Highlight Extraction from Scientific Articles

SUBMITTED IN PARTIAL FULFILLMENT FOR THE
DEGREE OF MASTER OF SCIENCE

NISHANT MINTRI
11635274

MASTER INFORMATION STUDIES : DATA SCIENCE TRACK
FACULTY OF SCIENCE
UNIVERSITY OF AMSTERDAM

2018-06-28

	Internal Supervisor	**External Supervisor**
Title, Name	Dr. Evangelos Kanoulas	Dr. Georgios Tsatsaronis
Affiliation	UvA, ILPS	Elsevier
Email	e.kanoulas@uva.nl	g.tsatsaronis@elsevier.com

Highlight Extraction from Scientific Articles

Nishant Mintri
University of Amsterdam
Amsterdam
nishantmintri@gmail.com

ABSTRACT

With the plethora of information on the web, users usually waste a lot of time delving deep into articles only to frustratingly realize later that the article was actually not what they were looking for. This is usually the case when searching for scientific papers, with multiple papers on the same topics, but each paper focusing on a more specific aspect of the topic. Summaries of the papers, which highlight the salient aspects of it would thus help the user decide if the paper is relevant to his search needs or not without diving deep into the paper. In this thesis with the support of *Elsevier*, we investigate the extent to which we can use different classification methods to automatically generate highlights for a rather noisy data set consisting of scientific articles present on their research platform *ScienceDirect*. We introduce novel methods of creating training datasets for the purpose of summarization. Our results indicate that we can successfully apply classification methods to identify most of the important sentences as highlights from scientific articles.

KEYWORDS

Highlights, scientific articles, Classifier, ROUGE, extractive summarization

1 INTRODUCTION

ScienceDirect[1], a major Elsevier platform for searching and accessing scientific content, has been one of the first platforms to offer the ability to its users to browse and search the highlights of scientific articles. *Highlights* of an article refers to a set of statements that best describe the salient aspects of the article. Therefore these highlights, often contain the most important outcomes of the reported work, and traditionally have been provided during the submission of the final article version by the authors.

Highlights of an article, thus help in creating a condensed summary of the original article. This process of condensing and transforming an article into a shortened form is referred to as *summarization.*

Generating a condensed summary from an article has been one of the main fields of research in Natural Language Processing (NLP) for a long time, beginning in the late 1950s [15]. A considerable number of strategies and methodologies have been produced in this field of research [23]. The approaches for summarization can broadly be classified into two categories:

- **Extractive Summarization**: In this technique, summaries are produced by selecting or copying few sentences exactly as they appear in the article to form the summary.
- **Abstractive Summarization**: The abstractive summarization techniques focus on Natural Language Generation to generate phrases and sentences that best describe the original content of the article.

[1]http://www.sciencedirect.com

Summarization can be performed on either a single document or multiple documents dealing with similar topics [8]. The two vital parts of automatic text summarization that need to be addressed are:

- How to select the most important and informative parts from a document [36]?
- How to express the chosen content in a condensed way [35]?

This thesis focuses on extractive summarization approaches to select sentences from scientific articles that could possibly form highlights for the given article. We formulate the problem as a binary classification problem and compare the performance of different classifiers on a rather noisy dataset of full scientific articles.

The rest of the document is divided as follows. Section 2 outlines the research questions that we aim to answer through this thesis. Section 3 formulates the problem of automatic summarization on the scientific articles. Section 4 elaborates on the work done previously on extractive summarization. Section 5 discusses the preparation of the dataset and choice of sentence features. Section 6 introduces the baselines that we compare our systems performance with. Section 7 focuses on the metrics used for evaluation of the summaries. Section 8 describes the models we used and Section 9 provides a detailed discussion on the research question and the experiments performed. Section 10 looks at an example article where our system performs poorly and Section 11 lists out the improvements that can be incorporated in the future to enhance our models.

2 RESEARCH QUESTIONS

In this thesis we seek to answer the following questions:

- To what extent can extractive summarization techniques be used on single scientific articles to highlight and select important sentences from them?
  - How do different classifiers perform on a noisy dataset for extractive summarization of scientific articles?
  - How do supervised machine learning algorithms perform compared to unsupervised algorithms for the task of extractive summarization from scientific articles?
  - Which features of a sentence (including linguistic structure) are helpful in identifying the sentence as a 'significant highlight' towards a scientific article?
  - How do the generated summaries vary when using these techniques on the whole article as compared to using them on specific sections inside these well structured scientific articles?

- How do we generate a labelled dataset from an article that could be used in training the classifiers to perform supervised extractive summarization?
- What is the optimum number of sentences that should be contained in the generated summary?

3 PROBLEM FORMULATION

For the supervised learning algorithms developed, the task of extractive summarization is formulated as a binary classification problem. While training, we train our classifiers on the positive and negative examples as described in Section 5.3. During testing, each sentence in an unseen article is classified either as a positive (assigned value 1) or a negative highlight sentence (assigned value 0) with a probability for this classification. We sort and rank the classified positive sentences according to these probability values and select the top 'k' sentences to form the summary as described in Section 9.5. Formally, for a given document D, represented by a collection of sentences

$$D = \{(x_1, y_1), (x_2, y_2), (x_3, y_3)..., (x_n, y_n)\}$$
$$x_i = \{f_1, f_2, f_3..., f_m\}, 1 \leqslant i \leqslant n$$
$$y \in (0, 1)$$

where:

x_i represents the i^{th} sentence in document D
y_i represents the corresponding label of the i^{th} sentence
n is the number of sentences in D
f_j is the j^{th} feature in sentence x_i as described in Section 5.5
m is the total number of features representing a sentence

The task is to learn a function H, that produces a mapping between the features of the sentence and the probability of the sentence having a label 1.

$$H(x_i) = probability(y_i = 1), 1 \leqslant i \leqslant n$$
$$X^* = \underset{probability(y)}{\text{argmax}} \quad H(x_i)$$
$$S = \{x_1^*, x_2^* ... x_k^*\}, 1 \leqslant k \leqslant n$$
$$X_k^* \subset X$$

where:

k is the number of sentences in the generated summary S.
X_k is the k^{th} sentence in the generated summary S.

4 LITERATURE SURVEY

State of the art techniques apply machine learning concepts modeling the problem of summarization as a classification problem. This problem can be handled in both a supervised and unsupervised setting. In the supervised setting we have access to the highlights associated with the articles. If these highlight sentences occur in exactly the same way as they occur in the article, they can be labelled as positive examples. However, this is usually not the case. Thus to form labelled examples for the classifier to learn from and train the following approach is used. Each sentence in the highlight is compared to every other sentence in the article using a *similarity metric*. A score for every highlight-sentence pair is generated. For all the highly scored pairs (scores above a certain threshold), the corresponding sentences in the article are labelled as positive highlight sentences. All other sentences are marked as negative highlight sentences. These sentences and their labels are then used to train machine learning algorithms to predict which sentences should be included in the summary. Neural networks [26], Naive Bayes classification [11], Decision Trees [31], Probabilistic Support Vector Machines [11] are some of the supervised algorithms that are applied.

In the unsupervised methods, graph based [10] extractive summarization techniques [9], clustering [18], Hidden Markov Models [6] are used. In the graph based technique sentences are represented as nodes. Edges between the nodes indicate how similar sentences are. The importance of the nodes is estimated by a variation of the traditional PageRank [4] ranking algorithm. Other approaches in summarization include using evolutionary algorithms like genetic algorithms [26], Integer Linear Programming [37]. After the recent advancement of Deep Learning [25] and Deep Neural Networks [14], the focus has rather shifted to having an encoder-decoder structure using RNNs and LSTMs to generate abstractive summaries [33] [22] [17]. However, in this thesis we limit our focus to using classification models to generate extractive summaries. An overview of the related techniques is as follows:

- **LexRank** [10] It is an unsupervised graph based ranking algorithm. Every sentence in an article is represented by a node while the edges represent the similarity between these sentences. They measured this similarity by using the TF-IDF scores while considering the sentences as a Bag of words (BOW) model. A similarity matrix is built using this measure. They then compute the PageRank [4] centrality measure for each sentence in the article. Sentences with the highest scores are ideal candidates for forming the summary of the article.
- **TextRank** [24] This algorithm is very similar to the LexRank algorithm [10]. The major difference is in the way the similarity matrix is created. In the original paper [24], the authors use a count of the number of common words between two sentences normalized by the length of the sentences under consideration. Different variations of the similarity function [2] have been used over the years. Both these approcahes are based on the relative importance of senteces. The similarity scores can be used directly as weights on the edges, or edges could be formed only between sentences having a similarity score greater than a threshold value.
- **SummaRuNNer (SR)** [27] In this recent paper by Nallapati and et al. a RNN based sequence classifier model is proposed for extractive summarization. Each sentence is visited sequentially depending on the order they appear in the document. A binary decision is made of whether to include the sentence in the summary or not taking into account previous decisions made. The paper also proposes the idea of creating extractive summary labels for sentences that are needed for training the model as described earlier in this section.

5 DATASET DESCRIPTION

5.1 Data Gathering and Preprocessing

The dataset used in this thesis has been downloaded from ScienceDirect[2]. The scientific papers are downloaded in a compressed 'zip'

[2]http://www.sciencedirect.com

format according to the branch of science that it deals with. Individual scientific papers are stored as XML files in the downloaded 'zip' files. The XML files are named according to a unique Publisher Item Identifier (PII) number. These files contain both the metadata and the entire content of the scientific article.

Scientific articles published in journals have a somewhat common outline to them [34] i.e, they usually have an abstract, followed by an introduction to the problem, any related work, author's approach, results and conclusions. However, the internal structure and contents of these articles vary a lot depending on many factors. Some of the major factors are the unique individual writing style of the authors, the field of science the article belongs to and the journal that it is published in. There are no restrictions on the number of sections or subsections an article can have. The authors thus have complete freedom on maintaining the internal contents and structure of the article as per their needs. The major part in the preprocessing step is to identify these sections and its associated content.

A JAVA based XML parser is designed to parse and extract the data from the XML documents whenever the document contains the 'HIGHLIGHTS' section. This is done so that we can compare and evaluate the performance of the highlights generated by our models to the gold set of highlights. The XML parser creates a text (.txt) file corresponding to each scientific article. Each line in the generated text file corresponds to the contents of a section within the article. For easy differentiation between the section and its contents an unique separator is used as a delimiter. '#&#' is used as the delimiter as this sequence of characters is almost never encountered within the text of the scientific articles. The format of each line is thus as follows:

'Separator' **Section-Title** *'Separator'* **Text content**
Example: *#&#* **Introduction***#&#* **This is an introductory line.**

The data within each section is tokenized into sentences such that the contents of each section are represented by a list of sentences that appear in that particular section.

Contents from many sections of the article like 'References', 'Acknowledgement', 'Sources of Funding', 'Disclaimer', etc., are not taken into consideration, as the content from these sections do not contribute to the highlights of the article. These articles also contain graphs, tables and figures embedded in them. These do not directly contribute to the summary that we intend to generate. Hence, these need to be filtered out from the XML file before the generation of the text files.

5.2 Preprocessing Steps

- **Tokenization**: The scientific articles are first tokenized into sentences using the NLTK tokenizer[21]. For the tokenizer to correctly tokenize the sentences words like 'Fig.', 'Table.', 'et al.', 'Equ.' are added as exceptions while tokenizing. Each sentence is then converted to lower case.
- **Stop word removal**: Functional words like 'a', 'an', 'the', 'is' which do not provide much information are removed from the sentences.
- **Stemming**: Porter stemmer [30] is used on the tokenized sentences to convert each word in the sentences to their corresponding root/stem words.

5.3 Preparing Training Set

While preparing the training set, two problems associated with the highlight sentence set provided by the authors when submitting the article were identified. Firstly, when the authors write these highlights, they usually do not copy the sentences as it is from the article itself. Instead, they 'generate the sentences' which summarize the article according to them. Hence the highlight sentences cannot be directly considered as positive examples while training our classifiers. Secondly, the number of highlight sentences for the articles are very small when compared to the length of the entire article. On average the dataset contains **4** highlight sentences per article while the article contains an average of **375** sentences. If we only use these highlights directly as examples the resulting dataset would be very skewed. To mitigate these problems the following approach is taken into consideration. We use **ROUGE-L** [19] as a similarity measure to compute a score between each sentence in the article and sentences in the highlight set. We rank the sentences in the article according to these scores and select the top 'n' sentences as positive examples of sentences which should be considered as highlight sentences from the article. Conversely, the bottom 'n' sentences are taken as negative examples of sentences which should not be considered as highlight sentences. This results in an equally sized positive and negative set of sentence examples for each article. Different values of 'n' are experimented with while selecting the positive and negative examples, to see the effect it has on the generated summary as described in Section 9.7. If the length of the article is shorter than '2n' sentences, we skip the article and do not use it in the algorithms.

For each of the articles, a json file is created from the corresponding text files. Each json file contains a list of '2n' sentences and their extracted corresponding features as described in Section 5.5. The gold highlight set of the article is also stored in the json file. The format of the json file is as shown in Table 3. The entire processing pipeline is shown in Figure 1.

Finally, all the json files are merged into a larger json file which contains the sentences and their associated features from all the articles and is used in training the classifiers.

5.4 Preparing Test Set

As described in Section 5.1 a text file is created for each article from the XML file. The contents of the text file are then tokenized into a list of sentences. The section name and the position of the sentence are appended to each sentence in the list. For each of the sentence we create its corresponding features as described in Section 5.5. These features are the inputs to our binary classifier which predicts whether the sentence should be a highlight sentence or not.

5.5 Sentence Features

For each sentence in the articles the following features are calculated:

- **Sentence Location:** Absolute position of sentence within the article and within a section. Sentences at the beginning

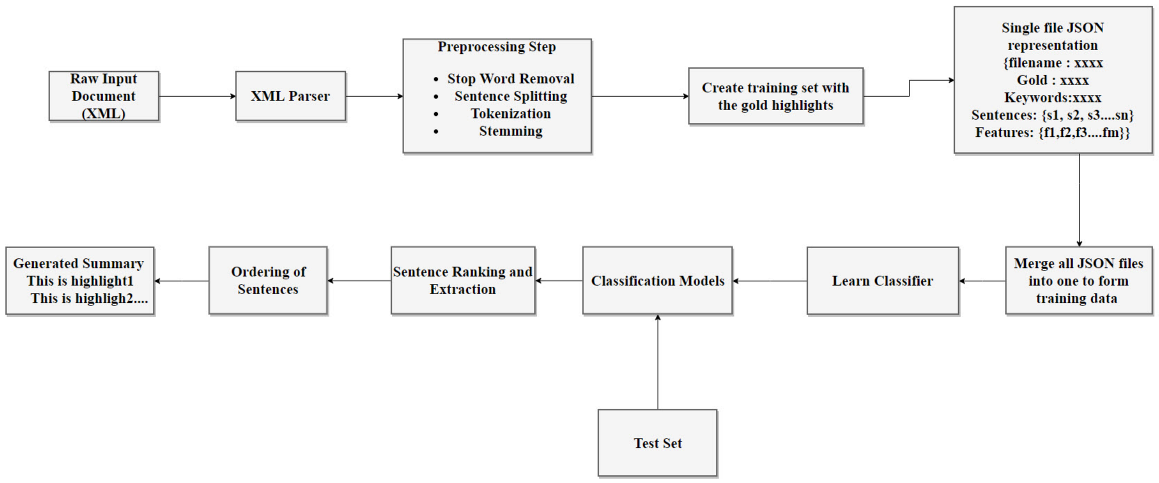


Figure 1: Data processing pipeline

and end of sections are usually considered to be more informative than the rest of the sentences.

- **Numeric Data Feature:** Scientific articles usually report their performance and results in the article to compare how their approaches compare to other approaches. These results usually have a numerical element associated with it. Hence, it could be an important feature to identify a highlight sentence.
- **Term Frequency Score:** We sum up the term frequency (tf) and inverse document frequency (idf) [32] values of all words in the sentence.
- **Title Word Similarity:** A sentence is considered more informative if it is similar to the title of the article than a sentence which is dissimilar. The similarity is measured both by taking a cosine similarity match between the sentence and title and also by the count of the occurrence of overlapping words.
- **Document TF-IDF:** This feature is calculated similar to the Term Frequency score. The term frequency is replaced by the count of the words in the sentence and the document frequency by the total words in the article.
- **Section Location:** The importance of a sentence usually depends on the section it is a part of. A simple hypothesis is that sentences in the results are more important than sentences in the acknowledgements.
- **Sentence Length:** Short sentences usually are not considered good highlight sentences as they do not convey a lot of information to the reader.
- **Sentence To Abstract Similarity:** The way an abstract is written, it is often considered as an overall summary of the article. If a sentence is similar to the abstract it is more likely to be considered as a highlight sentence over a one which is quite dissimilar.
- **Keyword Match:** The articles in our collection have a section which list the keywords of the article. These keywords are often a list of 6-8 words that tell us what are the major points that the article discusses. A sentence which covers many keywords should thus be considered as a more informative sentence.
- **Noun Phrases:** A sentence involving a large number of noun phrases, is considered more informative than sentences which do not contain any noun phrases. A POS tagger is used on the individual sentences to count and extract the different parts of speech for the words in the sentence.
- **Performance of sentences from other summarization algorithms:** We apply already developed summarization algorithms e.g LexRank [10] , TextRank [24], Latent Semantic Analysis (LSA) summarizer [28], Luhn Summarizer [15] and SummaRuNNer [27] directly on the article. For each of these techniques, we score the sentences according to the algorithms as discussed in Section 4 and store the scores obtained by each sentence.

6 BASELINES

We consider the following baseline systems and use them to compare the performance of our system.

- **Previously published algorithms:** The following algorithms TextRank [24], Luhn Summarizer [15], LexRank [10] and a more recent algorithm SummaRuNNer [27] as described under the Section 4 form three baseline systems.
- **Heuristic Rule Based Approach:** In this approach, we form rules to score the different features from Section 5.5. These rules are manually created. An example of such a rule could be as shown in Algorithm 1, where $S_{Position}$ represents the position of the sentence with respect to the section

Algorithm 1: Heuristic ranking approach

```
1 if $S_{Position}$ = $1^{st}$ or $2^{nd}$ Sentence in Section then
2     score ← 2.0 ;
3 else
4     score ← 0.75 ;
```

it belongs to. Different weights are assigned to the different features according to their importance, while computing the final score of the sentence. The top ‘k’ sentences with the highest scores are selected to form the summary.

- **Random Selection:** In this approach, we generate ‘k’ random numbers between 1 and the number of sentences in the article. We then select the corresponding sentence from the article and add it to the generated summary.
- **Beginning Sentence Selection:** In this approach, we generate the summary by selecting the first and last sentences from each of the sections in the article. The intuition being that the opening and ending sentences of sections, talk about the main idea behind writing the section.

7 EVALUATION METRICS

7.1 Evaluation of generated summaries

Evaluation of the extracted highlights is an interesting and challenging task in itself. The absence of a standard evaluation metric has made evaluation a difficult task. A good summary, is a subjective term and can vary both from domain to domain and also for the final audience of the summary [12]. It is not possible for humans to manually read the large amount of articles and the summaries to evaluate how good the generated summaries are.
Many evaluation metrics have been proposed over time, however we use the *Recall-Oriented Understudy for Gisty Evaluation* (ROUGE) [19] metric in our evaluation. Most of the summarization contests held by the Document Understanding Conference (DUC) usually use this measure for evaluation of the generated summaries. ROUGE metrics have a variety of different measures for the evaluation of the summaries. These metrics measure are a ratio of the count of the number of overlapping word sequences between the system generated and gold summary. These measures are as follows:

- **ROUGE-N:** This measure is used to compute the N-gram similarity between the generated and gold summary, where N stands for the length of the n-gram. Lin [20] recommends using ROUGE-1 and ROUGE-2 for the evaluation of single document summarization tasks.
- **ROUGE-L:** This metric is used to measure the Longest Common Subsequence (LCS) between two sentences. It is computed by dividing the length of the LCS between the sentences in the generated and gold summary by the length of the gold summary.
- **ROUGE-SU:** This metric is used to measure skip-bigram and unigrams between sentences. A skip-bigram refers to matching words which are non-consecutive and have arbitrary words between them. Thus any word pairs that are in the same sentence order for the generated and gold summaries constitute a valid bigram.

These metrics are commonly used to measure both the recall and precision between the generated and gold summaries. Recall is used to measure the ratio of the number of overlapping word sequences in the gold summary and the generated summary to the number of word sequences in the gold summary. Thus it quantifies the number of gold summary word sequences present in the generated summary.

$$ROUGE_r = \frac{\sum_{S \in S_m} \sum_{gram_n \in S} Number\ of\ matching(gram_n)}{\sum_{S \in S_m} \sum_{gram_n \in S} (gram_n)} \tag{1}$$

where S_m are the generated model summary sentences and and n represents the n-gram under consideration.

Precision on the other hand measures the number of overlapping word sequences in the gold and generated summaries to the number of word sequences in the generated summary. Thus it quantifies the number of generated summary word sequences that are present in the gold summary.

$$ROUGE_p = \frac{\sum_{S \in S_m} \sum_{gram_n \in S} Number\ of\ matching(gram_n)}{\sum_{S \in S_g} \sum_{gram_n \in S} (Totalgram_n)} \tag{2}$$

where S_g is the sentences in the gold summary.

Generated summaries which are very long, thus would have a very high recall score but a very low precision score. The F1 score is a single value which takes into account both the recall and precision scores and is computed as the harmonic mean of the recall and precision values.

$$F_1 = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \tag{3}$$

We report the average recall, precision and F1 values for the above mentioned three metrics for all the generated summaries. The evaluation is done by extending the ROUGE 2.0 package published by Ganesan et.al [13].

7.2 Evaluation of classifiers

For the different classifiers that we train to predict the highlight sentences in the paper, we measure the performance of the classifiers by measuring their recall, precision and F1 scores on the 100,000 sentences that constitute the test split. We use a 10 fold cross-validation technique.

8 MODELS

To train and test the summarizers developed, we used different classification algorithms on them. These classifiers were as follows:

- **Naive Bayes (NB):** It is based on the Bayes theorem assuming the independence between all the features. It is one of the simplest yet a very powerful classifier. It is very fast to train and thus can be effectively used for very large data sets. Scikit-learn [29] was used for the implementation of Gaussian Naive Bayes, which is a machine learning library for Python.
- **Decision Trees (DT) [31]:** It uses a tree like structure for the classification model as well as to make decisions. From the training data features, decision rules are made. The value

of the dependent variable is predicted using these decision rules [29].

- **Support Vector Machines (SVM) [7]:** SVMs are one of the most powerful supervised machine learning algorithms for classification. In this algorithm each data point is plotted in a n dimensional space, where n is the number of features. The goal of the classifier is to find a hyper-plane that separates and classifies the different classes.
- **Random Forest (RF) [3]:** It is an ensemble approach which is based on decision trees. It creates a forest with a number of decision trees in it. Random forests are usually not prone to over-fitting due to large number of decision trees.
- **Gradient Boosted Decision Trees (GBDT):** This algorithm is very similar to the random forest algorithm. However, unlike random forests, the trees are added sequentially in this case.
- **Logistic Regression (LR):** Logistic regression can be applied whenever we have a binary dependent variable.
- **K-Means Clustering [16]:** It is a unsupervised algorithm that tries to find 'k' clusters among the data, by separating the data into groups of equal variance. It scales well to large number of samples and has been used across wide range of areas.
- **Ensemble Model:** We use all the different models described above on our dataset. We follow both a hard voting and a soft voting approach to select highlight sentences. In the soft voting approach we sum up the probability of a sentence being a highlight sentence for each of the models and then normalize the result to get an overall ensembled probability. We then sort the sentences based on this ensembled probability score and select 'k' sentences from this list to form the highlight sentences. In the hard voting approach we count the number of times each sentence is selected by a model as a highlight sentence. We then select 'k' sentences with the maximum votes as the highlights.

9 EXPERIMENTS AND RESULTS

All models were trained on 14,472 papers belonging to the Computer Engineering domain. During the training phase, a 75:15:10 (Train - Test - Validation) split was used for the training of the classifiers. The highlights were generated for 250 papers belonging to the same domain that were not encountered during the training.

9.1 RQ1: Which classifier performs best for classifying highlight sentences in scientific articles?

We here analyze how the different classification algorithms of Section 8 perform on our rather noisy dataset of scientific articles. The classification performance for the different classifiers are shown in Table 1. The best performing classifier in terms of F1 score for the purpose of classification of sentences as a highlight sentence is the GBDT classifier.

To evaluate the quality of the summaries generated we compare the ROUGE [19] scores between the summaries generated by the different models and the gold highlight sets. The various ROUGE scores (after stop word removal from the generated highlights and gold summaries) are shown in Tables 2, 4 and 5. Figure 2 shows the change in the ROUGE-L scores.

Table 1: Performance of classifiers with a 10 fold CV

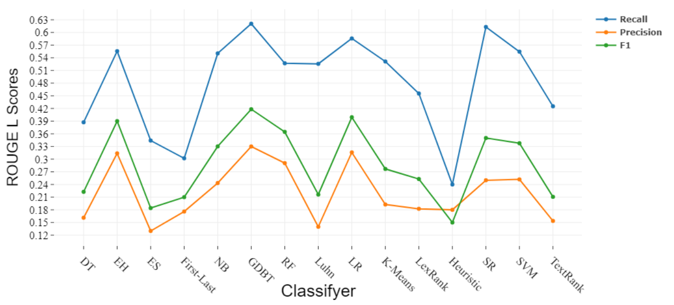
Model	Recall	Precision	F1 Score
Naive Bayes	0.973	0.663	0.788
Logistic Regression	0.828	0.782	0.805
GBDT[*]	**0.885**	**0.875**	**0.880**
SVM	0.877	0.873	0.875
Decision Tree	0.827	0.831	0.829
Random Forest	0.893	0.8389	0.846

[*] Indicates the best performing algorithm in terms of F1 score for classification.

The GBDT classifier achieves the best score for all the variations of ROUGE (Recall, Precision and F1), we use this model for other experiments throughout the paper. The corresponding scores for SVM, Naive Bayes and Decision Trees classifiers also outperform the baselines TextRank [24], Luhn Summarizer [15], LexRank [10] and SummaRuNNer [27] summarizers. Further, the best performing classifier model also resulted in the best generation of summaries.

9.2 RQ2: How do supervised learning algorithms perform in comparison to the unsupervised ones?

Since we already have highlight sets for many papers, unlike in most of the real world scenarios, we would like to see how this data could help us. We compare the performance of supervised and unsupervised learning algorithms on our data set. From the results in Tables 2,4,5 we see that the supervised techniques outperform the unsupervised ones in terms of the F1 score by a large margin. The difference between the results are statistically significant for both the recall and precision measures (p value = 0.0041 and 0.0314 respectively, student's t-test). The main reason for this is because for the task of extractive summarization, where we have to pick sentences directly from the text, a classifier seems to learn better what features of a sentence contribute towards making a highlight



(EH stands for Ensemble Hard and ES for Ensemble Soft)

Figure 2: Performance of classifiers

Table 2: ROUGE-1 score for different classifiers

Model	Recall	Precision	F1 Score
Lex Rank	0.567	0.162	0.245
Text Rank	0.544	0.128	0.193
SVM	0.632	0.221	0.318
Decision Tree	0.436	0.162	0.223
Naive Bayes	0.623	0.216	0.312
Logistic Regression	0.667	0.244	0.348
Luhn	0.655	0.127	0.208
K-Means	0.599	0.197	0.289
Random Forest	0.589	0.239	0.332
GBDT*	**0.704**	**0.262**	**0.372**
Ensemble-SOFT	0.392	0.132	0.210
Ensemble-HARD	0.635	0.240	0.339
Heuristic Approach	0.134	0.093	0.114
Random Selection	0.312	0.132	0.195
First-Last Selection	0.342	0.122	0.205
SummaRuNNer	0.517	0.212	0.301

* Indicates the best performing algorithm in terms of ROUGE F1 score for classification.

sentence, when we train it with positive and negative examples of highlight sentences than an algorithm where we do not feed in any examples of a highlight sentence.

For K-means clustering algorithm we form clusters of different sizes from the sentences of the article. From each of the generated clusters we then select a few sentences as summary sentences. In our experiments setting the cluster size to the number of sentences in the summary - seven, and then selecting the top sentence of the cluster resulted in the best performance.

9.3 RQ3: How do the ROUGE scores of the generated summaries vary when selecting sentences from certain sections instead of the whole paper?

Different sections within the paper have different roles in the paper itself. It is hypothesized that usually the most important points of the paper are found in either the conclusion, discussion or introduction sections of the article. Figure 3 shows the contribution of the sections when compared with the highlights during training. The introduction, abstract and conclusion sections do have a significant contribution towards producing highlight sentences.

To see what impact the abstract of the paper has on the selection of highlights we perform two experiments. In the first experiment, while training the classifiers, we select sentences as positive examples from throughout the paper. In the second experiment we do not allow the sentences taken from the abstract to be labelled as positive examples. When we allow the selection of sentences from the abstract as positive examples during the training phase, the average ROUGE-L score for the recall and precision both increase by 12% and 7% respectively. This difference in performance of the classifiers is statistically significant (p value = 0.034 and 0.0487 respectively, student's t-test). The main reason for such a large percentage change is because the abstracts are themselves a summary of the entire article. When we choose sentences from the abstract, we take a better sample of sentences for our classifiers to learn from.

While the previous two experiments focused on how the sentences were selected as positive examples during training for our classifiers to learn from, the following experiment compares the performance when we limit the selection of sentences to particular sections instead of the whole article during testing. So, in our experiments we compare the performance of the highlights generated when using our best model from Section 9.1 on the entire contents of the article and only on the contents of the introduction, abstract and conclusion sentences. When we reduce the candidate sentences during testing by only allowing sentences that are part of the introduction, abstract or conclusion as summary sentences we surprisingly found a decrease in the recall scores by about 3.8% (p value = 0.145, student's t-test), and an increase in the precision by 1.3% (p value = 0.121, student's t-test). As the results are not statistically significant we can't conclude that these sections can separately contribute towards generating better highlights. Figure 4 shows the distribution of the sections that the sentences belong to when we select the highlights while testing.

9.4 RQ 4: How can redundancy be reduced among sentences in the generated highlights?

To reduce the redundancy between the sentences in the generated summary we employ a few approaches. In the first approach, we train another classifier that predicts whether a sentence should be added to the summary or not, given the sentences that have already been added to the summary.

The training set for this classifier consists of the positive sentence examples that we used to initially train the GBDT classifier in Section 9.1. The features of the sentences for this classifier are the sentence, the prediction probability of it being a highlight sentence by using the GBDT model from Section 9.1 and the average ROUGE similarity score of the sentence with respect to the sentences that

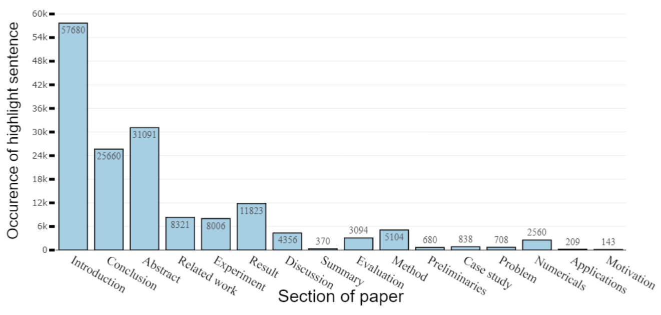


Figure 3: Distribution of sections which best match highlights during training

have already been added to the summary. During training, for each sentence we compute the average ROUGE similarity score of the sentence and the highlight set. For each highlight sentence in the article we label the sentence which maximizes the ROUGE score with that highlight as a positive example and label the rest of the sentences as negative examples. This gives us as many positive examples as there are highlights in the article. During testing, we always add the sentence for which we achieve the highest probability using the GBDT model of Section 9.1 to the summary. For the other sentences, we feed them into our classifier and add it to the summary only if the classifier classifies it as a summary sentence.

The second approach uses K-Means clustering to form clusters equal to the number of sentences in the summary, i.e 7 clusters. From the ranked list of sentences generated by the classifier, we add sentences to the summary if and only if not more than two sentences belong to the same cluster. The last approach uses the concept of Maximal Margin Relevance (MMR) [5]. We compute the cosine similarity between a sentence not yet added to the summary, to the sentences already in the summary. If the cosine similarity is greater than a threshold value of 0.30, we do not add the sentence in consideration to the summary. Although, these methods reduced the average ROUGE recall and ROUGE precision scores of the summary in our experiments, it helped in increasing the coverage of the summary. The recall scores decrease by 8.7% and the precision scores by about 12.3% when we use the K-Mean clustering technique. When using the MMR technique with a fixed cutoff similarity value of 0.3, the recall scores again decrease by 5.8% and the precision scores by about 6.3%. These reductions in scores are mainly because we do not consider a highly ranked sentence as a summary sentence if it either belongs to the same cluster as a sentence already added to the summary, or when it is too similar to the sentences that have already been added to the summary.

Both the recall and precision scores decreased only by about 2.1% and 4.3% respectively when we used a second classifier to select summary sentences, considering the sentences that had already been added to the summary sentences. Such a low difference in the results could mainly be because of an imbalanced dataset during training of the second classifier, where we consider the positive examples equal to the number of sentences in the gold summary, and mark the other sentences as negative examples.

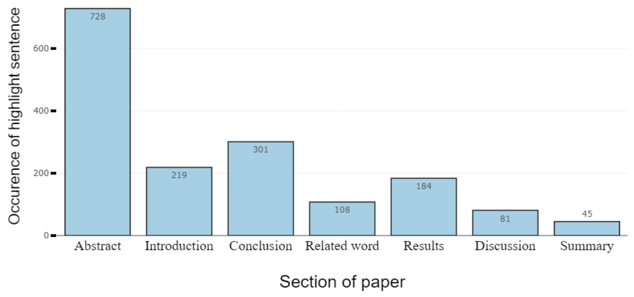


Figure 4: Distribution of sections when choosing highlights during test time

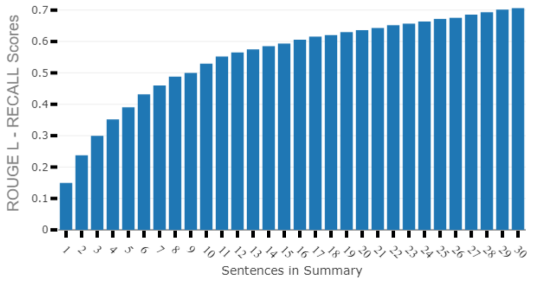


Figure 5: Variation of ROUGE recall score with sentences in summary

9.5 RQ 5: What should be the optimum length of the summary to maximize the ROUGE scores?

We conducted experiments to compare the variation in ROUGE scores when generating summaries of different lengths. The number of sentences is not the same in the different papers. The papers range from having 56 to 945 sentences, with the average number of sentences being 204. [1] suggests that the size of the generated summaries (in general, not domain specific) should be around 5-10% of the original article. We used our best performing algorithm and generated summaries with sentences ranging between 1 to 30. All the average ROUGE recall scores as shown in Figure 5, increase with the increase in the length of the summary. Recall measures the number of words that we are successfully able to match in the gold summary. As we increase the length of sentences in the generated summary, we maximize the chances of extracting all the words from the gold summary. The ROUGE precision scores is highest when the there is only a single sentence in the summary. The score decreases with the increase in number of sentences as shown in Figure 6. This is because,as more sentences are extracted, higher are the chances that some of them may not actually be the highlight sentences resulting in a decreased precision score. Since F1 scores are the harmonic mean of the recall and precision scores, we see they increase up to 7-8 sentences, but then start decreasing as shown in Figure 7. The ROUGE F1 score is maximum for summaries with 6 sentences in them. The ROUGE F1 score for summaries with 7 sentences, is marginally less, but with a higher ROUGE recall score and hence, we generate summaries with 7 sentences through out the thesis.

9.6 RQ 6: Which features of the sentences are most important while selecting a sentence as an highlight?

We use a wide range of features as described in Section 5.5 to train our classifiers. All of the features are not equally important. Some

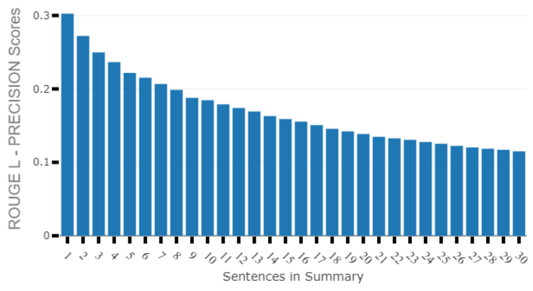


Figure 6: Variation of ROUGE precision score with sentences in summary

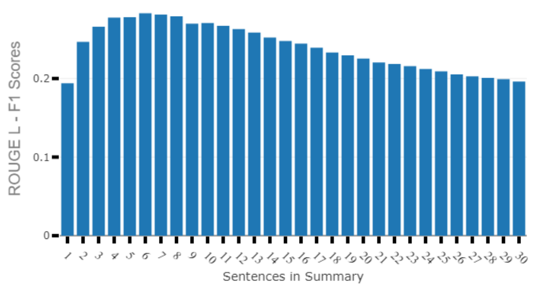


Figure 7: Variation of ROUGE F1 score with sentences in summary

features contribute more when the classifier makes a decision, while some do not contribute in the decision process. We would like to know which are the most discriminative features of a sentence that help in the classification. For our best performing algorithm GDBT, the 'feature importance score' of all the features is as shown in Figure 8.

Figure 8 shows that the most discriminative feature for our classifier is the **Document TF-IDF** score followed by the scores of the sentences when we rank them with the different summarization algorithms. An abstract is usually considered as a summary of an scientific paper, hence a simple hypothesis could be that if a sentence is similar to the abstract of the paper, it is more likely to be a highlight sentence. This hypothesis is proved right from Figure 8, which shows that the *Abstract rouge score* has a very high feature weight associated with it.

The scores from the other summarization algorithms also show that they constitute as important features for the classifier. The TF-IDF of the sentences is also an important feature of the sentences while being selected as a summary sentence. The numeric count feature surprisingly does not have a high feature weight. This could be because there are many mathematical equations within the

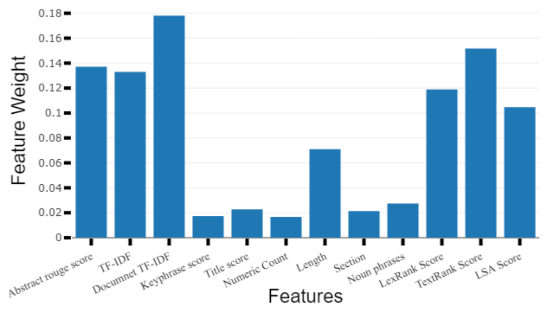


Figure 8: Importance of different features

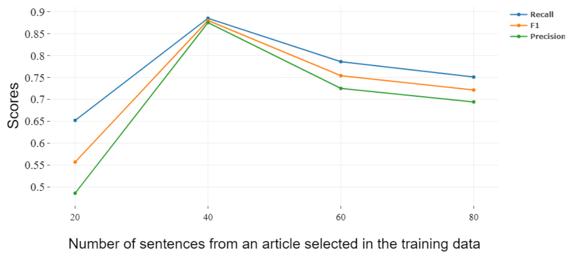


Figure 9: Variation of classification scores with number of examples in training set

papers which have a lot of numerical elements in them, which are not considered as highlights.

9.7 RQ 7: How does the size for the generation of the labelled training dataset effect the classifiers?

We experiment with the selection of different number of sentences as positive and negative examples from the papers. We would like to see how the length of the training data varies the performance of the classifiers. We perform experiments by training the classifiers with 10, 20, 30, and 40 positive and negative examples respectively. The performance of different algorithms with different training length of the training data is shown in Figure 9. The scores are obtained from a 10 fold cross-validation.

We conclude that in our case having 40 sentences (20 as positive and 20 as negative examples) from a single paper, results in the best performance of the classifier. The main reason for this could be that choosing less than 20 sentences underfits, with our classifier not having enough examples to learn from. Contrarily, when more than 20 sentences are chosen, we tend to overfit the classifier.

10 ERROR ANALYSIS

We manually look at some of the articles, their gold highlights and our system generated highlights where the F1 ROUGE-L scores for the generated summary are less than 0.10, to analyze where our model performs poorly. These highlights are shown in Figure 12. From Figure 12, it is clear that our system is not able to model sequential relationships and abbreviations in the article. Using a RNN should help overcome this issue. Moreover, sometimes the gold highlights are not detailed in themselves. While it seems that our generated highlights actually do a good job in covering the major contributions of the paper, we still achieve a very low ROUGE-L F1 score of 0.07 because of very less overlapping words between the gold and generated summary. Thus we cannot fully rely on the highlights provided by the authors and consider them as a gold set. We also see that mathematical symbols, letters of the Greek alphabets used across main scientific domains and chemical symbols like the chemical formula CO_2 for Carbon Dioxide are not parsed properly which possibly effects the performance of our classifiers.

11 FUTURE WORK

- **Co-Reference Resolution:** Extractive summarization can lead to sentences in summaries being out of context. For example a generated highlight sentence could be: 'This approach shows that setting a value of m to 0.34 is the most efficient choice'. This sentence does not give a lot of information to the reader if the definition or reference of m is not found in the other sentences of the summary. Hence using different parts of speech taggers and co reference resolution techniques could be a possible way to eliminate the confusion with respect to co-reference of terms.
- **Citation Context:** When a paper is cited by another paper, the citing paper generally describes the novelty and important features of the cited paper in a few sentences. Analysis of this citation network can help determine important key words and aspects of the paper. This can be used as an additional feature or by appending it to the list of key phrases of a paper, which may result in a better classifier being trained.
- **Using a RNN or a bi-directional LSTM:** Deep neural models like RNN and LSTM seem to have been quite successful for a few NLP tasks like machine translation and speech to text [25]. Encoding the sentences and using a pre trained word embedding to represent sentences and then model and learn important parts of the text can be used to improve the models.
- **Parsing of special symbols:** Scientific papers usually contain letters from the Greek alphabets as special symbols and abbreviations which have a subscript or a postscript attached to them. Parsing these values correctly is a rather difficult task, which we did not do during this thesis. We considered these words as simple words. Taking into account the actual words could lead to a change in the performance of our classifiers.
- **Abstractive Summarization:** Recent state of the art techniques focus more on abstractive summarization. A further extension to the project would be to form abstractive summaries of these scientific articles by using an Encoder-Decoder technique and compare their performance to the extractive techniques developed in this thesis.

12 CONCLUSIONS

Two example articles, with its gold highlight and the generated highlights by our model are shown in Figure 10 and 11. Most of the words in the gold summary are actually retrieved in our generated summary. However, the generated summary still seem to be somewhat long and verbose hence the precision scores of the models are not as high as the recall scores.

In this thesis we have shown that classification algorithms are able to perform very well for the task of highlight extraction for well structured scientific articles. We generated a training candidate dataset of sentences from available highlights that can be used as input to different machine learning algorithms. We analyzed which features of sentence make them suitable to be candidate highlight sentences. We also analyzed how sentences from different sections of the paper contribute towards forming a highlight sentence. We experimented with the size of an 'ideal' system generated summary. We used different ways to reduce the redundancy in the highlight sentences generated. To reduce the redundancy among the highlight sentences we trained a second classifier which classifies sentence from a list of candidate highlight sentences, as a highlight sentence taking into consideration sentences which have already been added to the generated highlights. We also used the scores from previously published algorithms as additional features to train our classifiers. The ROUGE F1 scores of the summaries generated by our model for scientific articles are comparable with the state of the art summarization techniques[23] when generating summaries which are domain independent.

ACKNOWLEDGMENTS

I would like to take the opportunity to thank Dr.Evangelos Kanoulas and Dr.Georgios Tsatsatronis for their constant advice, endless support, motivational words and key insights on how to improve the thesis. I am also grateful to Ms.Tharangni Sivaji for her help and regular suggestions. Lastly, I would like to thank all my colleagues at Elsevier for their help and encouragement. This thesis would not have been possible without you guys !

REFERENCES

[1] Babar, S., Tech-Cse, M., and , R. (2013). Text summarization:an overview.

[2] Barrios, F., López, F., Argerich, L., and Wachenchauzer, R. (2016). Variations of the similarity function of textrank for automated summarization. *CoRR*, abs/1602.03606.

[3] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

[4] Brin S, P. L. (1998). The pagerank citation ranking: Bringing order to the web. In *In proceedings of the 7th international conference on world wide web*, pages 161–172.

[5] Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 335–336, New York, NY, USA. ACM.

[6] Conroy, J. and O'leary, D. (2001). Text summarization via hidden markov models. pages 406–407.

[7] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20(3):273–297.

[8] David M. Zajic, Bonnie J. Dorr, J. L. (2008). Single-document and multi-document summarization techniques for email threads using sentence compression. *Information Processing and Management*, 44(4):1600–1610.

[9] Elena Baralis, LucaCagliero, N. A. (2013). Graphsum: Discovering correlations among multiple terms for graph-based summarization. *Information Sciences*, 249(1):96–109.

[10] Erkan, G. and Rade, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

[11] Fattah, M. A. (2014). A hybrid machine learning model for multi-document summarization. *Applied Intelligence*, 40(4):592–600.

[12] Fiori, A. (2014). *Innovative Document Summarization Techniques: Revolutionizing Knowledge Understanding*. IRCC.

[13] Ganesan, K. (2015). Rouge 2.0: Updated and improved measures for evaluation of summarization tasks.

[14] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., and Wang, G. (2015). Recent advances in convolutional neural networks. *CoRR*, abs/1512.07108.

[15] H, L. (1958). The automatic creation of literature abstracts. *IBM J Res Dev 2*, page 159âĂŞ165.

[16] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C., Silverman, R., and Wu, A. Y. (2000). An efficient k-means clustering algorithm: Analysis and implementation.

[17] Li, P., Lam, W., Bing, L., and Wang, Z. (2017). Deep recurrent generative decoder for abstractive text summarization. *CoRR*, abs/1708.00625.

[18] Libin Yang, Xiaoyan Cai, Y. Z. P. (2014). Enhancing sentence-level clustering with ranking-based clustering framework for theme-based summarization. *Information Sciences*, 260:37–50.

[19] Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries.

[20] Lin, C.-Y. and Hovy, E. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. pages 71–78.

[21] Loper, E. and Bird, S. (2002). Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

[22] Ma, S. and Sun, X. (2017). A semantic relevance based neural network for text summarization and text simplification. *CoRR*, abs/1710.02318.

[23] Mahak Gambhir, V. G. (2017). Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66.

[24] Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into text. In *EMNLP*.

[25] Minar, M. R. and Naher, J. (2018). Recent advances in deep learning: An overview.

[26] Mohamed Abdel Fattah, F. (2009). Ga, mr, ffnn, pnn and gmm based models for automatic text summarization. *Computer Speech & Language*, 23(1):126–144.

[27] Nallapati, R., Zhai, F., and Zhou, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*.

[28] Papadimitriou, C. H., Tamaki, H., Raghavan, P., and Vempala, S. (1998). Latent semantic indexing: A probabilistic analysis. In *Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '98, pages 159–168, New York, NY, USA. ACM.

[29] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[30] Porter, M. F. (1997). Readings in information retrieval. chapter An Algorithm for Suffix Stripping, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[31] Quinlan, J. R. (1986). Induction of decision trees. *Mach. Learn.*, 1(1):81–106.

[32] Ramos, J. (2003). Using tf-idf to determine word relevance in document queries.

[33] Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. *CoRR*, abs/1509.00685.

[34] Schultz, D. M. (2009). *The Structure of a Scientific Paper*, pages 29–47. American Meteorological Society, Boston, MA.

[35] Sparck Jones, K., E.-N. B. (1995a). Automatic summarizing. *Information Processing & Management*, 31(5):625âĂŞ630.

[36] Sparck Jones, K. (1995b). What might be in a summary ? *Information Retrieval.*

[37] Woodsend, K. and Lapata, M. (2010). Automatic generation of story highlights. In *In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 565–574.

A APPENDIX

Table 3: JSON file format

Attributes	Description
FileName	Filename of the Article
Gold Summary	Gold Summary of the article
Title	Title of the article
Abstract Vectors	Averaged non stop word vectors of the abstract
Sentences	Shuffled list of sentences from the article
Sentence Features	List of sentence features and associated labels for the sentences

Table 4: ROUGE-2 scores for different classifiers

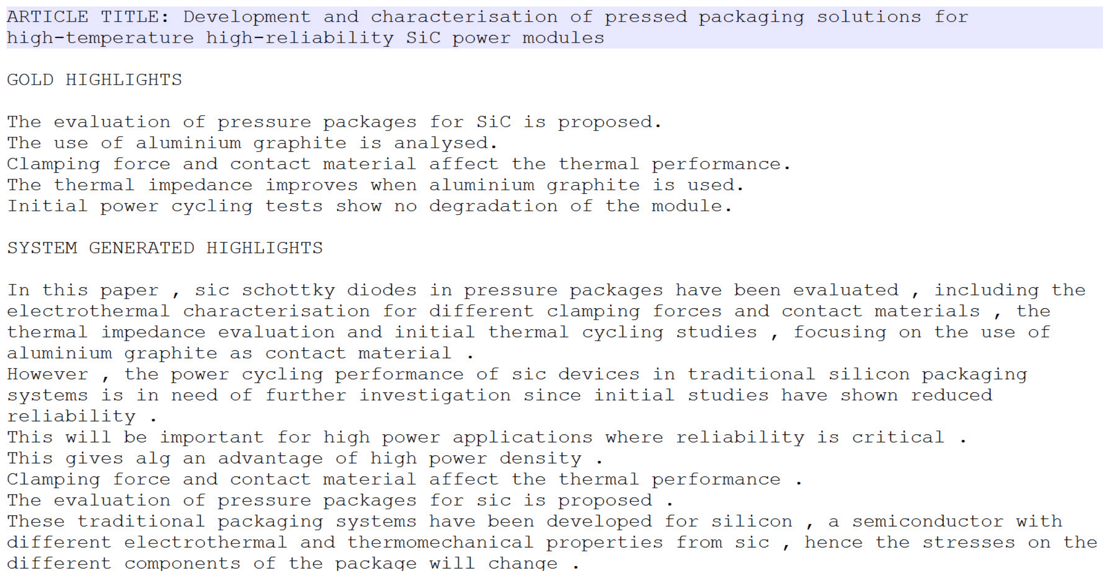
Model	Recall	Precision	F1 Score
Lex Rank	0.314	0.036	0.058
Text Rank	0.323	0.016	0.029
SVM	0.370	0.080	0.127
Decision Tree	0.185	0.027	0.048
Naive Bayes	0.358	0.073	0.118
Logistic Regression	0.546	0.058	0.089
Luhn	0.401	0.015	0.027
K-Means	0.356	0.098	0.105
Random Forest	0.389	0.064	0.093
GBDT[*]	**0.555**	**0.202**	**0.301**
Ensemble-SOFT	0.159	0.054	0.078
Ensemble-HARD	0.466	0.176	0.245
Heuristic Approach	0.104	0.076	0.098
Random Selection	0.212	0.124	0.156
First-Last Selection	0.204	0.104	0.138
SummaRuNNer	0.478	0.193	0.278

[*] Indicates the best performing algorithm in terms of ROUGE-2 F1 score for classification.

Table 5: ROUGE-SU4 scores for different classifiers

Model	Recall	Precision	F1 Score
Lex Rank	0.240	0.044	0.074
Text Rank	0.208	0.020	0.037
SVM	0.149	0.075	0.100
Decision Tree	0.136	0.041	0.061
Naive Bayes	0.290	0.034	0.065
Logistic Regression	0.253	0.066	0.104
Luhn	0.229	0.019	0.035
K-Means	0.235	0.074	0.098
Random Forest	0.210	0.051	0.083
GBDT[*]	**0.219**	**0.070**	**0.106**
Ensemble-SOFT	0.113	0.048	0.083
Ensemble-HARD	0.164	0.071	0.091
Heuristic Approach	0.134	0.093	0.094
Random Selection	0.083	0.022	0.035
First-Last Selection	0.142	0.045	0.069
SummaRuNNer	0.111	0.080	0.924

[*] Indicates the best performing algorithm in terms of ROUGE-SU4 F1 score for classification.



ARTICLE TITLE: Development and characterisation of pressed packaging solutions for high-temperature high-reliability SiC power modules

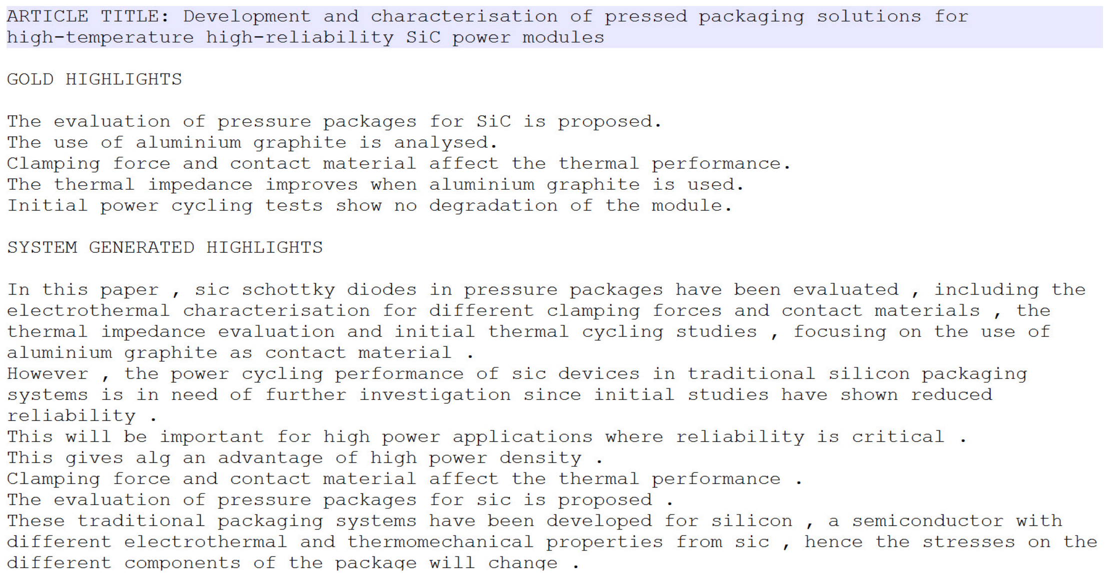
GOLD HIGHLIGHTS

The evaluation of pressure packages for SiC is proposed.
The use of aluminium graphite is analysed.
Clamping force and contact material affect the thermal performance.
The thermal impedance improves when aluminium graphite is used.
Initial power cycling tests show no degradation of the module.

SYSTEM GENERATED HIGHLIGHTS

In this paper , sic schottky diodes in pressure packages have been evaluated , including the electrothermal characterisation for different clamping forces and contact materials , the thermal impedance evaluation and initial thermal cycling studies , focusing on the use of aluminium graphite as contact material .
However , the power cycling performance of sic devices in traditional silicon packaging systems is in need of further investigation since initial studies have shown reduced reliability .
This will be important for high power applications where reliability is critical .
This gives alg an advantage of high power density .
Clamping force and contact material affect the thermal performance .
The evaluation of pressure packages for sic is proposed .
These traditional packaging systems have been developed for silicon , a semiconductor with different electrothermal and thermomechanical properties from sic , hence the stresses on the different components of the package will change .

Figure 10: System and Gold Highlights -1



ARTICLE TITLE: Development and characterisation of pressed packaging solutions for high-temperature high-reliability SiC power modules

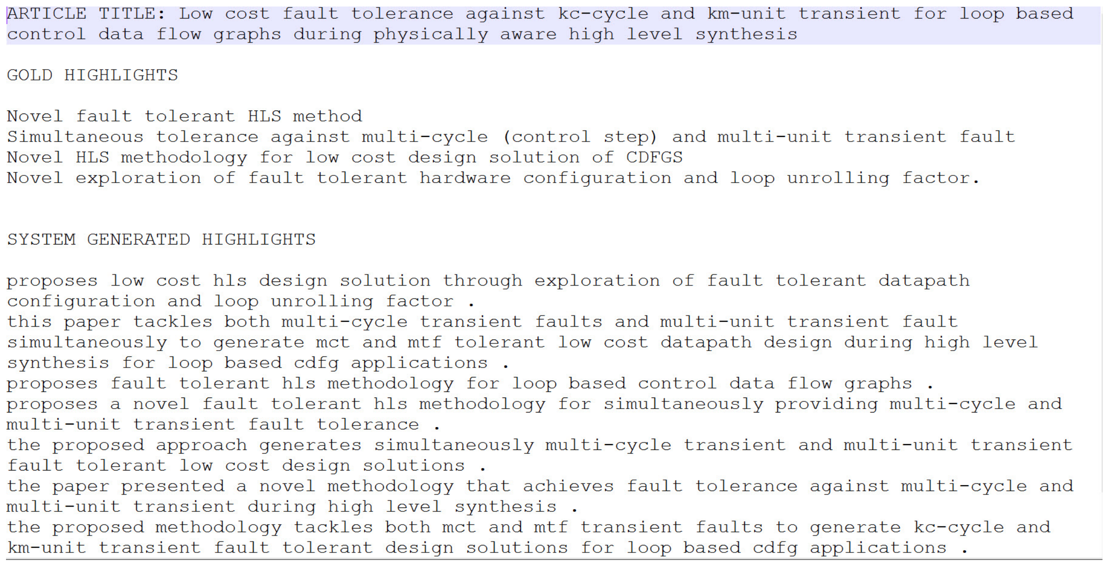
GOLD HIGHLIGHTS

The evaluation of pressure packages for SiC is proposed.
The use of aluminium graphite is analysed.
Clamping force and contact material affect the thermal performance.
The thermal impedance improves when aluminium graphite is used.
Initial power cycling tests show no degradation of the module.

SYSTEM GENERATED HIGHLIGHTS

In this paper , sic schottky diodes in pressure packages have been evaluated , including the electrothermal characterisation for different clamping forces and contact materials , the thermal impedance evaluation and initial thermal cycling studies , focusing on the use of aluminium graphite as contact material .
However , the power cycling performance of sic devices in traditional silicon packaging systems is in need of further investigation since initial studies have shown reduced reliability .
This will be important for high power applications where reliability is critical .
This gives alg an advantage of high power density .
Clamping force and contact material affect the thermal performance .
The evaluation of pressure packages for sic is proposed .
These traditional packaging systems have been developed for silicon , a semiconductor with different electrothermal and thermomechanical properties from sic , hence the stresses on the different components of the package will change .

Figure 11: System and Gold Highlights -2



```
ARTICLE TITLE: Low cost fault tolerance against kc-cycle and km-unit transient for loop based
control data flow graphs during physically aware high level synthesis

GOLD HIGHLIGHTS

Novel fault tolerant HLS method
Simultaneous tolerance against multi-cycle (control step) and multi-unit transient fault
Novel HLS methodology for low cost design solution of CDFGS
Novel exploration of fault tolerant hardware configuration and loop unrolling factor.


SYSTEM GENERATED HIGHLIGHTS

proposes low cost hls design solution through exploration of fault tolerant datapath
configuration and loop unrolling factor .
this paper tackles both multi-cycle transient faults and multi-unit transient fault
simultaneously to generate mct and mtf tolerant low cost datapath design during high level
synthesis for loop based cdfg applications .
proposes fault tolerant hls methodology for loop based control data flow graphs .
proposes a novel fault tolerant hls methodology for simultaneously providing multi-cycle and
multi-unit transient fault tolerance .
the proposed approach generates simultaneously multi-cycle transient and multi-unit transient
fault tolerant low cost design solutions .
the paper presented a novel methodology that achieves fault tolerance against multi-cycle and
multi-unit transient during high level synthesis .
the proposed methodology tackles both mct and mtf transient faults to generate kc-cycle and
km-unit transient fault tolerant design solutions for loop based cdfg applications .
```

Figure 12: System and Gold Highlights when the F1 ROUGE-L score is less than 0.1