Automatic Summarization of Technical Documents in the Oil and Gas Industry

João Marcos Correia Marques
Escola Politécnica
Universidade de São Paulo
São Paulo, Brazil
joao.marques@usp.br

Fabio Gagliardi Cozman
Escola Politécnica
Universidade de São Paulo
São Paulo, Brazil
fgcozman@usp.br

Ismael Humberto Ferreira dos Santos
CENPES - Petrobras Research Center
Petrobras
Rio De Janeiro, Brazil
ismaelh@petrobras.com.br

***Abstract*—We address extractive summarization of technical documents in the oil and gas industry, a major and urgent task due to the large volume of critical reports in that industry. We examine five distinct state-of-the-art extractive algorithms; to assess performance, a new open dataset was created using the open access Journal of Petroleum Exploration and Production Technology (JPEPT). Abstracts for papers in this journal were used as ground truths for summarization. Algorithms were refined to work with these documents in the best possible way. Our most effective algorithm achieved a state-of-the-art ROUGE-2 score of 0.123, taking 83 minutes to summarize the entire JPEPT dataset.**

***Index Terms*—Automatic Text Summarization, Extractive Summarization, Natural Language Processing**

I. Introduction

Hundreds of pages of reports are generated on a daily basis within large organizations; those reports contain precious data that could lead to actionable insights for companies [1]. This is particularly true in the oil and gas industry, where reports are often safety-critical and failure to act in due time on these reports may cause catastrophic ecological and social damage. Most of the information, however, is stored in a human-friendly but machine-adverse format. Alas, manual extraction of information does not scale well and automatic extraction of structured information from unstructured text remains an active area of research and debate within the machine learning community [2].

One option is to keep the human in the loop, by producing summaries of documents, and letting the end user analyze those summaries. This is the path we have followed in our effort to enhance information processing within a large oil company. In this paper we describe automatic summarization techniques aimed at the oil and gas industry.

Automatic text summarization is divided into three approaches: Abstractive, Extractive and Mixed summarization [3]. In Extractive summarization, the goal is to devise algorithms that select the most relevant sentences from a given source and to present them as-is to the reader. In Mixed summarization, the most relevant sentences are selected from the text, but undergo post-processing prior to presentation to ensure cohesion between sentences. Finally, Abstractive summarization creates completely new sentences to summarize the original text based on complex language models [3].

The construction of complex language models for adequate post-processing or fully abstractive summarization, however, is an open problem, with most solutions requiring large labelled datasets, being applied to overly simplified problems and presenting serious cohesion and legibility issues [4]. Indeed, most applicable solutions available today are Extractive. This work aims at providing a benchmark for Extractive techniques of Automatic Text Summarization applied to technical literature in the oil and gas industry, considering not only their textual performance, but also their execution times.

This paper is organized as follows. Section II describes metrics that are used to evaluate summarization techniques. We describe our literature review and selection of techniques in Section III. Section IV presents the dataset we have created, a contribution that will be valuable to other researchers interested in summarization of technical reports. Experimental results are described and discussed in Section V.

II. Automatic Summary Evaluation Metrics

Let us first define the evaluation metric that is used in ranking automatically generated abstracts. While there exist many human-based strategies for assessing the quality of an automatically generated abstract, such as the ones used by Parveen and Stube [5], the most used metric in the field is the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [3], [6]. ROUGE is, in fact, a set of metrics. Relevant to this work, however, are the ROUGE-N and ROUGE-L metrics.

ROUGE-N is defined as the n-gram recall between a candidate summary and its set of references [6]:

$$\text{ROUGE-N} = \left(\frac{\Sigma_{S \in RefSums} \Sigma_{gram_n \in S} Count_{match}(gram_n)}{\Sigma_{S \in RefSums} \Sigma_{gram_n \in S} Count(gram_n)} \right), \quad (1)$$

where S is the set of sentences in the reference summaries, $gram_n$ is an n-gram contained within this set of sentences and $Count_{match}$ is the number of matching n-grams between the automated summary and its references.

ROUGE-L, on the other hand, is based on the Longest Common Sub-sequence between the abstracts, given that the longer the sub-sequence of words that can be found between

2643-6264/19/$31.00 ©2019 IEEE
DOI 10.1109/BRACIS.2019.00082

two different texts, the more similar they should be [6]. It is defined in (2),

$$R_{lcs} = \frac{\Sigma_{i=1}^{u} LCS_U(r_i, C)}{m}, \quad (2)$$

where r_i is a sentence from the reference summaries, c_i is a sentence from the automated summary, C is the set of all sentences in the automated summary, containing n words in total, and R is the set of all sentences in the reference summaries, containing m words in total. Similarly, it is defined in (3),

$$P_{lcs} = \frac{\Sigma_{i=1}^{v} LCS_U(c_i, R)}{n}. \quad (3)$$

We may then define the ROUGE-L metric in (4):

$$ROUGE - L = F_{lcs} = \frac{(1+\beta^2) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}, \quad (4)$$

where β is the F-beta weighting factor, usually set to 1.

These metrics correlate well with human judgment of abstracts, in particular the ROUGE-2 variant of ROUGE-N and ROUGE-L [6]. These metrics, were, therefore, used to determine which algorithms were worth exploring for our task.

III. Extractive Automatic Text Summarization Algorithms

There are many summarization algorithms in the literature, so the first challenge in this work was to determine which algorithms were appropriate to the realities of the oil and gas industry. When doing so, we considered the constraints of working within the context of a large oil company with a substantial number of stored reports and an urgent need to summarize them, so that this work could be applicable to the real world. This helped us set the limits on execution times and computational complexity of the algorithms to be tested.

Algorithms in the literature were ranked according to their performance on the Document Understanding 2002 (DUC2002) dataset, whose main focus was on developing generic summarization algorithms for summarizing news stories. This dataset is comprised of several news articles spanning multiple topics and, for each topic, has at least 3 handcrafted abstractive and extractive summaries of varying lengths made by different people. In their Survey on Automatic Text Summarization, Gambhir and Gupta [3] aggregated the performances of several state-of-the-art algorithms on that dataset, as can be seen in Figure 1. One can clearly see that the algorithms OCDSum-SaDE [7], BSTM [8], FGB [9] and Sum_Coh [5] were the best performing algorithms according to the ROUGE-2 metric. In addition, when examining the performances reported by Gambhir and Gupta for the similar summarization dataset, DUC2004, which can be seen in Figure 2, we note that the Progressive algorithm [10] clearly dominates its competitors and is, therefore, also worthy of testing. These five algorithms were thus selected to be tested in this work and are detailed in the following subsections. Their selection is interesting as they also represent three different approaches to automatic text summarization: Pure Optimization-based (Sum_Coh and OCDSumSaDE), Maximum Likelihood Estimation on Bayesian unigram language models (BSTM, FGB) and Graph-based (Progressive) algorithms - and their evaluation might provide valuable insight on the suitability of each technique to the summarization of technical literature.

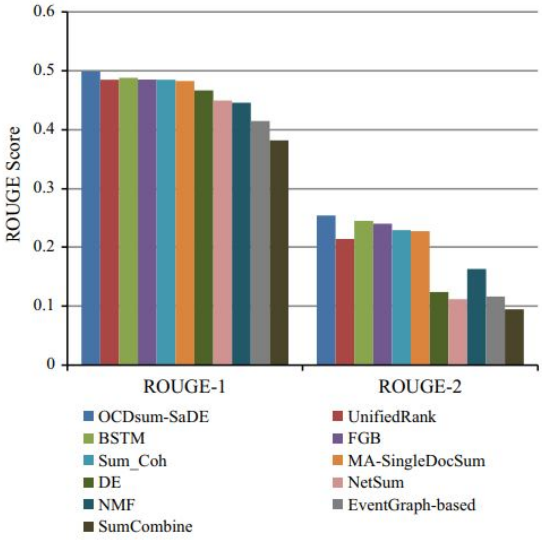

Fig. 1. State-of-the-art ROUGE-1 and ROUGE-2 performances on the DUC2002 dataset [3]

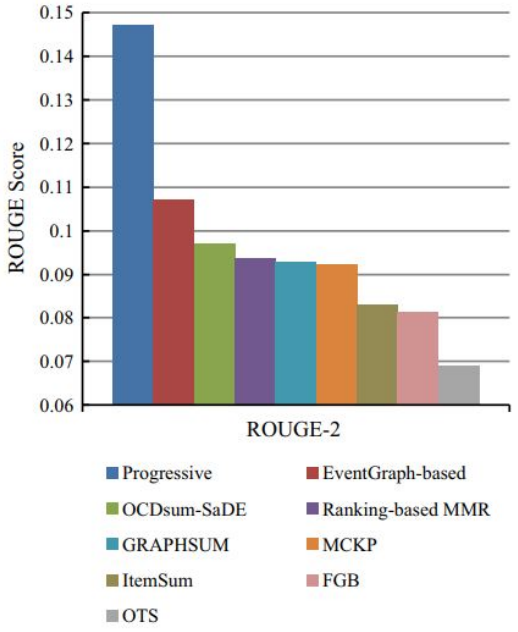

Fig. 2. State-of-the-art ROUGE-2 performances on the DUC2004 dataset [3]

A. OCDSum-SaDE

Alguliev et al. [7] formulate the summarization problem as a 3-fold optimization task, aiming to optimize **content coverage**,

i.e., the amount of new information in the summary, **diversity**, understood here as non redundancy between selected sentences and length. In order to do so, the authors define sentences as being represented by their Term Frequency-Inverse Sentence Frequency (TF-ISF) matrices.

Letting the similarity between two sentences s_i and s_j be defined as the cosine similarity between their rows in the TF-ISF matrix, as given by Expression (5), the authors define coverage and diversity according to Expressions (6) and (7), respectively,

$$sim(s_i, s_j) = \frac{s_i \cdot s_j}{\|s_i\| \cdot \|s_j\|}, \tag{5}$$

$$f_{cover}(X) = sim(O^D, O^S) \cdot \sum_{i=1}^{n} sim(O^D, s_i) x_i, \tag{6}$$

$$f_{diver}(X) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} sim(s_i, s_j) \cdot x_i x_j, \tag{7}$$

where O^D stands for the average vector of the summary sentences, O^S stands for the average vector for all the sentences within the document, *n* is the total number of sentences in the document, and *X* is a binary vector of size *n*, in which x_i is one if, and only if, sentence x_i has been selected as part of the summary.

Having defined those metrics, they formulate summary creation as the optimization of

$$X_{opt} = \arg\max_{X} \left(\frac{f_{cover}(X)}{f_{diver}(X)} \right) \tag{8}$$

subject to

$$\sum_{i=1}^{n} l_i x_i \leq L, x_i \in 0, 1, \tag{9}$$

where *L* is the maximum summary length. The authors then solve this optimization problem using a variant of the differential algorithm described by Storn and Price [11], selecting all sentences for which x_i is 1 to compose the final summary.

B. BSTM

Proposed by Wang et al. [8], the Bayesian Sentence-based Topic Models is a converging iterative algorithm that generates a unigram model-based probability distribution for each sentence within each of the topics discussed in the main texts by estimating the convergence of its variational bound via Dirichlet Adjustment.

They first define the Dirichlet adjustment y of size K of a vector x of size K with respect to the Dirichlet distribution $D_K(\alpha)$ as being described by

$$y_k = \exp\left(\Psi(\alpha_k + x_k) - \Psi\left(\sum_i (\alpha_i + x_i) \right) \right), \tag{10}$$

denoting it as $y = P_D(x; \alpha)$. They then define the operation $Y = \xleftarrow{P_D(\cdot,\alpha),2} X$ as the following sequence of operations: $z = P_D((X_d, .)^T; \alpha)$ followed by $y_{d,k} = z_k / \sum_k z_k$, operation $Y \xleftarrow{1} X$ as obtaining Y via normalizing each column of X and finally, X as the element-wise multiplication between matrices X and Y.

They then define iterative equations:

$$U \xleftarrow{1} B^T \left[\frac{Y}{B\tilde{U}\tilde{V}^T} \right] \tilde{V} \circ \tilde{U} \tag{11}$$

and

$$V \xleftarrow{P_D(\cdot,\alpha),2} \left[\frac{Y}{BU\tilde{V}^T} \right]^T (BU) \circ \tilde{V}, \tag{12}$$

where Y is the term-document matrix, B is the term-sentence matrix, U is the sentence-topic matrix (the probability distribution of each sentence over each topic), V is an auxiliary document-topic matrix, K is the total number of latent topics and $\tilde{V}$ and $\tilde{U}$ are their estimates at the current step. They also define the convergence metric - the variational bound - according to

$$\tilde{f}(U, V; Y) = \prod d \frac{B(\alpha + \gamma_{d,.})}{B(\alpha)} \prod_{vkwd} \left(\frac{B_{wv} U_{vk}}{\phi_{vk;wd}} \right)^{(Y_{wd} \phi_{wk;wd})}, \tag{13}$$

where $\phi_{vk;wd} = B_{wv} U_{vk} V_{dk} / \left[BUV^T \right]_{wd}$ and $\gamma_{dk} = \sum_{wv} Y_{wd} \phi_{vk;wd}$.

The authors then define Algorithm 1 to obtain the final sentence-topic matrix U_{final}. The most probable sentences for each topic from matrix U are selected to form the final summary.

Algorithm 1 BSTM iterative algorithm

```
0: Randomly initialize U and V and normalize them
1: repeat
1:     Update U using (11)
1:     Update V using (12)
1:     Compute f̃ using (13)
2: until f̃ converges.
3: return U
4: end =0
```

C. FGB

Wang et al. [9] have proposed the Factorization with Given Bases (FGB) algorithm, which uses models that are similar to BSTM, but that speeds up calculations by applying a slightly different convergence criterion as described in Algorithm 2. The most likely sentences for each topic in matrix U are selected to compose the summary.

D. Sum_Coh

This method, introduced by Parveen and Strube [5], is an example of graph-based summarization, which takes into consideration sentence importance, non-redundancy and local coherence when creating the final summary. It starts by representing the whole text as a bipartite graph $G = (V_s, V_e, E_{e,s}$ with two sets of nodes: sentences and entities. Such graph, called entity graph, was introduced by Guinaudeau and Strube [12] to evaluate local coherence of documents. They provide initial ranks for each entity, taking into consideration its

Algorithm 2 FGB iterative algorithm

0: Randomly initialize U and V to follow Dirichlet distribution with hyperparameters α_U and α_V, respectively.
1: **repeat**
1: Compute $C_{ij} = Y_{ij} / \left[BUV^T\right]_{ij}$;
1: $U \xleftarrow{U} \left[B^T CV\right] + \alpha_U$ and normalize columns to 1;
1: Compute Compute $C_{ij} = Y_{ij} / \left[BUV^T\right]_{ij}$
1: $V \xleftarrow{V} \left[C^T BU\right] + \alpha_V$ and normalize rows to 1;
2: **until** U converges
3: **return** U,V
4: end =0

frequency in the text and its presence in the title and for each sentence, and its similarity to the document's title by using Expression (5). They then apply the HITS algorithm [13] to determine final ranks for each sentence. They finally create a one-mode projection of this graph on the sentence nodes (P) and use their outdegrees in this projection as a proxy for the position penalized coherence between sentences:

$$f(s_i) = \frac{coherence(s_i, P)}{position(s_i)} = \frac{Outdegree(s_i, P)}{position(s_i)}. \quad (14)$$

Having defined both sentence ranks and coherence, the authors then treat the summarization as the optimization problem

$$X, Y = \arg\max_{X,Y} \left(\sum_{i=1}^{n} Rank(s_i)_i + \sum_{i=1}^{n} f(s_1)_i + \sum_{j=1}^{m} y_j \right), \quad (15)$$

subject to the constraints

$$\begin{cases} \sum_{i=1}^{n} x_i \leq Len(summary); \\ \sum_{j_i} y_j \geq Entities_{x_i i}, for i = 1, ..., n; \\ \sum_{i_j} x_i \geq y_j, for j = 1, ..., m, \end{cases} \quad (16)$$

where x_i is a Boolean indicating whether a sentence has been selected, while y_i is a Boolean indicating whether an entity has been selected, with the final summary being constructed from the sentences indicated by x_i.

E. Progressive

Ouyang et al [10] proposed a greedy selection method based on the concept of conditional saliency. In order to define this concept, they first establish the notion of subsuming relationships between words. They start by defining the coverage of a word w over a set of words W ($COV(w|W)$) according to (17), in which SPAN(x) is the set of sentences in which the set of words x is observed in the original text.

$$COV(w|W) = \frac{|SPAN(w) \cap_i SPAN(W)|}{|SPAN(w)|}, \quad (17)$$

They then define the concept of subsuming words. For a word w_0 to subsume word w (denoted here as $w_0 \supset w$) condition (18) must be satisfied, in which $\lambda_1, \lambda_2 \in [0, 1]$ are hyperparameters in this model. An acyclic directed graph G is then constructed having the words as nodes and the subsuming relationships as directed edges.

$$w_0 \supset w \text{ iff } \begin{cases} COV(w|w_0) \geq \lambda_1 \text{ and} \\ COV(w|S) < \lambda_2, \text{ where } S = w_i \in W | w_0 \supset w_i. \end{cases} \quad (18)$$

The authors then finally define the concept of conditional saliency in (19), where s is a candidate sentence to be added, s' is the set of current summary sentences and $CON(w_i, w'_l)$ is a Boolean variable indicating whether there exists a path between word w_i and word w'_j in a subgraph of G containing only the nodes and edges relating to the words in $s \cup s'$.

$$CS(s|s') = \sum_{w_i \in s} \log \left(\max_{w'_j \in s'} CON(w_i|w'_j) \cdot |SPAN(w_i)| \right), \quad (19)$$

Once the initial conditional saliencies for each sentence are calculated, the algorithm adds the most salient sentence to the summary and recalculates the conditional saliencies of the remaining sentences, iteratively adding the most salient sentences at each step (given the current summary) to the summary until maximum summary length has been achieved.

IV. Dataset Collection

To properly evaluate the performance of the previous five algorithms in the task of summarizing technical literature about oil and gas, a proper dataset was needed. To the best of our knowledge, no open dataset of this kind is now available. We, therefore, set forth to create it, following the methodology described in Ref. [5], where a similar dataset on the field of medicine was created using the PLOS ONE open set of papers. We took as a source of documents the open access *Journal of Petroleum Exploration and Production Technology* (JPEPT). We manually collected all the papers of the first three volumes of the journal (proper citations omitted here due to length restrictions, but found at *https://github.com/joaomcm/Automatic-Summarization-of-Technical-Literature-on-Oil-and-Gas*, this project's Github repository). These papers were then processed as follows:

- The abstracts were removed from the papers' body and stored in a separate file.
- The remaining body of text was purged of tables, figures and equations, while also removing sections of the papers that merely described deductions and images.
- This clean version of the body was stored on another file.

This procedure yielded the JPEPT technical summary database, described together with the DUC2002 dataset in Table I. One can clearly see that the documents are, on average much larger than those of DUC2002, both in the number of words and sentences, which suggests that performance of algorithms evaluated on the latter might be different from those observed in the former.

TABLE I
DUC2002 AND JPEPT DATASET CHARACTERISTICS

Dataset	# of Topics	Avg. Docs/Topic	Avg. Words/Doc	Avg. sentences/Doc
DUC2002	59	6	613.2	30.8
JPEPT	69	1	2475.1	109.2

V. RESULTS

We produced a Python implementation of each algorithm and tested it in the DUC2002 dataset to verify whether their implementation's performance was consistent with the performance reported in Ref. [3]. The algorithms were then tested on the newly created JPEPT dataset and the automatically generated abstracts were compared with the actual abstract of each paper. In addition, following Ref [5], two alternative algorithms were proposed as benchmarks: Random and First. The Random algorithm randomly picks sentences from the body of the file until the maximum abstract length has been reached, whereas the First algorithm picks the first k sentences of the document until maximum abstract length was reached. The desired abstract length was set to IEEE's 250 word limit for all algorithms running on the JPEPT dataset. Both algorithms serve as trivial and random benchmarks, respectively.

All these experiments were run on an ASUS ROG G75VW notebook, equipped with an Intel Core-i7 3630QM 2.4 GHz processor with 16Gb of RAM and a GTX 670MX graphics card running Ubuntu 16.04. As can be seen in Figure 3, execution times varied widely, but were all well bellow the 8-hours threshold set for a system that should run overnight.

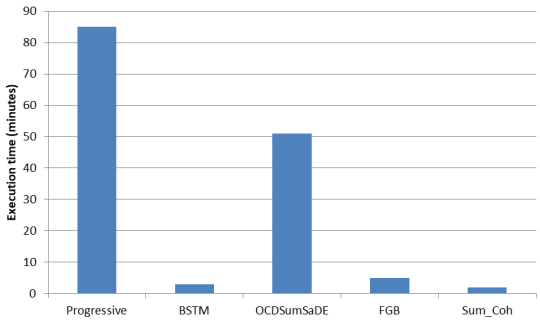

Fig. 3. Algorithm execution times on entire JPEPT dataset

With respect to performance, there was a technical tie between Sum_Coh and the Progressive algorithms. It can be seen in Figure 4 that both algorithms clearly outperform their competitors (including Random and First), but tie in ROUGE-L performance by a small margin. This tendency is confirmed by Table II, where we can see that both of them significantly differ from competing algorithms (with at least 10^{-2} significance), whereas no statistically significant difference may be traced between their performances when compared to one another.

A similar tendency is observed when analyzing ROUGE-2 performance, seen in Figure 5, with both Sum_Coh and the Progressive algorithm clearly outperforming their competition,

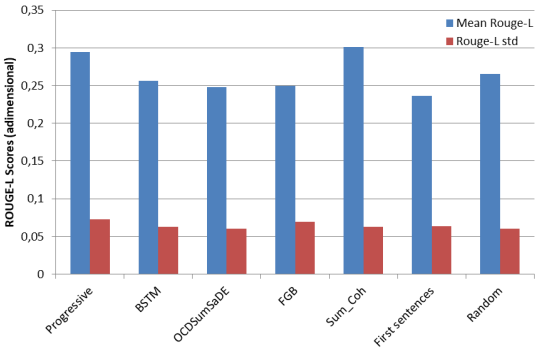

Fig. 4. Mean ROUGE-L scores on the JPEPT dataset

TABLE II
P-VALUE FOR THE SINGLE SIDED T-STUDENT TEST ON ROUGE-L SCORES

Algorithm	Progressive	BSTM	OCDSumSaDE	FGB	Sum_Coh	First	Random
Progressive	-	$>10^{-3}$	$>10^{-5}$	$>10^{-3}$	-	$>10^{-6}$	$>8.10^{-3}$
Sum_Coh	0.27	$>10^{-3}$	$>10^{-5}$	$>10^{-3}$	-	$>10^{-6}$	$>10^{-3}$

but presenting little difference between them. This tendency, however, is shifted when we analyze the statistical significance of the results, shown in Table III, where it can be seen that Progressive significantly outperforming the other algorithms (with at least 10^{-2} significance) and outperforming Sum_Coh at the 0.1 significance level, whereas Sum_Coh fails to have the same level of significance when compared to the BSTM and Random algorithms.

Shockingly, the leading algorithm in the DUC2002 dataset in Figure 1 performed the worst in all metrics when applied to the JPEPT dataset. We conjecture that optimal parameters reported in Ref. [7] for the differential evolution-base genetic algorithm are inappropriate for the present task, as the datasets — and hence, the search spaces — were considerably different, with DUC2002 having much redundant information between files.

It is noteworthy, as well, that the standard deviation of ROUGE-2 scores seems to be of the same order of magnitude as the mean scores. To investigate the cause of such a high ratio between standard deviation and mean ROUGE-2 scores, a qualitative analysis of the results was performed. When contrasting the top performing with the worst performing abstracts created by the Progressive algorithm (which incidentally also had the biggest standard deviation of scores), a clear distinction could be traced. Indeed, in the worst performing summaries, there was a clear tone mismatch between the papers and their abstracts, which typically happened in papers describing new modelling equations, rather than experiments. On the other hand, the best performing papers tended to have a similar discourse style, as well as longer introduction and conclusion sections, which mostly served as the source of the selected sentences. This suggests that more descriptive technical reports would be better summarized by this method, which is appropriate when considering its future applications.

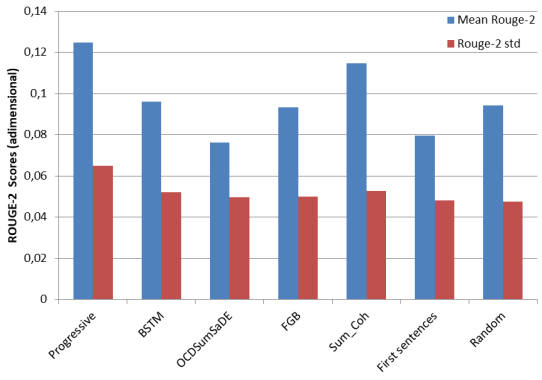

Fig. 5. Mean ROUGE-2 scores on the JPEPT dataset

TABLE III
P-VALUE FOR THE SINGLE SIDED T-STUDENT TEST ON ROUGE-2 SCORES

Algorithm	Progressive	BSTM	OCDSumSaDE	FGB	Sum_Coh	First	Random
Progressive	-	$>2.5.10^{-3}$	$>10^{-5}$	$>10^{-3}$	0.15	$>10^{-6}$	$>10^{-3}$
Sum_Coh	-	$>2.10^{-2}$	$>10^{-5}$	$>10^{-3}$	-	$>10^{-5}$	$>2.10^{-2}$

VI. CONCLUSIONS

This work determined that, in terms of ROUGE performance on the summarization of technical literature on oil and gas, Sum_Coh [5] and Progressive [10] are indistinguishable from one another with any statistical significance, while significantly outperforming competing algorithms and keeping their execution times well bellow the 8-hour overnight threshold. We note, however, that before considering those algorithms for commercial use, other aspects must be considered, such as adaptability, reproducibility and interpretability. When taking these factors into consideration, Progressive, being a deterministic white-box model, allows for greater interpretability and is more easily configured to suit one's needs, while always yielding the same performance. Sum_Coh, on the other hand, is a purely optimization-based method using Mixed Integer Linear Programming (MILP), which is, in general, NP-Hard [14]. Its solution is, therefore, hard to interpret in an intuitive manner — and its consistency depends on the MILP solver used. Finally, Sum_Coh's original code makes use of Gurobi [15], a commercial optimization tool, whose use could potentially increase the cost of development of a commercial summarization solution. We must, then, conclude, that the Progressive algorithm is a better fit for our domain of interest.

We also note that Progressive's ROUGE-2 performance in the JPEPT dataset also surpasses Sum_Coh's reported in [5] on a similar task of technical literature using the PLOS ONE journal of medicine, of 0.09.

The Python implementations of all the above-mentioned algorithms, as well as the pre-processing scripts used to prepare their inputs and the newly created JPEPT dataset can be found in this project's public Github repository:

https://github.com/joaomcm/Automatic-Summarization-of-Technical-Literature-on-Oil-and-Gas.

We hope that this initial work and our collected dataset will help drive the research in the area towards the production of a viable commercial automatic technical literature summarizer.

ACKNOWLEDGEMENTS

The second author has been partially supported by the Conselho Nacional de Desenvolvimento Cientifico e Tecnológico (CNPq), grant 312180/2018-7. The work was also supported by the Fundação de Amparo a Pesquisa do Estado de São Paulo (FAPESP), grant 2016/18841-0, and also by the Coordenação de Aperfeiçoamento de Pessoal de Nivel Superior (CAPES) - finance code 001

REFERENCES

[1] H. Chen, R. H. L. Chiang, and V. C. Storey, "Business Intelligence and Analytics: From Big Data to Big Impact," *MIS Quarterly*, vol. 36, no. 4, p. 1165, 2012. [Online]. Available: http://www.jstor.org/stable/10.2307/41703503

[2] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[3] M. Gambhir and V. Gupta, "Recent automatic text summarization techniques: a survey," *Artificial Intelligence Review*, vol. 47, no. 1, pp. 1–66, 1 2017. [Online]. Available: https://doi.org/10.1007/s10462-016-9475-9

[4] F. Liu, J. Flanigan, S. Thomson, N. Sadeh, and N. A. Smith, "Toward Abstractive Summarization Using Semantic Representations," *ArXiv e-prints*, 5 2018.

[5] D. Parveen and M. Strube, "Integrating Importance, Non-Redundancy and Coherence in Graph-Based Extractive Summarization." in *IJCAI*, 2015, pp. 1298–1304.

[6] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," *Text Summarization Branches Out*, 2004.

[7] R. M. Alguliev, R. M. Aliguliyev, and N. R. Isazade, "Multiple documents summarization based on evolutionary optimization algorithm," *Expert Systems with Applications*, vol. 40, no. 5, pp. 1675–1689, 4 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417412010688

[8] D. Wang, S. Zhu, T. Li, and Y. Gong, "Multi-document Summarization Using Sentence-based Topic Models," in *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ser. ACLShort '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 297–300. [Online]. Available: http://dl.acm.org/citation.cfm?id=1667583.1667675

[9] D. Wang, S. Zhu, T. Li, Y. Chi, and Y. Gong, "Integrating Document Clustering and Multidocument Summarization," *ACM Trans. Knowl. Discov. Data*, vol. 5, no. 3, pp. 14:1–14:26, 8 2011. [Online]. Available: http://doi.acm.org/10.1145/1993077.1993078

[10] Y. Ouyang, W. Li, R. Zhang, S. Li, and Q. Lu, "A progressive sentence selection strategy for document summarization," *Information Processing & Management*, vol. 49, no. 1, pp. 213–221, 1 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306457312000672

[11] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 12 1997. [Online]. Available: https://doi.org/10.1023/A:1008202821328

[12] C. Guinaudeau and M. Strube, "Graph-based local coherence modeling," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2013, pp. 93–103.

[13] J. M. Kleinberg, "Authoritative Sources in a Hyperlinked Environment," *J. ACM*, vol. 46, no. 5, pp. 604–632, 9 1999. [Online]. Available: http://doi.acm.org/10.1145/324133.324140

[14] S. Burer and A. N. Letchford, "Non-convex mixed-integer nonlinear programming: A survey," *Surveys in Operations Research and Management Science*, vol. 17, no. 2, pp. 97–106, 7 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1876735412000037

[15] L. L. C. Gurobi Optimization, "Gurobi Optimizer Reference Manual," 2018. [Online]. Available: http://www.gurobi.com