An Statistical Tool for Multi-Document Summarization

Mr.Vikrant Gupta[1], Ms Priya Chauhan[2], Dr. Sohan Garg[3]

Mrs. Anita Borude, Prof. Shobha Krishnan

[1]Research Scholar, Mewar University, [2]Senior Lecturer, MCA Deptt., RKGIT, Ghaziabad,
[3] Principal Deptt. of MCA, IIMT, Meerut.

Abstract- In this project, an auto-summarization tool is developed using statistical techniques. The techniques involve finding the frequency of words, scoring the sentences, ranking the sentences etc. The summary is obtained by selecting a particular number of sentences (specified by the user) from the top of the list. It operates on a single document (but can be made to work on multiple documents by choosing proper algorithms for integration) and provides a summary of the document. The size of the summary can be specified by the user when invoking the tool.

This paper presents a new statistical approach to automatic summarization based on the Kernel of the source text. The Kernel-based system, called KernelSum (KERNEL SUMMarizer), uses the Kernel as a guideline to identify and select text segments to include in the final extract. Automatically produced extracts have been evaluated under the light of Kernel preservation and textuality.

Index Terms- Auto-summarization, KernelSum, KERNEL SUMMarizer, Statistical summarizers, Statistical summarizers.

I. INTRODUCTION

Auto-summarization is a technique used to generate summaries of electronic documents. This has some applications like summarizing the search-engine results, providing briefs of big documents that do not have an abstract etc. There are two categories of summarizers, linguistic and statistical. Linguistic summarizers use knowledge about the languange (syntax/semantics/usage etc) to summarize a document. Statistical ones operate by finding the important sentences using statistical methods (like frequency of a particular word etc). Statistical summarizers normally do not use any linguistic information.

Pre-processing interfaces are there to handle the following document types: Plain Text, HTML, Word Document. Using simple statistical measures, Kernel is identified as the most important passage of the source text, conveyed by just one sentence. It serves, then, as the guideline to identify and select other sentences to compose the final extract. Those are added to the extract provided that they satisfy summarization requirements, namely, Kernel preservation, textuality, relevance, and compression constraints.

II. FUNCTIONAL COMPONENTS OF THE TOOL

Following is a list of the functional components of the tool.

1. **Text pre-processor**. This will work on the HTML or Word Documents and convert them to plain text for processing by the rest of the system.
2. **Sentence separator**. This goes through the document and separates the sentences based on some rules (like a sentence ending is determined by a dot and a space etc). Any other appropriate criteria might also be added to separate the sentences.
3. **Word separator**. This separates the words based on some criteria (like a space denotes the end of a word etc).
4. **Stop-words eliminator**. This eliminates the regular English words like 'a, an, the, of, from..' etc for further processing. These words are known as 'stop-words'. A list of applicable stop-words for English is available on the Internet.
5. **Word-frequency calculator**. This calculates the number of times a word appears in the document (stop-words have been eliminated earlier itself and will not figure in this calculation) and also the number of sentences that word appears in the document. For example, the word 'Unix' may appear a total of 100 times in a document, and in 80 sentences. (Some sentences might have more than one occurrence of the word). Some min-max thresholds can be set for the frequencies (the thresholds to be determined by trial-and-error)
6. **Scoring algorithm**. This algorithm determines the score of each sentence. Several possibilities exist. The score can be made to be proportional to the sum of frequencies of the different words comprising the sentence (ie, if a sentence has 3 words A, B and C, then the score is proportional the sum of how many times A, B and C have occurred in the document). The score can also be made to be inversely proportional to the number of sentences in which the words in the sentence appear in the document. Likewise, many such heuristic rules can be applied to score the sentences.
7. **Ranking**. The sentences will be ranked according to the scores. Any other criteria like the position of a sentence in the document can be used to control the ranking. For example, even though the scores are high, we would not put consecutive sentences together.
8. **Summarizing**. Based on the user input on the size of the summary, the sentences will be picked from the ranked list and concatenated. The resulting summary

file could be stored with a name like <originalfilename>_summary.txt.

9. **User Interface**. The tool could use a GUI or a plain command-line interface. In either case, it should have easy and intuitive ways of getting the input from the user (the document, the size of the summary needed).

III. Evaluation

An ongoing issue in this field is that of evaluation. Evaluation techniques fall into intrinsic and extrinsic, inter-texual and intra-texual. An intrinsic evaluation tests the summarization system in of itself while an extrinsic evaluation tests the summarization based on how it affects the completion of some other task. Intrinsic evaluations have assessed mainly the coherence and informativeness of summaries. Extrinsic evaluations, on the other hand, have tested the impact of summarization on tasks like relevance assessment, reading comprehension, etc. Intra-texual methods assess the output of a specific summarization system, and the inter-texual ones focus on contrastive analysis of outputs of several summarization systems. Human judgement often has wide variance on what is considered a "good" summary, which means that making the evaluation process automatic is particularly difficult. Manual evaluation can be used, but this is both time and labor intensive as it requires humans to read not only the summaries but also the source documents. Other issues are those concerning coherence and coverage. One of the metrics used in NIST's annual Document Understanding Conferences, in which research groups submit their systems for both summarization and translation tasks, is the ROUGE metric (Recall-Oriented Understudy for Gisting Evaluation). It essentially calculates n-gram overlaps between automatically generated summaries and previously-written human summaries. A high level of overlap should indicate a high level of shared concepts between the two summaries. Note that overlap metrics like this are unable to provide any feedback on a summary's coherence. Anaphor resolution remains another problem yet to be fully solved. Evaluating summaries, either manually or automatically, is a hard task. The main difficulty in evaluation comes from the impossibility of building a fair gold-standard against which the results of the systems can be compared. Furthermore, it is also very hard to determine what a correct summary is, because there is always the possibility of a system to generate a good summary that is quite different from any human summary used as an approximation to the correct output .

IV. Kernelism Description

By making KernelSum Kernel-based, we assume it emulates human summarization in that, when a person summarizes a text, s/he first tries to identify the Kernel and, then, adds information drawn from the text to complement it. Considering that the amount of complementary information to appear in the extract depends on how long the extract is intended to be2, extraction is based, thus, on two parameters: the Kernel, which triggers the process, and the intended compression rate of the corresponding extracts. Kernel can be determined through either the Keywords or the Text Mining method. Its determination itself is very simple: a) based on the former method, Kernel is calculated on the basis of a list of keywords of the source text, considering a threshold of word significance; b) based on the latter, it is the result of the measurement of the representativeness of intra- and interparagraphs sentences. In both cases just one sentence is assigned to Kernel: in the former, it is the sentence that corresponds to the most significant distribution of keywords; in the latter, it is that whose frequency distinguishes it as the most representative of the source text, similarly to the way a topic or a search phrase is derived.

KernelSum Processes

KernelSum comprises three processes, namely, segmentation, sentence ranking, and extract production. Segmentation addresses sentences as minimal units. After delimiting them, sentence ranking proceeds to Kernel determination through the selected ranking method (either Keywords or Text Mining). Hereafter, we will refer to KernelKey to signal the use of the Keywords method by KernelSum; otherwise, as KernelTFISF, after the measure TF-ISF (Term Frequency – Inverse Sentence Frequency). Besides indicating the Kernel sentence, sentence ranking also classifies the other sentences to identify those that will appear in the extract. In this stage, for a more accurate calculation KernelSum makes use of the following sub-processes: stopwords removal, stemming and case folding. Extract production finally identifies sentences to include in the final extract that satisfy

(1) Kernel correlation, (2) relevance and (3) compression rate constraints. In what follows, sentence ranking and extract production are detailed and exemplified for the sample text shown in Figure 1, whose sentence segments are numbered. This has been extracted from a corpus of scientific texts in Computer Science.

V. Sentence Features

In our current system, every sentence ***s*** is represented by five normalized features:

• **Location of the Paragraph (P):**

P = *Y* / *M* (1)

where ***M*** is the total number of paragraphs in a document; ***Y*** is the index of the paragraph ***s*** belongs to.

• **Location of the Sentence (S):**

S = *X* / *N* (2)

where ***N*** is the total number of sentences in the paragraph; ***X*** is the index of sentence s.

• **Length of the Sentence (L):**

The length of the sentence is the number of words it contained, i.e., ***l(s)***, normalized by Sigmoid function:

$$L = \frac{1-e^{-\propto}}{1+e^{-\propto}}, \qquad \propto = \frac{l(s)-\mu(l(s))}{std(l(s))}$$

Where ***u(l(s))*** is the average length of sentences, and ***std(l(s))*** is the standard deviation of the sentence lengths.

• **Heading Sentence (H):**

H = 1, if ***s*** is a title, subtitle or heading, 0 otherwise.

• **Content-word Frequencies (F):**

$$F(s) = \frac{1-e^{-\propto}}{1+e^{-\propto}}, \qquad \propto = \frac{CW(s)-\mu(CW(s))}{std(CW(s))} \qquad (4)$$

$$CW(s)(x+a)^n = \sum_{i-1}^{k} \log[\mathrm{Freq}(\mathrm{w_i})], \qquad w_i \in S \qquad (5)$$

where ***Freq(wi)*** is the frequency of ***wi*** in that document; ***μ(CW(S))*** is the mean of all the sentence scores, and ***std(CW(s))*** is the standard deviation.

A. Sentence ranking function

We assume that for a certain type of documents, the mechanism to perform summarization would be the same. Therefore, we only need to find one algorithm that links a collection of documents and their corresponding summaries. We process the text summarization learning task in two stages: training and testing. In the training stage, a set of training documents with their summaries are provided, and the text features are preprocessed using statistical methods and natural language processing methods as defined , then each sentence in a document is scored based on a sentence ranking function constructed by GEP. Fitness value of the summarization task is the similarity between the summary produced by the machine and the summarization text of training document. The top ***n*** ranked sentences will be returned as the summary of that document and presented in their nature order. In the testing stage, a different document set is supplied to test the similarity between the machine summarized text and the human or other system summarized text.

Table 1 shows the sentence scores for the sample text shown in Figure 1. As it can be seen, the Keywords method signals sentence 4 as the Kernel sentence, but the TF-ISF one pinpoints sentence 3 (the highest-scored ones). By thoroughly reading the sample text and comparing it with such choices, we confirm that the former method identifies more clearly the Kernel sentence, since it mirrors the main idea more properly than sentence 3.
Indeed, if we discourse-analyze the sample text based on the RST Theory , for example, sentence 3 still refers to background information.

Table 1 – Sentences scores

Sentence	Keywords	TF-ISF
1	24	0.465
2	22	0,628
3	23	0,671
4	42	0,598
5	22	0,643
6	37	0,663
7	17	0,571
8	25	0,575

So, it could not be the Kernel sentence. Alternatively, the main topic refers to a tool to help nonnative English speakers. So, sentence 4 suffices well such a role3, being pretty satisfactory as the Kernel of the text. Corroborating this, we can also acknowledge that the remaining sentences 5-8 just add further details on the tool itself. Having determined the Kernel sentence, KernelSum can now proceed in selecting complementary sentences to build the corresponding extract.

B. Extract production

To build the extract, KernelSum executes the following steps:

- It averages the sentence scores, determining their threshold;
- Besides the Kernel sentence, KernelSum selects others that both:
  - ➢ Contain at least one word whose stem also corresponds to some word in the Kernel sentence (i.e., it assures that lexical cohesion be observed);
  - ➢ Have scores above the threshold (i.e., it guarantees that only relevant sentences to
    Kernel will be chosen).

The above steps are also constrained to satisfying the compression rate. If this is too strict that only the Kernel sentence satisfies it, the extract will be mono-sentential and as informative as its Kernel sentence (step 2 will thus be excluded). So, there is a compromise between informativeness and compression that will even delineate if Kernel can be complemented or not. Clearly, step (2a) is responsible for the distinctive idea underlying KernelSum when compared with other extractive approaches: it is this step that is intended to prove Hypothesis (II). This is addressed in Section 3 along with the proof of Hypothesis (I).

Figures 2 and 3 show extracts of the sample text for a 60% compression rate, respectively referring to KernelKey and KernelTFISF methods. It is possible to notice that the first extract conveys the Kernel, while the second one does not. Besides not resolving anaphors, KernelSum also does not.

VI. EVALUATING KERNELISM PERFORMANCE

Our evaluation of KernelSum has been carried out aiming at two distinctive goals: 1) to see how effective the proposed ranking methods are in identifying the Kernel sentence of a source text, i.e., to certify Hypothesis (I) is pertinent; 2) to assess KernelSum performance as such, by means of focusing on the quality of the generated extracts, i.e., to certify Hypothesis (II) is attainable.

English is the dominant language in the writing and publishing of scientific research in the

form of scientific articles. In order to ease these users' problems, we developed a learning

environment for scientific writing named AMADEUS (Amiable Article Development for

User Support). The main goal of this research is to implement AMADEUS as an agentbased

architecture with collaborative agents communicating with a special agent embodying

a dynamic user model. We also provide details about intelligent agents which were

used to implement the user model for the AMADEUS environment.

Figure 2 – KernelKey-based Extract 1

However, many non-natives users of English suffer the interference of their mother tongues when writing scientific papers in English. These users face problems concerning rules of grammar and style, and/or feel unable to generate standard expressions and clauses, and the longer linguistic compositions which are conventional in this genre.

Figure 3 – KernelTFISF-based Extract 2

Experiment 1: evaluating the extracts overall quality

A different test corpus has been used in this experiment, composed of 20 newspaper texts in English, from the WSJ financial section (c.a. 410 words). For each text, 2 extracts were automatically generated considering also the 60% compression rate. We had 12 human judges reading the source texts and scoring their extracts based on two decision points:

Table 2 – Possible scores for quality

Kernel	Textuality	Score
Preserved	Assured	9
Preserved	Partially Assured	8
Preserved	Not Assured	7
Partially Preserved	Assured	6
Partially Preserved	Partially Assured	5
Partially Preserved	Not Assured	4
Not Preserved	Assured	3
Not Preserved	Partially Assured	2
Not Preserved	Not Assured	1

Kernel preservation and textuality (see Table 2). Textuality, here, is the property of a text being both cohesive and coherent.

Table 2 shows the distribution of the extracts according to the means, i.e., the scores average . Although only 55% of the ones generated through GistKey are above the means, this still outperforms GistTFISF. Table 4 shows the figures for gist and textuality. When using GistKey, 90% of the extracts have been judged to totally or partially preserve the gist. Textuality was also highly graded: 85% of them were totally or partially coherent and cohesive.

Table 3 – Means distribution

Methods	Means	
	Above	In It
KernelKey	55%	14%
KernelTFISF	39%	10%

Again, GistKey significantly outperformed GistTFISF.

Table 4 – Extracts quality

Methods	Points			
	Kernel Preservation		Textuality	
	Total	Partial	Total	Partial
KernelKey	50%	40%	50%	35%
KernelTFISF	10%	20%	5%	25%

VII. CONCLUSION

KernelSum has been devised to produce extracts from texts of any domain, genre, and natural language, provided that the corresponding NL resources (i.e., stopwords repository and stemmer) are assembled into it. So far, we have explored it for Brazilian Portuguese and English. We have chosen two simple statistical methods in order to determine a Kernel sentence, which is the backbone of text extraction, for two reasons: they are easy to implement and they do not demand complex and sophisticated linguistic resources. So, they seemed appealing for us to verify and compare their effectiveness in determining the Kernel sentence. The experiments described here made evident that the correct determination of the Kernel sentence usually influences the quality of the related extracts. Moreover, they show that Kernel conveys better the content of the corresponding source text when it is computed through KernelKey, instead of KernelTFISF. Two conclusions can be withdrawn from this: the Kernel identification method based on the keywords distribution performs better than that based on the inverse distribution of sentences in the source text, for the test corpus adopted so far. However, further investigations should explore more deeply such a difference, for other text genres and domains and for more significant corpora.

Although many authors have stressed the need to convey the main idea and to warrant the textuality of the results in automatic summarization , KernelSum is novel because of both the way Kernel is determined and used as a guide for extraction (through lexical cohesion): by observing the words co-occurrence, the extracts are more likely to be coherent; by including Kernel, they are more likely to convey well the main idea of their source texts.

REFERENCES

[1] Zhang, J., Loh, H.T., and Liu, Y. (2009). 'Gather Customer Concerns on Online Product Reviews– A Text Summarization Approach', Elsevier Expert Systems with Applications, pp. 2107- 2115.

[2] Fiszman, M., Demner-Fushman, D., Kilicoglu, H., and Rindflesch, T.C. (2009). 'Automatic Summarization of MEDLINE citations for Evidence-based Medical Treatment: A Topic-Oriented Evaluation', Journal of Biomedical Informatics, Vol. 42, No.5, pp.801-813.

[3] Verma, R., Chen, P., and Lu, W. (2007). 'A Semantic Free-Text Summarization Systems Using Ontology Knowledge', Document Understanding Conference DUC 2007, pp. 1-5.

[4] Pardo, T.A.S. and Rino, L.H.M.: DMSumm: Review and Assessment. In E. Ranchhod and N.J. Mamede (eds.), Advances in Natural Language Processing, pp. 263-273 (Lecture Notes in Artificial Intelligence 2389). Springer- Verlag, Germany (2002).

[5] Mani, I.: Automatic Summarization. John Benjamins Publishing Co., Amsterdam (2001).

[6] Feltrim, V.D., Nunes, M.G.V., Aluísio, S.M.: Um corpus de textos científicos em Português para a análise da Estrutura Esquemática. Série de Relatórios do NILC. NILC-TR-01-4 (2001).

[7] Radev, D.R., Jing, H., Budzikowska, M.: Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In the Proceedings of the ANLP/NAACL Workshop on Summarization. Seattle, WA (2000).

[8] Larocca Neto, J., Santos, A.D., Kaestner, A.A., Freitas, A.A.: Generating Text Summaries through the Relative Importance of Topics. In M.C. Monard and J.S. Sichman (eds.), Lecture Notes in Artificial Intelligence, No. 1952, pp. 300-309. Springer-Verlag (2000).

[9] White, J.S., Doyon, J.B., Talbott, S.W.: Task Tolerance of MT Output in Integrated Text Processes. In ANLP/NAACL 2000: Embedded Machine Translation Systems, pp. 9-16. Seattle,WA (2000).

[10] Teufel, S. and Moens, M.: Argumentative classification of extracted sentences as a first step towards flexible abstracting. In I. Mani and M. Maybury (eds.), Advances in automatic text summarization, MIT Press (1999).

[11] Barzilay, R. and Elhadad, M.: Using lexical chains for text summarization. In the Proceedings of the ACL-97/EACL97 Workshop on Intelligent Scalable Text Summarization. Madrid, Spain (1998).

[12] Hovy, E.H. and Lin, C.Y.: Automated Text Summarization in SUMMARIST. In M. Maybury and I. Mani (eds), Advances in Automatic Text Summarization. Cambridge, (1998).

[13] Rino, L.H.M.: Modelagem de Discurso para o Tratamento da Concisão e Preservação da Idéia Central na Geração de Textos. Tese de Doutorado. IFSC-USP. São Carlos – SP (1996).

[14] Aretoulaki, M.: COSY-MATS: A Hybrid Connectionist- Symbolic Approach To The Pragmatic Analysis of Texts For Their Automatic Summarisation. PhD. Thesis. University of Manchester (1996).

[15] Sparck Jones, K. and Galliers, J. R.: Evaluating Natural Language Processing Systems. Lecture Notes in Artificial Intelligence, Vol. 1083 (1996).

[16] Ono, K., Sumita, K., Miike, S.: Abstract Generation based on Rhetorical Structure Extraction. In the Proceedings of the 15th International Conference on Computational Linguistics – COLING'94, Vol. 1, pp. 344-348. Kyoto, Japan (1994).

[17] Witten, I.H., Moffat, A., Bell, T.C.: Managing Gigabytes. Van Nostrand Reinhold. New York (1994).

[18] Sparck Jones, K.: Discourse Modelling for Automatic Summarising. Technical Report No. 290. University of Cambridge (1993).

[19] Morris, J. and Hirst, G.: Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text Computational Linguistics, Vol. 17, No. 1 (1991).

[20] Hoey, M.: Patterns of Lexis in Text. Oxford University Press (1991).

[21] Salton, G.: Automatic Text Processing. The Transformation, Analysis and Retrieval of Information by Computer. Addison- Wesley (1989).

AUTHORS

First Author: Vikrant Gupta, M.C.A., Research Scholar, Mewar University. Email id - gupta.vikrant14@gmail.com

Second Author: Ms. Priya Chauhan, M.C.A., Senior Lecturer, MCA Deptt., RKGIT, Ghaziabad
Email id - chauhan.priya15@gmail.com

Third Author: Dr. Sohan Garg, Ph.D., Principal MCA, IIMT, Meerut, Email id - sohangarg@rediffmail.com