5-2020

Identifying Privacy Policy in Service Terms Using Natural Language Processing

Ange-Thierry Ishimwe

Identifying Privacy Policy in Service Terms Using Natural Language Processing

An Undergraduate Honors College Thesis

in the

College of Engineering
University of Arkansas
Fayetteville, AR

by

Ange-Thierry Ishimwe

Thesis Advisor: Dr. Qinghua Li

Thesis Defense Committee Members:

Dr. Qinghua Li (Chair)

Dr. James Parkerson

Dr. Susan Gauch

ACKNOWLEDGMENTS

I want to thank Dr. Qinghua Li for working with me and for guiding me throughout this research. I learned a lot on this project, so thank you so much! Additionally, I would also like to thank my parents and my brothers, you guys have been great role models for me, and I cannot thank you enough. I want to dedicate this thesis to you guys; it is the least I can do.

Contents

ABSTRACT

Ever since technology (tech) companies realized that people's usage data from their activities on mobile applications to the internet could be sold to advertisers for a profit, it began the Big Data era where tech companies collect as much data as possible from users. One of the benefits of this new era is the creation of new types of jobs such as data scientists, Big Data engineers, etc. However, this new era has also raised one of the hottest topics, which is data privacy. A myriad number of complaints have been raised on data privacy, such as how much access most mobile applications require to function correctly, from having access to a user's contact list to media files. Furthermore, the level of tracking has reached new heights, from tracking mobile phone location, activities on search engines, to phone battery life percentage. However much data is collected, it is within the tech companies' right to collect the data because they provide a privacy policy that informs the user on the type of data they collect, how they use that data, and how they share that data. In addition, we find that all privacy policies used in this research state that by using their mobile application, the user agrees to their terms and conditions. Most alarmingly, research done on privacy policies has found that only 9% of mobile app users read legal terms and conditions [2] because they are too long, which is a worryingly low number. Therefore, in this thesis, we present two summarization programs that take in privacy policy text as input and produce a shorter summarized version of the privacy policy. The results from the two summarization programs show that both implementations achieve an average of at least 50%, 90%, and 85% on the same sentence, clear sentence, and summary score grading metrics, respectively.

1. INTRODUCTION

Data collection from mobile and website applications (web apps) has seen exponential growth in the last decade, mainly because our usage data from mobile and web apps contains information such as our preferences, interests, and locations that can be sold to advertising agencies to use for targeted advertising (Ad). Now that collecting user data is a form of revenue stream, it motivates technology (tech) companies to track and collect as much data as possible from users. For instance, when a user uses his or her personal computer (PC) to shop online, the shopping website(site) creates a file called a cookie on your PC which contains details of your activities on the shopping site, and later that cookie file is used by other sites to generate Ads comparable to your shopping activity [1]. This amount of information on users which circulates to different companies raises an issue on data privacy. To combat this issue, there are a couple of solutions. For PCs, a user can add an Ad-Blocker extension to their search engine such as Google Web Search (Google), which blocks any tracking scripts and Ads. As for mobile phones, users can add restrictions to apps that disable tracking and collection of the phone's location and only permit it when the app is in use or never; also, users can control whether to permit or not permit apps to access their media files or phone's contact list. Although those solutions solve the issue, developers of apps and sites have found ways to circumvent the solutions. For instance, On PCs, if a user visits a news site using Google equipped with an Ad-Blocker, the site might not permit the user to see anything until the Ad-Blocker is disabled. As for phones, most apps need to be granted permission to media files or contact lists from the point of installation until the user removes the app, or they will never be able to use its functionalities.

Although software companies gather user data, they all have privacy policies that describe in detail how they collect a user's data, how they use that user's data, and lastly, how they share that user's data. Besides, every privacy policy clearly states that with the use of their app or site the user agrees to their terms and conditions. This brings us to this research, which was inspired by a mobile app called FaceApp which in 2019 was one of the most popular apps on the market because it would take an image of its user and age them realistically. FaceApp made many controversies since it would send the user's image to the cloud to be processed and aged instead of doing it locally on the phone without users' permissions, and IOS users reported that FaceApp was still able to access their camera roll even after denying permission [4]. Although an endless number of fears and issues have been raised on data privacy, the *Deloitte* survey has found that 91% of consumers agree to the legal terms and conditions without reading them [2]. To put it into perspective, in 2019, 204 billion mobile apps were downloaded [3], meaning 186 billion apps were downloaded and used without knowing how and why their data is collected. The Deloitte survey explains that the high percentage is due to the complexity of the terms and conditions [2].

This project aims to create a program that takes in any privacy policy as input and uses specific keywords extracted from multiple policies of the same category, along with the Ed Munson Natural Language Processing (NLP) algorithm [10], to summarize the terms and services into a more digestible privacy policy which highlights only the vital information as the output.

The remainder of this thesis is structured in the following manner. Chapter 2 reviews the related work on text summarization. Chapter 3 describes the implementation of the two summarizers. Chapter 4 presents the analysis and results, and Chapter 5 concludes this thesis and discusses future work.

2. RELATED WORK

NLP is part of the machine learning field concerned with understanding, analyzing, manipulating, and generating human language [5], as mentioned by Badreesh Sheety. NLP is responsible for autocorrect, machine translation, and, more importantly, summarization [5]. Before starting this thesis work, we researched on NLP and summarization techniques and found a website called SMMRY [6]. SMMRY takes in articles or text as input and uses a ranking system in its algorithm to assign points or weights to each sentence by calculating the frequency of each word in the full text, and afterward, it allocates points to each word based on their popularity [6]. Finally, the program outputs the phrases with the most weight in chronological order. Although this system works, it does not allow users to choose which words are more important than others. Therefore, if the program is summarizing a privacy policy, there is a high chance of ranking a sentence that contains crucial information on user data collection well below a sentence of no value to the user since points of phrases are assigned based on frequency. TLDR This [7] and I Lazy To Read [8] all follow the same concept as SMMRY but produce only five of the highest-ranked sentences in the article, unlike SMMRY, they do not provide a user with a choice for the number of phrases to generate for the summary. In addition, choosing only five sentences, for a summary, is ineffective because, in this thesis, it shows that the average phrase count in most policies is 205 while our summaries average 23 sentences. Therefore, the goal of this thesis is not only to be able to summarize any privacy policy but also to use the words chosen by the user to be able to generate summaries that contain information on details a user finds to be crucial.

3. IMPLEMENTATION

This chapter contains details on how different summarization algorithms were compared to find the best one, how data was collected and manipulated to work with the algorithm, and lastly, how the privacy policy summarizer was built with its two different variations.

3.1. SUMMARIZATION ALGORITHMS

We looked into summarization using python and, more specifically, using NLP. We found that there are mainly two methods used to summarize text using NLP, which are abstraction-based summarization and extraction-based summarization. Abstraction-based summarization is a method of summarization where a program analyzes and interprets a text by using NLP methods to produce a summation that still conveys the same information but, in the program's generated words [9]. The other method of summarization is extraction-based summarization, which uses a method of choosing sentences based on their scores [9]. While both methods achieve their goal of summarization, we chose to use the extraction-based method because Praveen Dubey, a Data scientist, published an article where he compared both methods against each other and concluded that extraction-based methods offer better summaries than abstractive methods [9]. He explained that it is because "abstractive methods cope with problems such as semantic representation, inference, and natural language generation, which is relatively harder than data-driven approaches such as sentence extraction [9]." There are a couple of different extractive summarization algorithms, such as Luhn Summarizer [12] or LexRank Summarizer [13], which all use some form of a scoring system based on the frequency of a word in a sentence. Although they produce decent summaries, they are not ideal for privacy policies since users cannot denote the words that are relevant to them. The only NLP algorithm able to meet all the criteria is the Ed Munson algorithm [10]. The ed Munson summarizer is an old algorithm that was published in 1969 by H.P. Edmundson. This algorithm is different from the other extractive methods because it includes the already known features in summarization, such as position and word frequency used in Luhn's method [12], and also, it introduces two more features: cue words and document structure to produce summaries [11]. Each of the four features has its methods used for assigning weights or points to words and sentences. The first method is the cue method, which deals with cue words [10]. Before running the algorithm, users need to enter cue words, and the program accepts three types of cue words: bonus words, stigma words, and null words. Bonus words like the name suggests are words that are of high significance to a user, for example, name, email, address, et cetera [10]. The program considers bonus words as positives when assigning weights to sentences. Stigma words are words that hold no value to a user, and they are negative when performing summarization. Lastly, null words are the neutral words meaning they have no effect on the summary, and null words are also known as stop words because they usually contain words such as which, at, et cetera. It should be noted that a group of cue words is known as a cue dictionary [10]. Therefore, the cue method uses a cue dictionary to assign weights to each word in a sentence that also appears in bonus, stigma, and null words. Subsequently, the weighted words in each sentence are all summed up to produce the weight of the phrase.

$$\boldsymbol{Cue\ Weight\ =\ \sum(Cue\ weight\ of\ each\ word\ in\ a\ sentence)}\ [11]$$

The second method is the key method, which contains a non-cue glossary dictionary—a list of words and frequency of a word in the document or input text not listed in the cue dictionary – where the non-cue words are sorted in decreasing order according to their frequency [10]. The paper *New Methods in Automatic Extraction* [10] explains that "the frequencies are cumulated in decreasing (highest downward) to a given percent of the total number of word occurrences in the document. Non-Cue words with frequencies above this threshold are designated Keywords and are assigned positive weights equal to their frequency of occurrence in the document [10]. The final key weight of a sentence is the sum of the Key weights of its constituent words." The third method is the title method, the title method, similar to the key method, has a title glossary – a list made up of non-null words of the title, subtitle, and heading off that document [10]. The same paper mentioned above explains that the words are all assigned positive weights and that the final title weight for each sentence is the sum of the title weights words found both in the title glossary and the sentence.

$$\boldsymbol{Title\ Weight} = \sum (\boldsymbol{Title\ weight\ of\ each\ word\ in\ a\ sentence})\ [\mathbf{11}]$$

The last method is the location method; the location method hypothesizes that in the text one will find the most relevant and vital phrases at the beginning or end of a document or text [10]. The last method uses a prestored heading dictionary – a list of selected words that are found in titles of documents such as introduction, implementation, conclusion, etc. The paper *New Methods in Automatic Extracting* [10] mentions that the heading dictionary contains 90 words, which are all assigned positive weights. The process of assigning weights to words is the same as the title method. In addition to the heading dictionary, the latter method assigns positive weights to sentences according to their ordinal position i.e., from the first and last paragraphs as well as the first and last sentences [10]. The final weight of a sentence is calculated by cumulating the weights of each sentence on ordinal position and heading weight. When all the methods are through calculating their final weight, the sum of the four calculated weights is derived to obtain a final score of the sentence. When a user specifies the number of sentences to produce in summary i.e., six sentences, the algorithm will output six sentences with the highest score in the order they appear in the text.

$$\boldsymbol{Score} = (\boldsymbol{W1} \times \boldsymbol{P}) + (\boldsymbol{W2} \times \boldsymbol{F}) + (\boldsymbol{W3} \times \boldsymbol{C}) + (\boldsymbol{W4} \times \boldsymbol{S}) \text{ [11]}$$

Position (**P**) Word frequency (**F**) Cue word (**C**) Document structure(**S**)

Weights (**W**)

The ability to use four features instead of the usual two to summarize text and especially the feature which allows users to pick which words are vital to them is why the ed Munson algorithm was chosen to be used in the implementation of a privacy policy summarizer.

3.2. DATA COLLECTION

.

Entertainment	Social Media	Photo Editor	Dating	Transportation
Musixmatch	Facebook	Visco	Bumble	Lyft
Shazam	Instagram	FaceApp	Tinder	Transit
SoundCloud	Snapchat	Prisma	Hinge	Uber
Spotify	TikTok	Instagram	Christian Mingle	Turo
Tidal	Visco	Visco	OkCupid	Veoride

Table 1. The full list of apps that were used in extracting the bonus words

Having a database of bonus words that can help summarize any privacy policy in producing applicable information means that the bonus words used should be extracted from each category of mobile apps on the market. For this project, five categories were investigated and are as follows: social media, transportation, entertainment (media), photo editors, and dating. Within each category, five mobile apps were chosen to use for building specific bonus words for each corresponding class as well as general bonus words that apply to any policy document on the app market. **Table 1** shows the list of apps that were chosen.

When we looked into different privacy policies, we found that policies share a lot of the same words and also that different categories have their own specific words. Therefore, we split bonus words into specific bonus words and general bonus words. One of the sub-goals of the program was to be able to use the specific bonus words and the general bonus words in unison to summarize any policy. To find the specific and general bonus words, we had to emulate how the ed Munsons algorithm summarizes by extracting complete phrases to make the summary for every app policy listed in **table 1**. Simultaneously, we took the point of view of a consumer concerned about data privacy, meaning we perused through the policies and would only extract phrases that we deemed would be vital to consumers. Subsequently, we would analyze every phrase in the summary and would only pick out words that stand out or that make the phrase vital to the consumer. **Figure 1** shows two phrases extracted from a summary of the Lyft mobile app policy, where the words highlighted in yellow are examples of words typically found in every privacy policy and the words in green are specific words found in each app category. Currently, most apps on the market require users to create an account in order to use their features, but the amount and the type of data collected from users differ from each app category. For instance, in **figure 1**, Lyft requires the users' driver's license, vehicle information, et cetera, but a mobile app such as Tinder that is in the dating category will ask for details such as height, religion, dating preference, et cetera.

If you apply to be a Driver, we will collect the information you provide in your application,including your name, email address, phone number, birth date, profile photo,physical address, government identification number (such as social security number), driver’s license information, vehicle information, and car insurance information.

We collect the payment information you provide us, including your bank routing numbers, and tax information.

Bonus Words	
General	**Specific**
apply	government
collect	identification
information	number
⋮	⋮
provide	social
application	bank
name	routing
email	tax

Figure 1. Two phrases from the summary of the lyft mobile app, and the bonus words from general to specific.

The different types of details required about users that usually differ with each app category were placed in the specific bonus words for that app category. The general bonus words received details such as name, email address, birthday, et cetera, because a profile cannot be created without those details. Also, words such as collect, information, create, and many others were also added to the general list of bonus words. Those words, although they are not part of a user's profile, they are readily found in phrases that carry the most weight in any policy. From **figure 1**, the words collect, information, and providers are used in both phrases, and they describe the action taking place. For example, the word collect in all the policies is always associated with information that is gathered from consumers during account creation or while using the app.

Currently, the general bonus words that can be used to summarise any policy amount to 80, and although the specific bonus word count differs for each app since some app groups require more information from their user than others, the average word count for specific words is 50.

3.3. IMPLEMENTATION OF THE SUMMERIZAR

As stated in the introduction, the goal of this thesis is to be able to summarize any privacy policy. Therefore, the program needs to be able to perform tasks from summarizing to bonus word editing, which are all explained below.

3.3.1. BONUS WORD FETCHER AND EDITOR

During the data collection stage, our goal was to gather specific and general words to use during summarization for building the database. The database is made up of a Microsoft Excel file where each column holds specific bonus words for every individual privacy analyzed. Also, all the specific bonus words are grouped according to their category, as shown in **figure 2.** The figure shows a small portion of the database. The header of the tables is highlighted with different colors to demonstrate how the specific bonus words are grouped and are arranged.

Currently, the database holds five groups of apps, as shown in **table 1**. Thus, when the summarizer starts executing, it prompts the user to choose the category of the app to summarize. Afterward, when a category is chosen, the program opens the excel file and collects all the specific words within that category and the general bonus words. Lastly, the program checks for duplicate words from the group of columns selected since all privacy policies will be from the same grouping so they will contain identical words, and thus, removing duplicate words puts the specific bonus words on equal footing.

	D	E	F	G	H	I	J	K	L	M	N
1											
2	Visco	Prisma	Bumble	Tinder	christian mingle	Hinge	Musixmatch	shazam	SoundCloud	Spotify	Tidal
3	privacy	privacy	download	create	credit	logs	provide	ID	platform	sign	solicit
4	policy	policy	account	account	card	third	automatically	credit	features	street	parental
5	companies	companies	collect	login	demographic	parties	receive	purchase	password	country	online
6	control	collect	information	credentials	dating	social	third	product	public	third	geolocation
7	accessing	use	name	details	profile	media	parties	update	city	party	metadata
8	service	share	username	service	via	info	join	register	country	accessing	commercial
9	collection	protect	email	gender	store	detail	lyrics	connect	picture	interactions	preferences
10	storage	information	mobile	date	IP	login	upload	credit	profile	queries	subscriptions
11	use	service	gender	birth	UDID	credentials	comments	card	image	streaming	formatting
12	personal	agree	date	profile	unique	profile	write	social	websites	playlists	categories
13	information	platform	birth	share	identifier	personality	forum	media	social	library	transferred
14	name	unregisterd	sexual	information	access	lifestyle	microphone	online	media	interests	receive
15	e-mail	user	photographs	personality	URL	interests	library	provide	links	preferences	identifiers
16	address	content	location	lifestyle	preferences	photos	play	develop	page	photos	usage
17	register	photos	login	interests	street	videos	songs	operate	communicate	URL	activity
18	user	visit	social	content	city	promotions	favourite	deliver	upload	IP	playlists
19	account	cookies	media	photos	state	events	communicate	identity	comments	network	tracks
20	messages	technologies	connect	videos	third	customer	activities	identification	discussions	operating	offlined
21	retain	beacons	contact	subscribe	party	interaction	public	occupation	interactions	system	content
22	send	pixels	customer	paid	ID	process	profile	language	visit	sensor	access

Figure 2. A small part of the bonus words used during summarization.

In short, when summarizing, the bonus words collected from that apps' category are grouped together to be used to summarize a policy of the same category.

3.3.3. ITERATIONS OF THE SUMMERIZER

The Ed Munson algorithm is already implemented in Python under the Sumy Natural Language Processing library [14], so there was no need to implement the algorithm since it is open source. When using the Ed Munson algorithm, users can control four features, which include bonus words, stigma words, null words, and the number of sentences to extract from the input phrases. Although the Ed Munson library does not provide much freedom when operating it, two distinct designs of the summarizer were implemented.

The first design is the simplest; it fetches the bonus words from the excel database as tuples and removes the tuples as described in section 3.3.1 bonus word fetcher and editor. Subsequently, the bonus words and stigma words are assigned. Lastly, the Ed Munson algorithm starts the task of summarizing the input text and outputs summarized phrases equivalent to the sentence count stated.

The second design was inspired by the computer science method recursion, where a complex problem is broken down into simpler versions of the problem to find the solution. Although the approach for the second design is inspired by recursion, it does not share the same mechanics. The second design starts by counting the number of phrases in the privacy policy entered, and it uses the phrase count to start a while loop where the number of sentences in the policy is compared against the desired number of sentences that the program should output. When the full privacy sentence count is greater than the desired summary count, the program enters a while loop and starts the second step. In the second step, the program divides the sentence count of the privacy by half, and then it follows the first and second steps from the first design. While the program is executing the second step, it uses the sentence count from the privacy policy that was divided by half to output a summary that has half as many phrases as the summary. The new summary produced will be used as a new input in the while loop, and the same process continues until when halving the sentence count of the input is less than the desired number of phrases, which breaks the program out of the while loop. Afterward, the program takes the last summary that was produced before the summary sentence count is less than the desired phrased count, to use as the last input to produce a summary with the desired number of phrases. For example, suppose a user wants to summarize the privacy policy for the mobile app Turo which has 205 phrases and wanted to generate 27 sentences. The program will first produce a summary of 102 sentences, which is half of the original when rounded down; afterward, the 102-sentence summary will then be used as input text to output a summary of 51 phrases. Since half of 51 is 25, which is lower than the desired sentence count, the computer recognizes this and uses the 51 phrases summary to produce a summary with the desired 27 sentence count. The results and evaluation of both implementations are found in Chapter 4.

3.4. Evaluation

As stated before, the program takes a large text and summarizes it. Therefore, the next logical step is to have a scoring system that analyzes each summary impartially and indicates how effective the output is. Henceforth, three evaluating metrics were created and are as follows: clear sentences, same sentences, and summary grade. The results of the scoring will be discussed in the results section.

3.4.1 CLEAR SENTENCES

```
With other Match Group businesses, Hinge is part of the Match
Group family of businesses which, as of the date of this Privacy
Policy, includes websites and apps such as Tinder, OkCupid, Plenty
of Fish, Match, Meetic, BlackPeopleMeet, LoveScout24, OurTime,
Pairs, ParPerfeito, and Twoo(for more details, click here).

We share your information with other Match Group companies: - For
them to assist us in processing your information, as service
providers, upon our instructions, and on our behalf.
```

Figure 3 A scenario where the computer chooses a phrase that is not clear

Ed Munson is an NLP algorithm, and it uses a set of predesigned rules to summarize sentences. Therefore, when it produces an output, it is because the sentences chosen have the same words as the bonus words set before running the program. This means that if a phrase has ample bonus words but is not clear when it stands alone without accompanying sentences that give context as to what is being done to users' data, that unclear phrase still has a high chance of being chosen. Henceforth, the evaluating metric clear sentences is used to show the number of sentences in the summary that are clear out of the sentences generated. **Figure 3** shows part of the privacy policy of Hinge, a dating app. In the figure, the phrase highlighted in green is chosen by the summarizer while the other sentence above it, is left out. In the generated summary, when a user reads the highlighted phrase alone, the phrase in green is not clear on who the Match Group companies are that they will be sharing user data with. However, if the algorithm had chosen both phrases, they would work in unison to give more context to the user. Hence the higher the score on clear sentences, the better the summarization**.**

3.4.2. SAME SENTENCES

As mentioned in the data collection section, we emulated the Ed Munson algorithm and extracted phrases that we thought would be crucial to any user without editing them. Therefore, to perform the same sentence analysis, we used two summaries, a summary from the program and a summary from a user. As it states in the title, this metric looks for the same sentences in both summaries. It is one way of telling how valid the bonus words are in choosing vital sentences. The same sentence metric is out of the number of sentences in our summary; henceforth, the more matching sentences the two summaries have, the more detailed the information it provides to the user as to how data is collected, used, and shared.

3.4.3. SUMMARY GRADE

The first two evaluating metrics analyze two distinct features of the summaries but do not provide a complete evaluation. Henceforth summary grade analyses the overall score of the summary. It uses three features to evaluate the summary. The first two features borrow from the same sentence and clear sentence grading criteria. The last feature considers how relevant the phrases chosen by the program are to the user. It accounts for two instances; the first, a sentence can be clear as a standalone phrase, but its talking points might not be crucial to a user. In the second instance, a sentence will be clear and crucial to a user but was not included in the summary produced by a user. The overall grade is always out of the number of sentences in our summary, and if the first instance occurs, it counts as a negative point towards the total, and with the second instance, it counts as a positive point towards the total.

4. RESULTS AND ANALYSIS

Transportation	Uber	Lyft	Transit	Turo	Veoride
full	288	173	87	205	148
summary	22	29	18	27	24
Dating	**Hinge**	**Mingle**	**Tinder**	**OkCupid**	**Bumble**
full	178	860	163	153	150
summary	31	18	31	21	20
Social Media	**Facebook**	**Instagram**	**Snapchat**	**TikTok**	**Visco**
full	171	178	189	110	225
summary	35	27	34	19	23
Entertainment	**Musixmatch**	**Shazam**	**SoundCloud**	**Spotify**	**Tidal**
full	138	167	237	158	140
summary	28	19	24	17	11
photo Editor	**Visco**	**FaceApp**	**Prisma**	**Adobe**	**Instagram**
full	225	175	251	166	178
summary	23	23	26	19	27

Table 2. The list of the privacy policies with the number of sentences in their privacy policy and the number of sentences in the generated summary.

In this Chapter, we present the results for all the summaries based on their scores along the three metrics. Since the total number of points for each metric is different for every summary, all the results presented are percentages of their score against the total. While carrying out the experiment, it was noted that the summarized output always remained the same, and the only time there was a change in the summary is when bonus words were altered, added, or removed from the list, or the sentence count was changed. **Table 2** lists all the mobile applications with their sentence count for the full privacy policy and summarized policy from the program. The data from **table 2** show that privacy policies average 209 phrases and our summaries average 23 phrases. The subsections below provide results and analysis of the summaries.

4.1. FIRST IMPLEMENTATION

The results are divided into two sections, which allows for more detailed evaluations, and the last section is for the blind test.

4.1.1. SAME SENTENCES

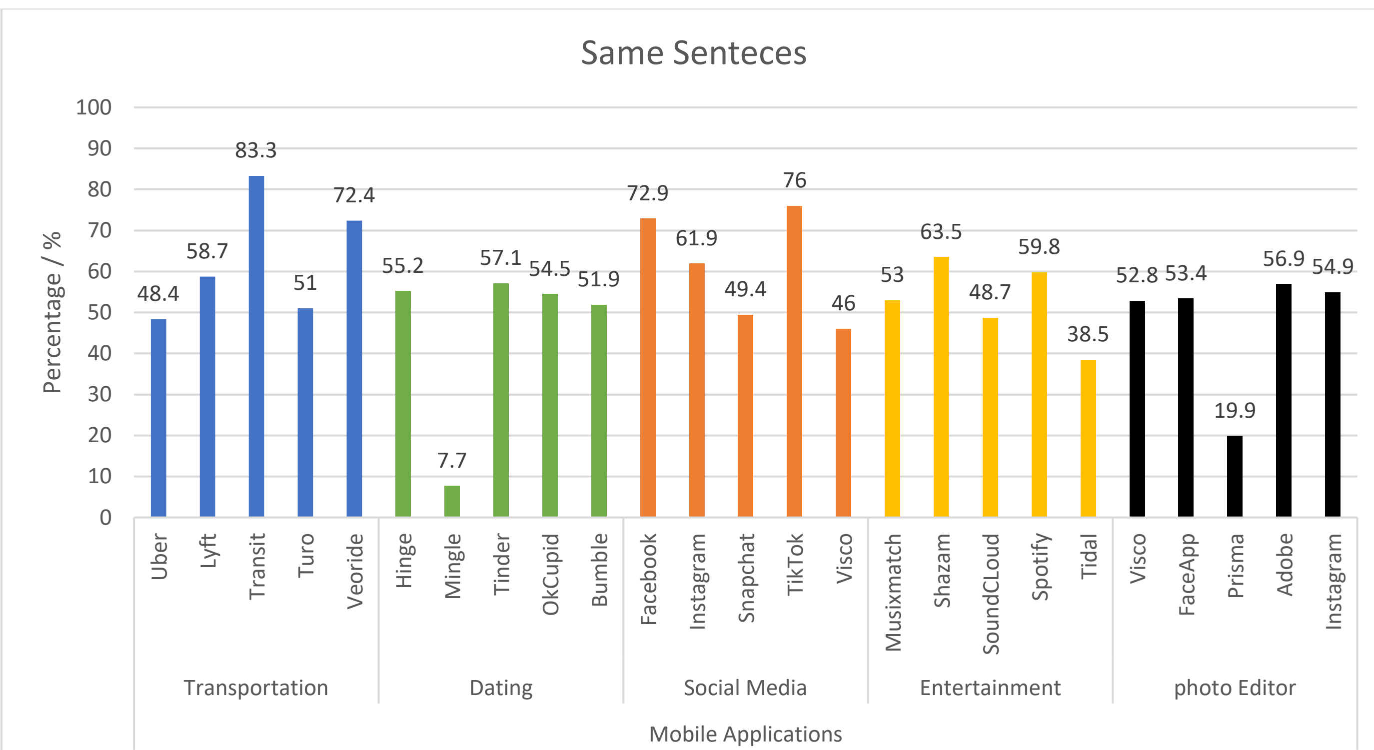

Figure 4. The results of the same sentence criteria as percentages of their scores.

The same sentence metric is meant to show how useful the bonus words are at picking the exact sentences in our summary of the same privacy policy. To have a fair comparison, we would always add three more phrases on top of the sentence count, which, as stated in the iterations of the summarizer, specifies the number of phrases to output. For instance, if the summary I made for a privacy policy is 15 phrases long, the program would output 18 phrases. The reason for adding three more phrases is because the program is never going to be perfect at producing the same phrases, so by adding three more sentences, the program has room to choose other phrases and not be penalized for it.

Figure 4 reports the results of the same sentences as percentages. Their bar graphs are color-coded according to their app category, as shown in the axis of the titles. From the graph, it can infer that no summary produced the exact same sentences as the manually generated summary, nor was there a summary that had zero matching sentences with the manually generated summary. Moreover, the average of all the summaries is 52.9%. Henceforth it shows that the program and bonus words used can pick at least half of all the sentences that were in our summary. The best result was from the Transit privacy policy which was able to produce a summary that matched mine by 83.3%, and looking at **table 2** it shows that it has 81 sentences which is a quarter of the average number of phrases in policies and is also the least out of all the privacy policies. The worst result is from Hinge, which matched our summary by only 6%. Also, Hinge has 860 phrases, which is four times the average policy, and it also has the most sentences out of all the policies. This pattern is not random; the results show that all privacy policies that have a phrase count equal to or less than the average, have 50% or more matching sentences as our summaries. The results are inversely replicated when the sentence count for privacy policy exceeds the average.

4.1.2. CLEAR SENTENCES AND SUMMARY GRADE

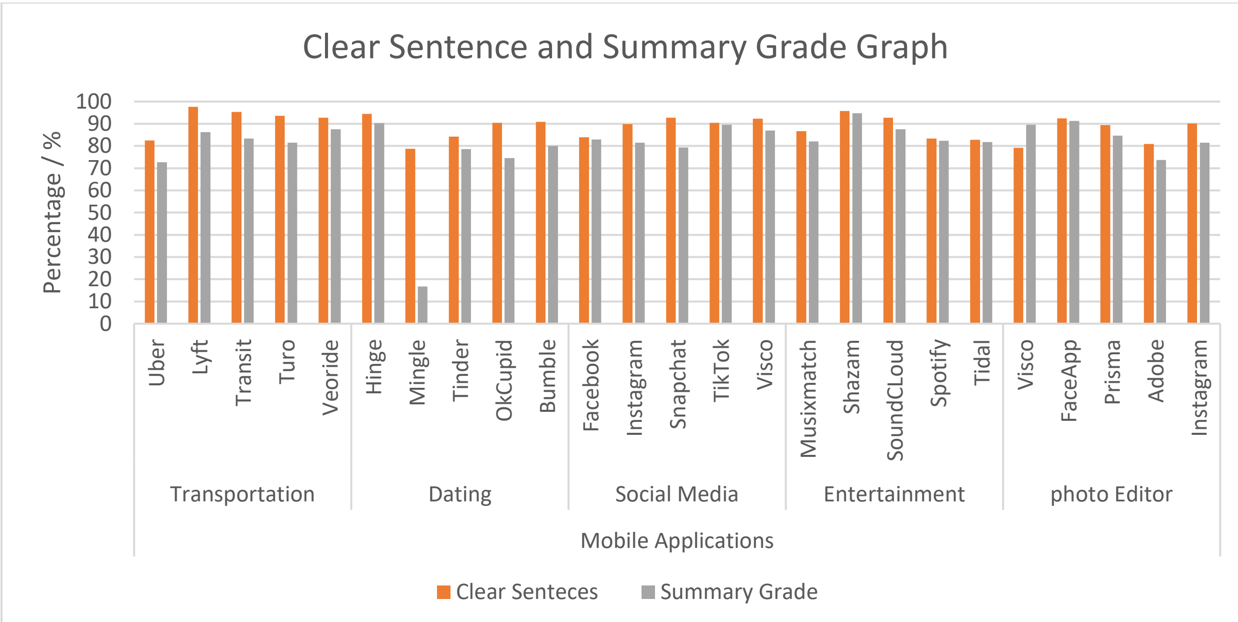

Figure 5. The results of the clear sentence and summary grade metrics.

As shown in the same sentences section, the program does not fully extract the precise same phrases as our summaries; it also extracts other sentences that were not chosen. Henceforth, there is a need for the clear sentences metric which checks how readable the phrases are to a user. **Figure 5** shows the results of the summaries for clear sentences and the overall score of the summary. Using the legend of the graph, the brown bars show that the program produces summaries that are at least 75% clear on the message they convey. The best result is from the Lyft privacy policy with a 96% score, while the worst score is from Adobe Photoshop with a score of 77%, which is also still high. There is no evidence to show that there is a pattern in clear sentences since five of the policies with the most number of sentences and the least number of sentences both average a score of 85%. The average score for all clear sentences score is 87.9%. The findings are positive since it means that as the number of sentences increases, the clarity of the sentences selected will more likely stay the same.

The last evaluation metric, which is represented by the gray bars on **Figure 5**, shows the overall score summary. The implementation section states that it analyses the phrases that are legible to a user and also whether they contain information that is valuable to the user. The findings show that the overall score is always within a 10% range below the clear sentence score. The average of all the summary grades amounts to 80%. Only one application, Christian Mingle, is an outlier with an overall score of 16.7%, while its clear sentence score is 77.8%. We concluded that the results for Christian Mingle do not follow the same pattern shown in other app policies because of the number of sentences (860) in the policy since the program has four times more phrases than the average privacy policy.

4.1.3. BLIND TEST

The blind test was carried out to evaluate how useful the bonus words are at summarizing an untouched document using the current database of bonus words. As mentioned earlier, to get the first results, we would read the privacy policy, summarize it, extract bonus words and lastly, use those bonus words to try and output a comparable summary. For this test, we took different procedures. First, we chose one app in each of the five categories that is not part of the apps listed in **table 1**. Afterward, we used all the bonus words collected from apps of the same category to produce a summary, the next step involved reading the applications policy to summarize it, and lastly, perform the evaluation criteria tests described in section 4.1 and 4.2.

The assessment on the blind test was carried out the same way in order to have a fair experiment. Each summary produced by the program was made up of 20 sentences, and the feature that changed were the bonus words used. The apps that were chosen for the experiment are Twitter, Instasize, flywheel, happn, and iHeartRadio. The results from the blind test are shown in **figure 6**. The results are Comparable to the first experiment; the average score for the same sentences, clear sentences, and summary grade are 51%, 91%, and 85%, respectively.

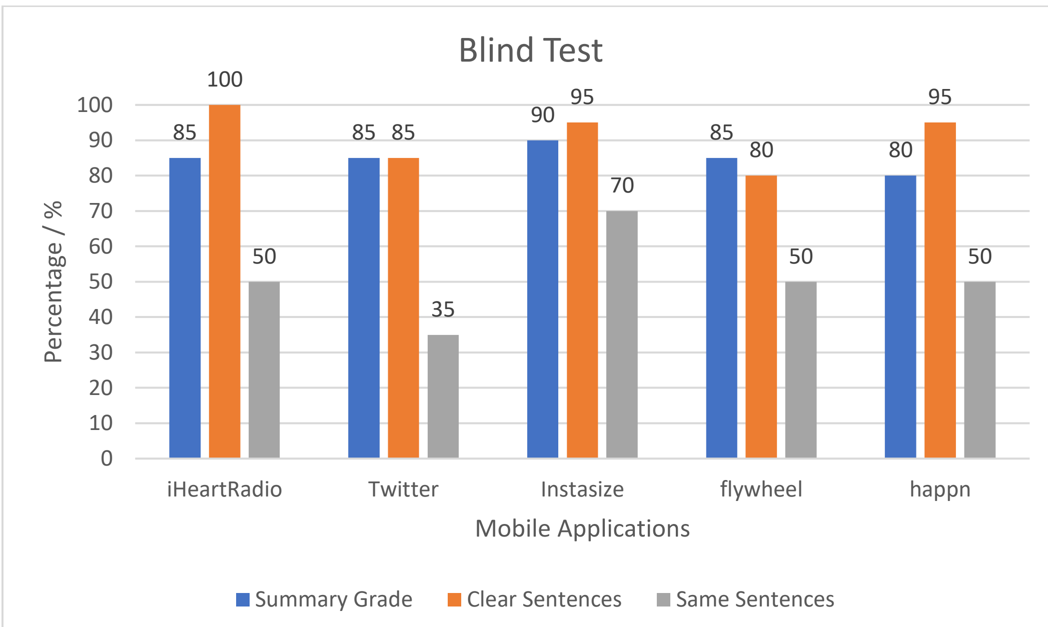

Figure 6. The results of the blind sentence test as percentages of their results.

4.2. SECOND IMPLEMENTATION

The second implementation is divided into only two sections; the first section discusses the results of the three metrics, and the last section discusses the results of the blind test for the second implementation.

4.2.1. SAME SENTENCE, CLEAR SENTENCE, and SUMMARY GRADE

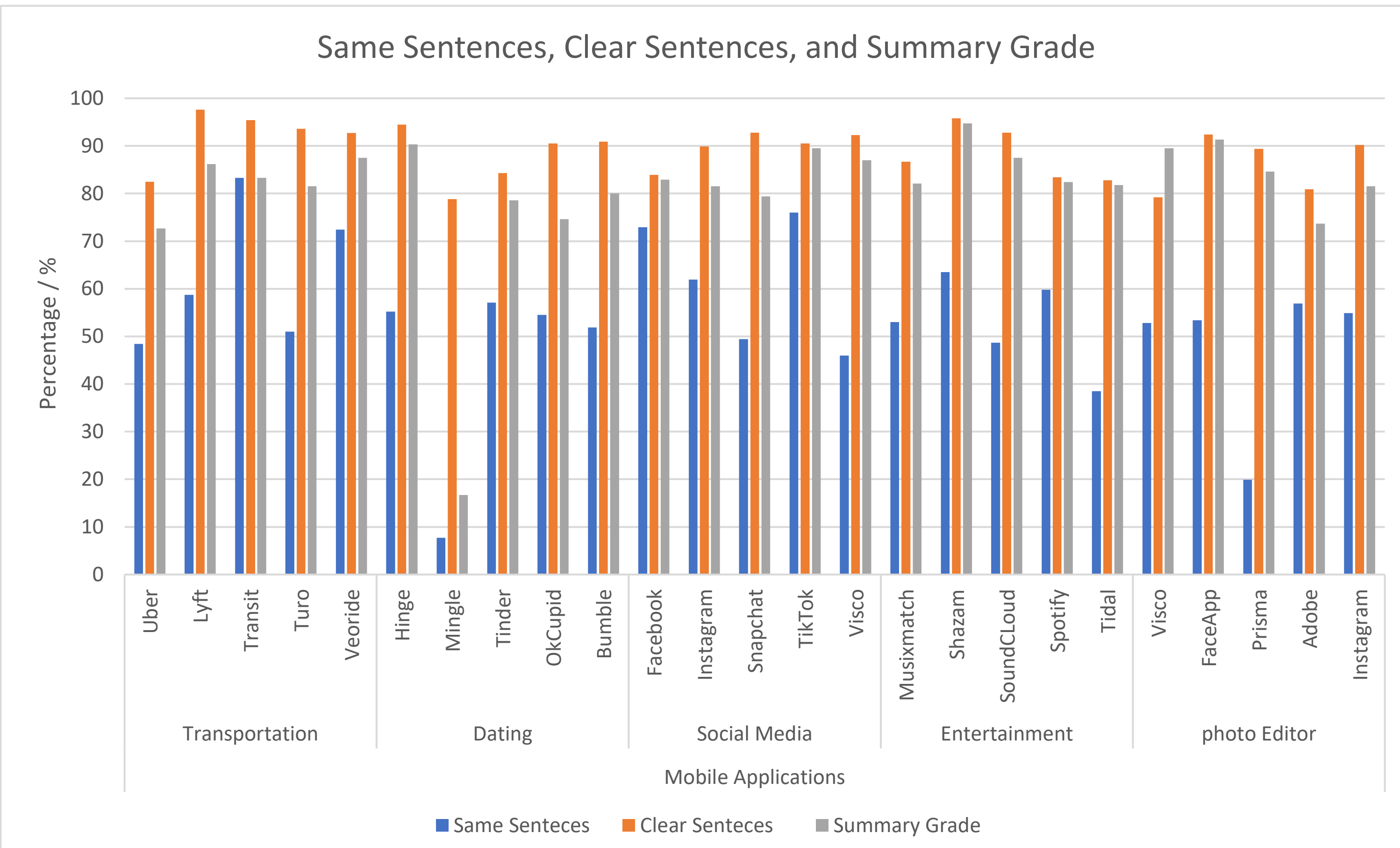

Figure 7. Shows all the results for the three grading metrics

We made the second design so that it can be compared with the first one to determine whether modifications to the design can improve the summarization, especially for the same sentence metric. The same sentence metric is emphasized because any user should be able to take any privacy policy and use our program to generate a summary that is almost a replica of what the user would produce if they tried to manually summarize a policy.

Testing of the second implementation was carried out the same way that testing of the first implementation was done. The results from the tests showed that there was not that much improvement in the program's ability to summarize. The average percentage scores for the same sentences is 53.9%, clear sentences 89.1%, and summary grade 81.3% as shown in **figure 7**. The results obtained all show positive increases in the scores, but the increase is not significant enough to show an improvement in the design of the program. While comparing the summaries that were produced by both implementations, we found that they are nearly duplicates of each other with most summaries containing no more than three different phrases.

4.2.2. BLIND TEST FOR THE SECOND IMPLEMENTATION

The blind test for the second implementation was carried out the same way as the first implementation. **Figure 8** shows the results of the second implementation, which are the blue bars; also, the results of the first implementation are added to show how the two implementations compare against each other visually.

The results show that the overall average of the same sentence metric is 56%, which shows a 5% increase from the first implementation. The rest of the results increased too. Clear sentences and summary grade both showed an increase of 3%, which brought their score to

94% and 88%, respectively. As mentioned earlier, the difference between the generated summary of the first and second implementation is always no more than three phrases. It is this subtle change that caused the increase in score because the computer chose sentences that held the most crucial information out of an already summarized text. Although it might be argued that only testing one app from each category is not enough to prove the effectiveness of the program, it does not take away from the fact that it proves that the idea and program are feasible.

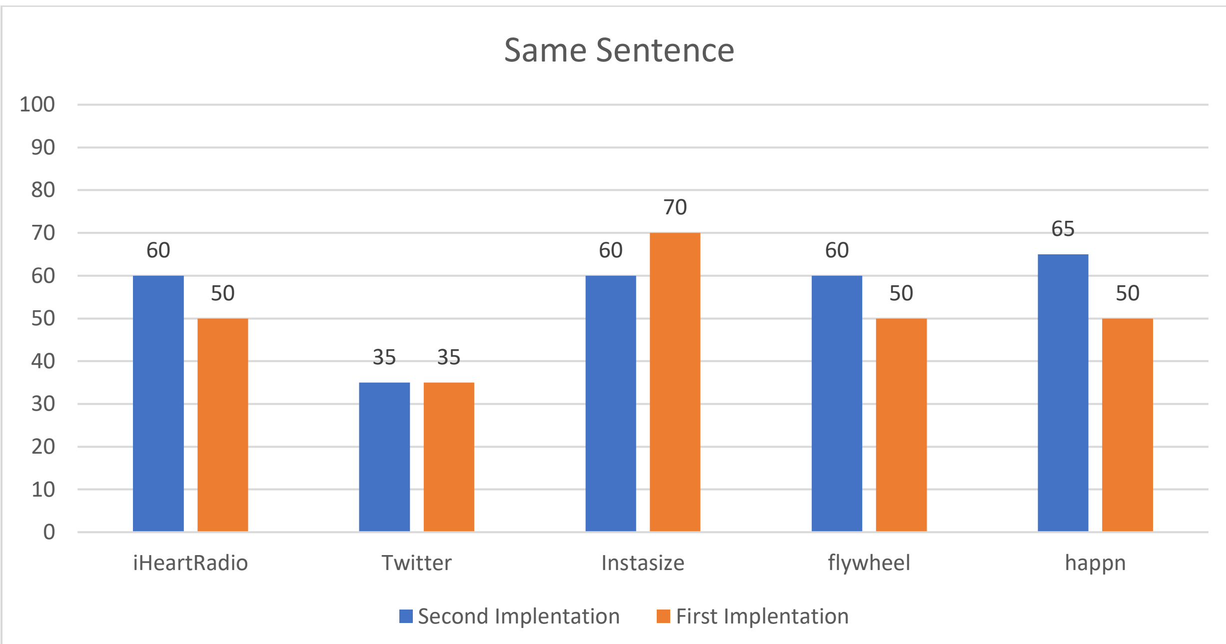

Figure 8. The blind test for both the first and second implementation.

5. CONCLUSION

The goal of this thesis was to create a program that can take any privacy policy found on any app store, and summarize the extended privacy policy to produce a shortened or condensed privacy policy that contains most of the crucial information in the policy, and is still brief enough to consume quickly. We introduced two different implementations of our privacy policy summarizer. The two summarizers are each made up of two central parts, and the first part is mainly used for fetching bonus words from the database and setting up other features that will be used during summarization, such as the desired number of sentences to produce in the summary. While the second part contains the code and logic that performs the summarization by using the ed Munson summarization algorithm to produce summaries.

The bonus words collected along with the first implementation demonstrated that the first program could produce summaries that average 51%, 91%, and 85% on the same sentences, clear sentences, and summary score metrics, respectively. Furthermore, the second implementation showed that summarization could be improved by increasing the number of summarizations that occur. The second implementation showed a gain of 5%, 3%, and 3%, which changed the scores for the same sentences to 56%, clear sentences to 94%, summary score to 88%. The results prove that the ed Munson algorithm, along with the bonus words collected, can be used to produce a summarized privacy policy that articulates at least half of all the crucial details compared to a user-generated summary.

Although the results show promise, the program performed poorly on the same sentence metric. Nonetheless, the program can still be improved upon. One of the improvements or inquiries could be to focus on all the three types of words (bonus, stigma, and stop words) instead of only focusing on the bonus words. Focusing on all three will increase the number of words in the database, which can improve the accuracy of the sentences. Another inquiry could look into improving the accuracy of the sentences by exploring the other implementations of the Ed Munson algorithm, for example, an algorithm that divides the policy into different sections that are summarized independently multiple times until all the combined different sections add up to the desired number of phrases in summary.

7. REFERENCES

[1] “The Now: What is Targeted Advertising?,” *GCFGlobal.org*. [Online]. Available: https://edu.gcfglobal.org/en/thenow/what-is-targeted-advertising/1/. [Accessed: 23-Apr-2020].

[2] “2017 GLobal Mobile Consumer Survey: US edition,” *2017 GLobal Mobile Consumer Survey: US edition*, pp. 1–18.

[3] J. Clement, “Annual number of mobile app downloads worldwide 2019,” *Statista*, 17-Jan-2020. [Online]. Available: https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/. [Accessed: 23-Apr-2020].

[4] N. Lomas, “FaceApp responds to privacy concerns ,” *TechCrunch*, 17-Jul-2019. [Online]. Available: https://techcrunch.com/2019/07/17/faceapp-responds-to-privacy-concerns/. [Accessed: 23-Apr-2020].

[5] B. Shetty, “Natural Language Processing(NLP) for Machine Learning,” *Medium*, 24-Nov-2018. [Online]. Available: https://towardsdatascience.com/natural-language-processing-nlp-for-machine-learning-d44498845d5b. [Accessed: 23-Apr-2020].

[6] “Summarize articles, text, websites, essays and documents,” *SMMRY*. [Online]. Available: https://smmry.com/. [Accessed: 23-Apr-2020].

[7] S. Garg, “TLDR This - Automatically summarize any article or webpage in a click,” *Product Hunt*, 23-Apr-2020. [Online]. Available: https://www.producthunt.com/posts/tldr-this. [Accessed: 23-Apr-2020].

[8] Z. Chew, "I Lazy To Read - Auto-summarize any online article," *Product Hunt*, 23-Apr-2020. [Online]. Available: https://www.producthunt.com/posts/i-lazy-to-read. [Accessed: 23-Apr-2020].

[9] P. Dubey, "Understand Text Summarization and create your own summarizer in python," *Medium*, 23-Dec-2018. [Online]. Available: https://towardsdatascience.com/understand-text-summarization-and-create-your-own-summarizer-in-python-b26a9f09fc70. [Accessed: 23-Apr-2020].

[10] H. P. Edmundson, "New Methods in Automatic Extracting," *Journal of the ACM (JACM)*, vol. 16, no. 2, pp. 264–285, 1969.

[11] A. Vashisht, "Edmundson Heuristic Method for text summarization," *OpenGenus IQ: Learn Computer Science*, 05-Nov-2019. [Online]. Available: https://iq.opengenus.org/edmundson-heuristic-method-for-text-summarization/. [Accessed: 23-Apr-2020].

[12] *The Automatic Creation of Literature Abstracts - IBM Journals & Magazine*. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5392672/authors#authors. [Accessed: 23-Apr-2020].

[13] Erkan, Gunes, Radev, and D. R., "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization," *arXiv.org*, 26-Sep-2011. [Online]. Available: https://arxiv.org/abs/1109.2128. [Accessed: 23-Apr-2020].

[14] "sumy", *PyPI*, 2020. [Online]. Available: https://pypi.org/project/sumy/. [Accessed: 28-Apr- 2020].