sentences or asking Turkers to delete unimportant sentences created a more relevant, cohesive summary. We included the auto-summary as a point of comparison as well to evaluate the quality of crowdsourced summaries compared to computer-generated summaries. After tallying the votes for each summary, the crowdsourced summaries were strongly preferred to the autosummary, with 73% of the votes . The Turkers also slightly preferred the selection summary to the deletion summary with the selection summary winning 40% of votes and the deletion summary winning 33% of votes. Typically, in evaluating the quality of summarization techniques, people prefer longer summaries because they convey more information. Interestingly, in our analysis, people preferred the shorter selection summary to both the slightly longer autosummary and the longer deletion summary, but the most significant divide was between the crowdsourced and automated summaries, with crowdsourcing far outpreforming automation.

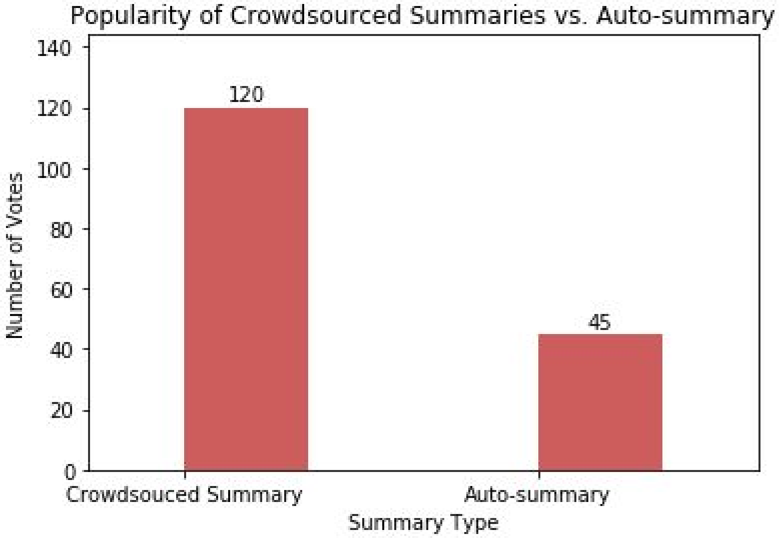


Chart URL: https://github.com/nicolasbarra/tldr/blob/master/docs/AnalysisResults2.png

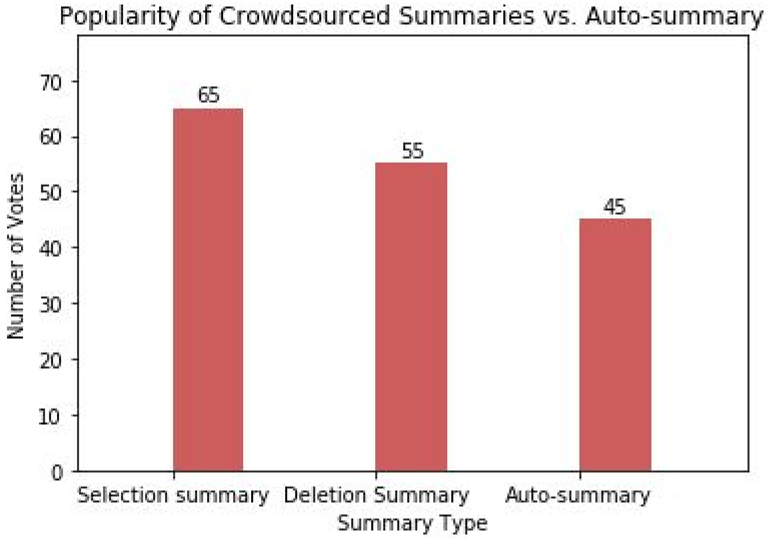


Chart URL: https://github.com/nicolasbarra/tldr/blob/master/docs/FinalResults.png

Scaling Up

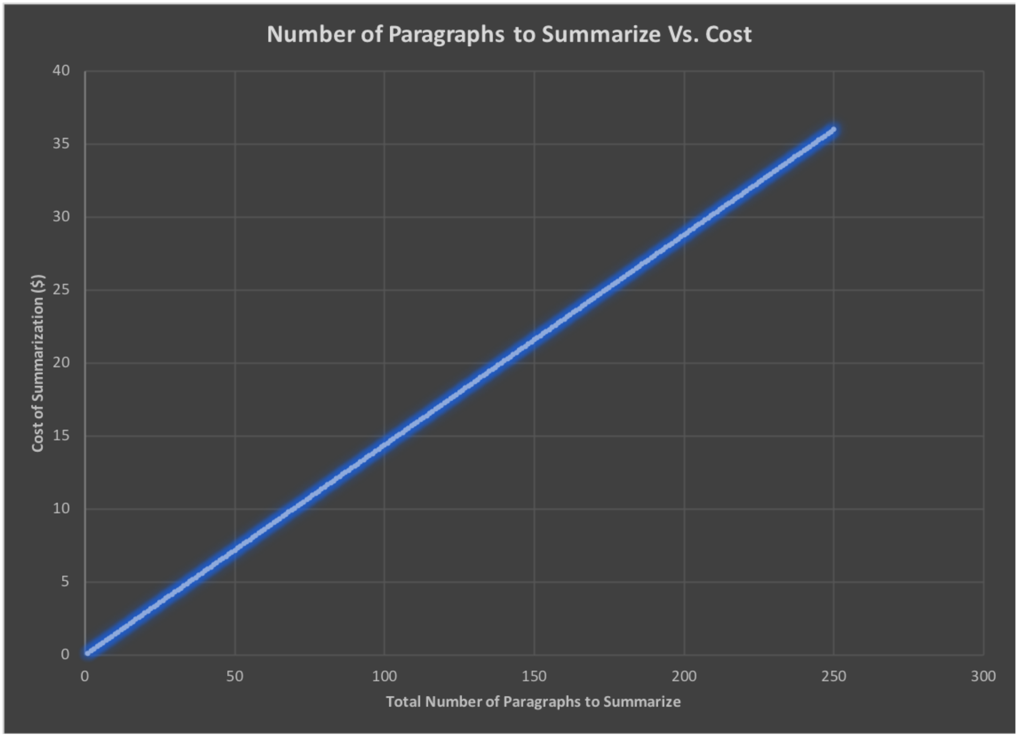
Given the broad application of a summarizing tool, our project could be used in a variety of different concepts and by different people. For example, while we used this project to create summaries of news articles, students could use this same idea to create summaries of academic papers they are tasked with reading. Users of services with long privacy policies could use TL;DR to make those policies more manageable to read and understand. We could also create a service where subscribers pay a small fee to receive a summarized version of all front page stories of the New York Times or a different newspaper. Thus, the problem of needing to take text in long form and generate a useful and accurate summary that allows the reader to get as much information as possible in a small amount of time is one that really can apply to anyone.

While our project might not necessarily need the input of thousands of people in order to generate a summary that is reasonably useful, it is likely that the more people that contribute, the stronger the summary will be. We would expect the aggregation of a large number of contributions would mean that the summary that is created would reflect the combined perspectives of a wide group of people and therefore create the most effective summary. Based on the results we saw, it is likely that we could create effective summaries with the contributions of only around 100-200 people, depending on the length of the article. The reason that we would need a much larger number of contributions would be so that we could quickly fulfill requests for a large number of articles simultaneously thus making this a scalable project to a wide audience.

The primary challenge of scaling to a large crowd would be with the ballooning cost of the project. If we required a large number of people to review selected articles that were significantly longer in length, the increased number of HITs that this process would require might make the project cost prohibitive, and thus not useful to users. In theory, the website that provided the service would use the revenue generated from the ads to finance the cost of posting the HITs. Thus, if we required a large crowd to review each article or tried to apply our project to massive terms of services documents, the fact that we have to break the articles up into chunks where there are a number of HITs posted for each chunk might mean that the project would no longer be financially sound to organize as a website accessible to the public. That said, because our aggregation process and quality control process rely only on the aggregate votes of the users, we would not expect to see any challenges introduced on the end of physically constructing the summary if we had thousands more inputs, as we would still be able to aggregate them using a majority vote to create our summary.

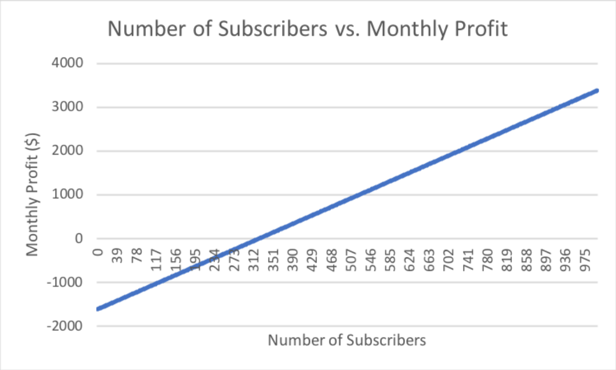
Currently, it costs us 14.4¢ per paragraph to create a summary of an article. We pay Turkers 3 cents per HIT, and 3 paragraphs are contained within each HIT, so we are paying 1 cent per paragraph. 10 Turkers complete each HIT, so 10 Turkers see each paragraph.

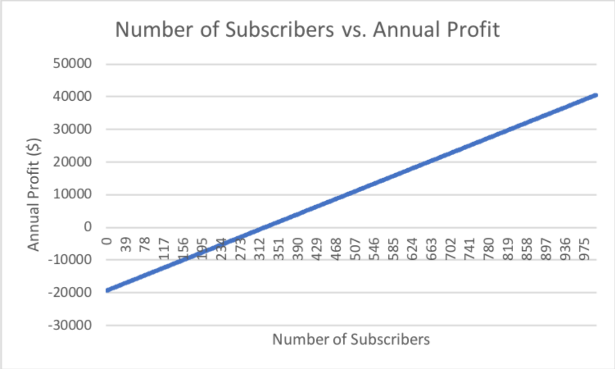
Additionally, Mechanical Turk charges 20% fee on the reward for the workers, and an additional 20% because each of our HIT batches contains more than 10 HITs. This comes out to 01¢ * 10 * 1.2 * 1.2 = 14.4¢ per paragraph. We created a graph to determine the cost of scaling this project. A long research paper or news article is approximately 25-30 paragraphs, which would cost around $4.00 to summarize.



Graph URL: https://github.com/nicolasbarra/tldr/blob/master/docs/CostAnalysis.png

We could also make TL;DR into a news compression service. If we wanted to summarize everything on the front page of the New York Times every day, it would probably cost about $54.00 dollars per day, as the average New York Times front page story has about 30 paragraphs, and there are about 9-12 stories on the front page. This comes to $1,620.00 per month. If we charged subscribers $5.00 per month, we would need 324 subscribers to break even.





Monthly Graph URL: https://github.com/nicolasbarra/tldr/blob/master/docs/MonthlyProfit.png
Annual Graph URL: https://github.com/nicolasbarra/tldr/blob/master/docs/AnnualProfit.png

We considered methods of decreasing the cost or summarization per paragraph in scaling up TL;DR, but we were unable to come up with a reasonable solution. The only way to decrease cost would be to pay workers less, which we were not willing to do, or pay fewer fees to Mechanical Turk. The only way to pay fewer fees to Mechanical Turk would be limiting our batches to only include 10 HITs. To limit our batch size, we would either have to make each section of each article its own HIT, which is a labor intensive and inefficient HIT design, or drastically increase the number of paragraphs per HIT. Because at least 10 Turkers must see each paragraph for quality control and aggregation purposes, and we cannot put more than 10 HITs in a batch without incurring the additional fee, each Turker would have to read and select important sentences in the entire article. Lengthy, dense articles would lead to bored Turkers and low quality work. Therefore, we concluded that we could not make TL;DR more cost effective if we scaled it up, however, the current costs are pretty minimal and reasonable relative to its purpose and use.

Project Analysis

Our project was successful in constructing summaries of articles that were selected to be more informative than the automatically-generated summaries from an online summarizer. We evaluated the efficacy of our summary by having workers on Mechanical Turk comparing the summaries that were created by aggregating the results of the HITs to a summary that was generated through an online summarizer tool. To do this comparison, we created an HIT on MTurk where workers were presented with the three summaries and asked to select the summary that they preferred. Specifically, the workers were asked to "select the summary that is more coherent and relevant to the title."

By far, the biggest challenges that we had to deal with were the technical challenges that we address below. One of these challenges that we were not able to overcome was an issue with the way we parsed sentences. As we will describe in more detail later on, we used the Natural Language Toolkit (NLTK) Library in Python to parse the articles into sentences. While this method worked fairly well, it did not work perfectly in that it resulted in some sentences being truncated too early or too late. Thus, when users were selecting sentences, they occasionally only selected one part of the sentence and so the summary that was constructed occasionally contained sentence fragments that were not completely sensical without the context provided by the rest of the sentence. This presented a challenge to the mechanism and quality of some of our summarizations as the automatic summerizers did a better job of not creating this kind of choppiness even if the overall summary they created was voted to be of lower quality. However, this issue only represents one small challenge that we had to deal with in this project. The majority of these challenges were technical, and, thus, we discuss them in greater detail below when we discuss the overall technical challenges that we faced with this project.

There were a number of different iterations between our original proposal and our final project. Initially, our plan was to use the crowd to generate a summary of long and tedious terms and conditions documents and compare these summaries to existing ones. Our motivation was that terms and conditions documents are incredibly dense thereby obscuring any relevant information. After spending some time looking at terms and conditions documents, we realized we were more interested in using this idea to synthesize news articles from The New Yorker. This was a topic that was of more personal interest to us given that we shared an interest in politics and had each personally experienced the difficulty of trying to quickly get a sense for what a New Yorker article was saying without having to read the article in its lengthy entirety.

Initially, we planned to first gather a large and diverse set of terms and conditions documents and break each into portions where we could then post HITs on Mechanical Turk and have multiple Turkers summarize those chunks of text. We then planned to post another round of HITs where Turkers rank the summaries written and used the best to train an algorithm to suggest summaries. We then hoped to post a HIT that allowed Turkers to vote between the summary our algorithm generated and other summaries. While we began with a similar method, in speaking with Professor Callison-Burch, we realized that the idea of trying to train a machine learning algorithm to generate summaries would require a significant amount of effort and more time than we had to complete this project. Thus, we decided not to pursue that aspect of the project, but we retained the component where we used workers to compare the summaries.

The primary limitation of our product is a timing issue. If we were to implement it as a website where users input the URL of an article that they wish to have summarized and then the website returns to them a Turker-generated summary of that article, there would likely be a considerable time delay between the initial input and the summary being returned to the user.

This would be especially true for longer articles. Thus, in order for this mechanism be an effective strategy, it would either have to require that there always be a large number of workers present and ready who can complete the tasks very rapidly necessitate that users input a URL and their email address in order for the completed summary to be sent to them by email once all the requisite HITs have been finished by the Turkers. While this mechanism might work, it might render our system less competitive than other automatic online summary generators, as they work instantaneously while ours would have a time delay.

Additionally, it is also possible that the cost of the HITs that would be required to generate the summary of the article would exceed the revenue that the website would be able to generate from ad revenue. If this were to be the case, it would make the project cost-prohibitive to implement, as we would be losing money in order to run the service for our users. This limitation would be further exacerbated if users use the service to try to summarize very long articles. This limitation on length is of primary concern because the primary purpose of our service is to specifically reduce the length of very long articles, and, thus, in struggling to successfully summarize articles that are particularly long, our service is inherently not useful. Additionally, if we wanted to fully implement this project, it would also require significant understanding and technical knowledge of how we would automate the process of posting HITs to Mechanical Turk and extracting output CSVs from the website so that we ourselves would not have to post and retrieve them by hand and would instead have that entire process happen automatically when the user submits a URL. These limitations would pose challenges if we were to scale our project up to a full website.

Our results did not deviate from what we would have expected based on our previous work nor from what we learned in class. For our initial analysis of data that we gathered from our classmates, we found that our summarizer was voted to be more effective than the online summarizer. In our analysis of the final conclusion of the project, we also found that our summarizer was better than the summarizer that we compared it to online based on the voting of the workers. This is in line with what we would expect based on what we learned in class in that we spent a good part of the semester discussing how the collective intelligence of crowds was an effective tool for efficiently and accurately completing large and challenging tasks. Thus, the task of summarizing an article (where some people might err on the side of being too brief while other people might err on the side of being too verbose while still others might spend a significant amount of time on picking the best sentences to keep or delete while others might just click randomly to complete the task as quickly as possible) is one where the aggregation of all of the possible contribution permutations from our crowd would, in aggregate, result in a product that actually was of high-quality. Thus, the fact that we found our results to create high-quality summaries is an indication that the aggregate intelligence of the crowd was an effective tool in completing this summarization task, which is in line with what we expect from what we learned about crowds over the course of the semester.

Technical Challenges

Our project required a substantial technical component. The majority of our project was coordinating the back end of going from an article pulled from The New Yorker to a summary of that article based on the votes of the crowd workers. The other technical component of the project that also required significant work was designing the HITs. Originally, our HIT design was rather rudimentary, as we only used the few things we were familiar with in HTML (namely, checkboxes). We wrote code to insert numbers at the end of every sentence and then created checkboxes that were labeled with numbers so that users could check the checkboxes that corresponded with the sentence numbers that they thought were important. While this HIT design worked, since many of the article chunks had a large number of sentences, we needed to have a very large number of checkboxes for workers to be able to select the important sentences.

Additionally, given that we wanted to compare two different summarizing techniques—one that asked users to select important sentences and another that asked users to delete unimportant sentences to compare the quality of the summaries generated from each technique—we needed a better method that was more reflective of the task that users were actually doing; that is, highlighting or blacking out important or unimportant sentences, respectively. Further, because of the way we were numbering sentences, we did not know how to pass in the sentences as a CSV file, so we resorted to hard-coding in the passage into each HIT so that each passage required its own HIT, as opposed to one HIT that would handle the voting for all the passages in the whole article. After using this HIT design for our initial round of data collection, we realized we needed to redesign our HIT in order for it to function a lot better. In order to allow people to select sentences that they thought they should delete or that they thought were important by simply clicking on the sentences, we needed to learn some JavaScript, which we were all completely unfamiliar with. While there was definitely a learning curve to understanding the sample code that used JavaScript that we were given by Professor Callison Burch, once we understood this code, it made our HIT design a lot cleaner and a lot more intuitive for our users. Now, our users were able to click on sentences in the sections of text from the articles and they would be immediately highlighted in the selecting-important-sentences HITs and immediately blacked out (highlighted in black so as to be completely redacted) in removing-unimportant-sentences HITs.

In addition to making the workflow better for the Turkers, this change also allowed us to pass in text to the HITs, so the passages were no longer hard coded. This also had the added benefit of making it a lot easier and more efficient to keep track of what votes were for what sentences, as we could easily match the vote to the sentence in the CSV while before we had to do this on a per section bases as not all the sections had exactly the same number of sentences. Further, this change also meant that we could use a single batch to post an entire article as opposed to having to do a number of independent batches for any one article, which makes automating the process of extracting the summary a lot easier as we know longer have to take all the outputted CSVs and compile them together in order to have a complete picture of all the data. However, this change was not perfect in that it also forced us to have to do some significant changes to our quality control and aggregation methods to accommodate the new format of the data. While this change might seem trivial, it actually required a significant amount of work and understanding and updating code to accurately be able to assess and interact with the new data structures.

Beyond the initial data collection HIT, we also needed to learn to improve the HIT we used to compare the quality of the summarizes we generated. While we initially only had a single Turker-generated summary that we needed to compare to the online one, we now have two summaries (one for the selection of sentences and one for the deletion of sentences) and we needed to compare both of these to the online generated one. Additionally, in order to try to eliminate bias in workers' selection to select the summary at the top in a list, we also wanted to order the summaries that appeared in the HIT into columns. This required us to deepen our familiarity with JavaScript and learn some CSS, which we were also completely unfamiliar with prior to this project. Using these new techniques, we were able to design a new HIT for the voting on summaries that change the order that the summaries appeared and allowed us to display all three summaries at once horizontally.

Another technical component that required us to have to expand our skill set was learning how to parse the articles we had selected in any meaningful way. In order to do this, we needed to become familiar with the Natural Language Toolkit (NLTK) Library in Python. This library allowed us to parse the articles into sentences while handling some of the edge cases where periods were used in acronyms or other obscurities so that we could delineate sentences in our HITs so we knew which sentences a user was clicking on. Put together, these challenges worked not only to challenge us, but gave us the opportunity to leverage the tools at our disposal to learn new things that we were able to use to make our final project better and work together as a whole. For all of us, this was the first time that we had created an entire software engineering project from start to finish—especially one that was so open-ended—and so it required having the mental flexibility to approach challenges with solutions we were unfamiliar with and pull on our knowledge from a variety of sources try and solve them.

By far the largest technical challenge we faced was trying to figure out how to create an aggregated summary from the votes of the workers. Initially, we struggled with determining how we were going to find the sentences that had the greatest number of votes for or against them given that we had been forced to divide the article into chunked portions that were then hard coded into each HIT. Thus, we needed to have some way of associating votes with sentences for each HIT and then also finding a way to compile all the output CSVs in order to be able to construct a summary. In our initial HIT structure, we had to do this more or less by hand by keeping track of the number of sentences in the chunk and associating that with the sentences that way before then manually merging the DataFrames for all the CSVs.

Additionally, we add to have some way of keeping track of the order the sentences appeared in the original article in order to construct a summary that was in chronological order. To do this, we wanted to create a dictionary where we mapped sentences to integers representing their order. While this wasn't challenging to initially create, we realized later that these dictionaries were not working because many of these political articles included the names of people and places that had accented characters and the encoding for these characters was being translated into random gibberish characters when we tried to pass this text into Mechanical Turk. Thus, we figured that we needed to clean the articles of these characters before we posted them and before constructing the dictionaries in order for the dictionaries to work as we wanted them to. In order to do this, we needed to familiarize ourselves with the unicodedata module in Python that allowed us to remove the accents from the text. With this, we were able to create a working dictionary where we could lookup sentences and find the order of the sentences in the article in order to properly construct the summary. Having overcome these challenges, we were able to efficiently construct effective summaries from the data using the updated HIT structure, code to aggregate the votes per sentence, the updated dictionary, and additional code to aggregate the selected sentences into a paragraph.

Other Info

If we had more time, we would consider trying other ways of evaluating the quality of our summaries. A way we could potentially have more quantitative feedback on the quality of our summary relative to that of the online summarizer would be the use the crowd to generate reading comprehension questions for some the articles that we chose to summarize that we could then turn around and use to test workers with after they have read our summary or the online generated one. Their results on these tests could be used to provide a more informed, quantitative analysis not just on what summary users prefer, but also as far as which summary is actually transmitting more information, as these two metrics—information transfer and personal reader preference—may not necessarily turn out to be the same.

TL;DR

Since this paper is quite extensive, we also decided to apply our system to this report! Here is the crowdsourced summary of our report:

Selection summary:

Beyond those specific to news, there are numerous online summary generators that allow users to put in text and the service automatically generates a summary of that text. TL;DR is a business idea which uses crowdsourcing. Finally, we will post another round of HITs where Turkers are presented with the two Turker-generated summaries and the summary created by the online text summarizer. Then we tally the number of votes each summary gets and determine how the quality of our summaries compared to that of summaries generated by existing tools. The primary limitation of our product is that, if we were to implement it as a website where users are able to input the URL of an article that they wish to have summarized and then the website returns to them a Turker-generated summary of that article, there would likely be a considerable time delay between the time that the user inputs the URL and the time the workers are able to complete all of the HITs required to summarize the article in order for the article summary to be returned to the user. We posted HITs on Mechanical Turk and offered a reward for completing each HIT in which Turkers select or delete sentences. 10 unique Turkers looked at each section of each article, and approximately 50 Turkers in total gave us input.

Deletion summary:

TL;DR uses crowdsourcing to generate a summary of long and verbose text. In the HIT for deleting unimportant sentences, when a worker clicked a sentence, it became highlighted in black, essentially redacted or deleted. Our crowd workers must be literate in English at a high school level at the least, as the passages we are asking them to read often contain complex concepts and difficult vocabulary words. We were concerned with the quality of the responses of the crowd because it was vital that the Turkers actually select important sentences which were not tangential to the main idea, or overly specific. We also created a HIT on Mechanical Turk which asked workers to select the best summary out of three summaries, one generated using each of the above strategies. Given the broad application of a summarizing tool, our project could be used in a variety of different concepts and by different people. A long research paper or news article is approximately 25-30 paragraphs, which would cost around $4.00 to summarize. Specifically, the workers were asked to “Select the summary that is more coherent and relevant to the title.” By far, the biggest challenges that we had to deal with were the technical challenges that we will touch on later. One of these challenges that we were not able to overcome was an issue with the way we parsed sentences. First, we gather a news articles from The New Yorker. Next, we aggregate the data and generate two summaries, one from the HITs where Turkers selected important sentences and one from the HITs where Turkers deleted non-important sentences. Finally, we will post another round of HITs where Turkers are presented with the two Turker-generated summaries and the summary created by the online text summarizer. Our project did require a substantial technical component in that the majority of our project was coordinating the back end of going from an article pulled from The New Yorker to a summary of that article based on the votes of the crowd workers. To incentivize the crowd to complete HITs where they need to select the best summary, we offer a $0.05 reward for each HIT.

Sources

For all the articles used for summarization, please see the data folder in the Github repository where they are stored as PDFs and also as txt files.