

Text Summarization – Exploring Potential of PSO Usage in Extractive Algorithms

Laura Bakala
Wojciech Bogucki
Karol Pysiak

April 2022

Contents

1	Problem description	2
2	Literature	3
2.1	PSO	3
2.2	Extractive text summarization	4
2.3	Metrics	4
3	Dataset	5
3.1	Explorative Data Analysis	5
3.1.1	Duplicates	5
3.1.2	Missing values	5
3.1.3	Fixing encoding	5
3.1.4	Extracting additional information from articles	5
3.1.5	Removing noise	6
3.1.6	Length of the articles and highlights	6
3.1.7	Removing outliers	6
3.1.8	Word frequency	10
4	Proposed solution	11
5	Model description	12
6	Initial results	13
6.1	Example summaries	13
6.1.1	Turkish siege image block	13
6.1.2	Dragging a dog behind a car	14
7	Grid search results	15
8	Deployment	16
8.1	Web application	16
8.2	Containerization	17
8.3	Performance tests	17
9	Summary	17
	References	18

1 Problem description

Text summarization is a task where the contents of a longer body of text, usually spanning multiple sentences, are condensed in a short summary, one or two sentences long. There are two parts to this task: one requiring extracting the critical information from the text and the other generating sentences out of the extracted data. Due to the conceptual difficulties this task poses, this field is still dominated by humans; although the research began as early as 1958, it was only in the last decade that we saw an influx of machine learning models capable of doing this task. Moreover, text summarization may be done on single or multiple documents alike, albeit requiring different models [6].

There are two approaches to text summarization, extractive and abstractive. The former is simpler since it ignores the generative part of the challenge; extractive models focus on extracting the key phrases from the original text. Abstractive models, on the other hand, extract abstract information that is later used to generate a summary that may not necessarily use words and phrases present in the original text but which is much more human-readable than a list of key terms. What is more, some algorithms use the so-called "hybrid" approach, which feeds the results of applying an extractive model to an abstractive model. This approach has an advantage of combining advantages of extractive and abstractive approaches, but also has a drawback of limiting the information fed to the abstractive model.

Like most other problems in the NLP domain, extractive text summarization is usually done by some complex, hi-tech, deep learning models. Whereas neural networks exhibit relatively high robustness in text processing, they are tremendously computationally expensive and require vast volumes of data to work properly. We want to explore an alternative to using more traditional methods, i.e., the Particle Swarm Optimization algorithm. We will challenge both approaches on the same data and check their performance as well as their run times to consider their advantages and disadvantages evenhandedly.

2 Literature

2.1 PSO

The PSO is an optimization algorithm that can handle very complex functions among many dimensions. It is not the first algorithm that comes to mind when we think about text summarization. However, when we properly define a target function that will reliably score the summary's quality, the PSO comes into play. Many researchers have proven that, while being simple, PSO can still be robust in this domain.

There are quite a few variations of PSO implemented for extractive text summarization [11][12][16][17][18]. Although different versions introduce many novel and ingenious modifications, the algorithm's core stays the same - particles fly through the search space trying to find the optimal value of a target function.

Unfortunately, summarization search space is not conventional for PSO. It is impossible to define a continuous space with words or sentences. What we can use here is the binary search space for PSO. It works similarly to the continuous one. We only exchange the position update step for the mapping step from the continuous velocity to the binary position change as follows:

$$p_{ij} = \begin{cases} 1, & \text{if } \text{rand}_j < \text{sigm}(v_{ij}(t+1)), \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where rand_j is the random number selected from a uniform distribution over $[0, 1]$ and:

$$\text{sigm}(v_{ij}(t+1)) = \frac{1}{1 + \exp(-v_{ij}(t+1))} \quad (2)$$

A target function may be – for example – defined by four metrics that define sentences properties. They help us choose the most informative sentences, keeping the length of the summarization to some reasonable limits:

- Title feature – sentence resemblance to the title

$$tf(s) = \frac{\text{Keywords in } s \cap \text{Keywords in Title}}{\text{Keywords in } s \cup \text{Keywords in Title}}$$

- Keywords – words being used repetitively in the document are considered important. In order to calculate keyword score:

$$\text{Score}(w) = f(w) \times \frac{\log N}{S(w)}$$

where $f(w)$ is the frequency of the word in a sentence, N is the number of words in that particular sentence, and $S(w)$ is the number of sentences in which the word occurs.

- Similarity to First and Last Sentence

$$S_f(s) = \frac{s_1 + s_2}{2}$$

where

$$s_1 = \frac{\text{Keyword in } s \cap \text{Keywords in First Sentence}}{\text{Keywords in } s \cup \text{Keywords in First Sentence}}$$

$$s_2 = \frac{\text{Keyword in } s \cap \text{Keywords in Last Sentence}}{\text{Keywords in } s \cup \text{Keywords in Last Sentence}}$$

- Sentence Cohesion – sentences that have continuity are extracted to ensure a continuous summarized text flow and to reduce redundancy:

$$Sc(s) = \frac{\text{Keywords in } s \cap \text{Keywords in another Sentence}}{\text{Keywords in } s \cup \text{Keywords in another Sentence}}$$

The target function is now just a sum of all these measures on a specific sentence.

Details of the whole PSO application in text summarization are described in the article [1]. Although we base on existing solutions, we are open to implementing new, innovative ideas.

2.2 Extractive text summarization

According to website <https://paperswithcode.com/sota/extractive-document-summarization-on-cnn>, currently HAHSum model [5] is the state-of-the-art of extractive summarization on CNN/Daily Mail benchmark. HAHSum stands for **H**ierarchical **A**ttentive **H**eterogeneous Graph for Text **S**ummarization. It uses a redundancy-aware graph to refine sentence representations. It consists of ALBERT Encoder, Abstract Layer, Redundancy Layer, and Output Layer [5].

The second state-of-the-art model is MatchSum [19] where the authors formulate the extractive summarization task as a semantic text matching problem, in which a source document and candidate summaries will be (extracted from the original text) matched in a semantic space.

Other state-of-the-art model is NeRoBERTa [7] where authors propose a nested tree-based extractive summarization model on RoBERTa (NeRoBERTa), where nested tree structures consist of syntactic and discourse trees in a given document.

More importantly for our research, there are available pre-trained state-of-the-art models for extractive summarization such as TransformerSum <https://github.com/HHousen/TransformerSum> which achieves 94% of the performance of MatchSum or BERTSumExt [9] available at <https://github.com/nlpyang/PreSumm>.

As most of the models use BERT as sentence encoder, there is also a possibility to use pre-trained BERT model [3] and combine it with clustering to obtain the best summarization as it was done for lectures summarization in [10].

2.3 Metrics

There are several challenges to evaluating summary quality. The key aspects are as follows:

- amount of key information extracted and included in the summary,
- as low redundancy as possible,
- summary coherence, i.e. sentences being grammatically correct and easily understandable,
- the key information being joined in a way that resembles the original meaning (e.g. correctly placed negations).

However, there is no metric that would evaluate all these aspects. Currently, the accepted standard metric for scoring text summaries is ROUGE (Recall-Oriented Understudy for Gisting Evaluation). It is a family of metrics: ROUGE-N metrics evaluate the overlap of N-grams (subsequences of elements, in this case – words) between a summary prediction and the reference summary, ROUGE-L and ROUGE-W base on longest common subsequence, ROUGE-S and ROUGE-SU use skip-bigrams with arbitrary skip sizes [8].

There are more metrics, though, that use n-grams, like BLEU [14] or NIST [4]. There are frameworks like QARLA, too, which combine a set of metrics into a scoring system [2].

3 Dataset

For this project we use CNN/Daily Mail dataset available at Kaggle.com [13]. It is an English-language dataset consisting of news articles written by CNN and Daily Mail journalists and highlight sentences from these articles.

The data consists of three variables [13]:

- **id**: a string containing the hexadecimal formatted SHA1 hash of the URL where the story was retrieved from
- **article**: a string containing the body of the news article
- **highlights**: a string containing the highlight of the article as written by the article author

The dataset is split into three sub-datasets – train, validation and test.

3.1 Explorative Data Analysis

3.1.1 Duplicates

During analysis of textual data we found duplicated articles and highlights. Interestingly, duplicated records had different IDs, which made them harder to find. Sizes of sub-datasets before and after removing duplicates are presented in Table 1.

Table 1: Sub-datasets overview.

Sub-dataset	Number of rows	Number of rows after removing duplicates
train	287 113	282 197
validation	13 368	13 300
test	11 490	11 449

3.1.2 Missing values

The sub-datasets do not contain any missing values.

3.1.3 Fixing encoding

After loading data, there were Unicode artifacts in text of articles and highlights e.g. ”
xa0”. They were normalized using function `normalize` from package `unicodedata`.

3.1.4 Extracting additional information from articles

Many articles had information about authorship, news station and date of publication at the beginning of the text. This information is obsolete for text summarization, but can give further insight about the data. Three news stations that frequently appeared in text were – CNN, Associated Press and Daily Mail. The columns describing if article was created by certain news station were added. Information about cardinality of each news station’s articles is presented in Table 2.

Table 2: Authorship of the articles.

News station	Number of articles
CNN	72 756
Associated Press	2 836
Daily Mail	13 958

3.1.5 Removing noise

After extracting information about the authorship of an article, we decided to remove such additional information from article text as it is not relevant for the purpose of summarization.

Four most commonly appearing formulas were removed using regular expressions:

```
"(By \. [A-Za-z ]+ \. )"
"(PUBLISHED: \. \d{2}:\d{2} \w+, \d+ \w+ \d{4} \. )"
"(\| \. UPDATED: \. \d{2}:\d{2} \w+, \d+ \w+ \d{4} \. )"
"(\(CNN\)s*-- )"
```

For example, an article which looked like this:

```
By . Associated Press . PUBLISHED: . 14:11 EST, 25 October 2013 . — . UPDATED:
. 15:36 EST, 25 October 2013 . The bishop of the Fargo Catholic Diocese in North
Dakota has exposed potentially hundreds of church members...
```

after removing noise looks like this:

```
The bishop of the Fargo Catholic Diocese in North Dakota has exposed potentially
hundreds of church members...
```

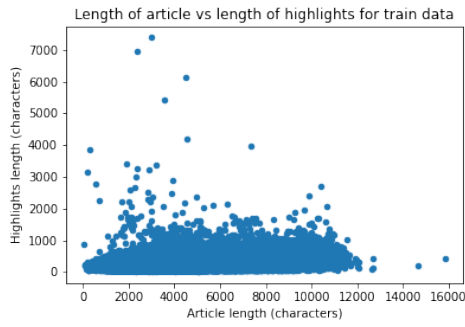
3.1.6 Length of the articles and highlights

To better understand how lengths of articles and highlights are distributed, we calculated number of characters in a text. Later, after extracting sentences and words, we calculated count metrics for them. For two latter metrics, we used NLTK package in Python.

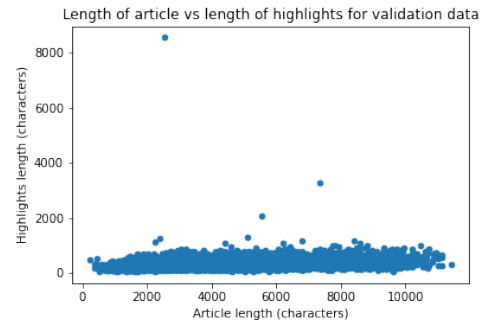
3.1.7 Removing outliers

To determine if there are outliers in data, which in this case would be too long or too short articles, we examined scatter plots (Figure 1) and histograms of lengths of articles and highlights (Figure 2)

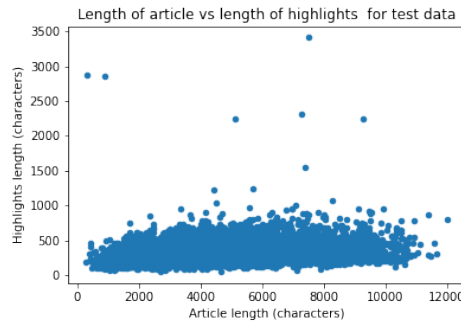
As can be seen on Figure 1b, there are examples of records, where highlights are longer than the related article. Such cases are probably the result of faulty parsing of text or mistaking the article and its highlights. To avoid these outliers, we decided to remove records where highlights are longer than half of the length of the related article [15]. As a result, 134 rows were removed from training dataset (10 from validation and 12 from test dataset).



(a) Comparison of lengths of article and its highlights for training data.

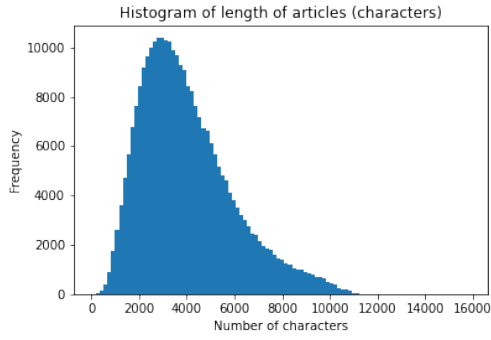


(b) Comparison of lengths of article and its highlights for validation data.

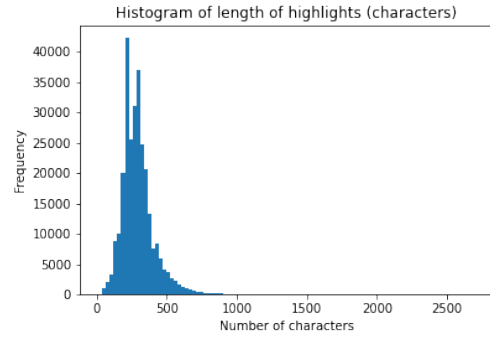


(c) Comparison of lengths of article and its highlights for test data.

Figure 1: Comparison of lengths of article and its highlights. As we can see, in every sub-dataset there are outliers – very long articles or highlights.



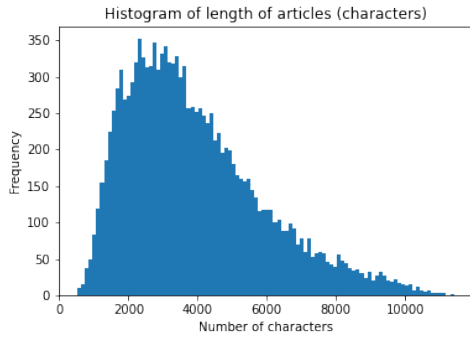
(a) Histogram of length of article in number of characters.



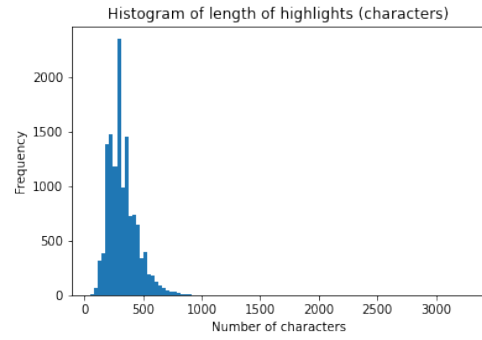
(b) Histogram of length of highlights in number of characters.

Figure 2: Distribution of length of articles and highlights in training data. Both distributions are very skew. This may be caused by some very long articles and highlights.

Similar distributions are observed for validation (figure 3) and test data (figure 4).

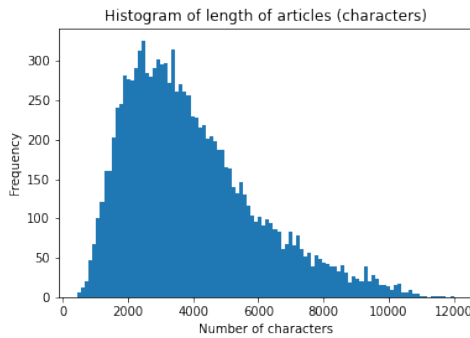


(a) Histogram of length of article in number of characters.

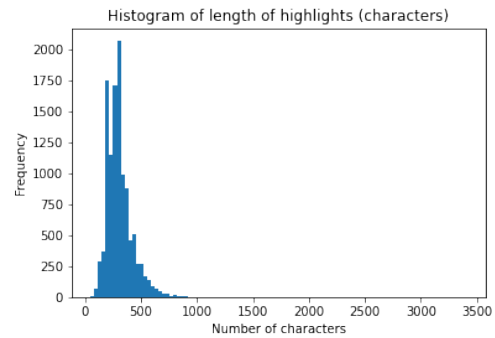


(b) Histogram of length of highlights in number of characters.

Figure 3: Distribution of length of articles and highlights in validation data.



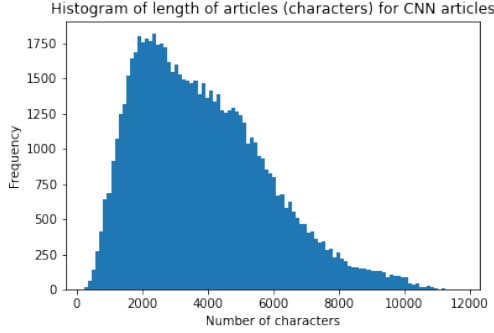
(a) Histogram of length of article in number of characters.



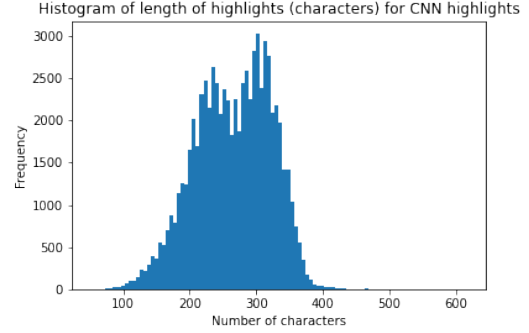
(b) Histogram of length of highlights in number of characters.

Figure 4: Distribution of length of articles and highlights in test data. Both distributions are very skew. This may be caused by some very long articles and highlights.

We also checked distributions of lengths for articles and highlights from CNN using a previously created feature. Results are presented on Figure 5. Interestingly, distribution of lengths of articles differs from distribution of lengths of highlight, indicating that longer article not always implicate longer highlights.



(a) Histogram of length of article in number of characters.



(b) Histogram of length of highlights in number of characters.

Figure 5: Distribution of length of articles and highlights created by CNN. Almost all highlights from CNN are shorter than 400 characters.

3.1.8 Word frequency

We also calculated word frequency for validation data to speed up computations. Frequency count was conducted as follows:

1. Articles were split to tokens using `word_tokenizer`
2. For every word, all punctuation was removed
3. Every word was converted to lowercase
4. Every word was lemmatized using `WordNetLemmatizer` with tags denoting what part of speech is a certain word
5. Stop words were removed
6. Number of occurrences for every word was calculated
7. Overall frequency for every word was calculated

The most popular words are: say, year, take, make, go, time, get, also, last, tell, leave, people. The most popular words are also presented in a form of world cloud (Figure 6).



Figure 6: Word cloud for validation data.

4 Proposed solution

We plan to implement an extractive text summarization algorithm with Particle Swarm Optimization. The set of features used will be taken from the subspace of all possible sets of features such that there doesn't exist a text summarization algorithm that uses these exact features. We'll use existing algorithms' scores and sets of features to predict the optimal feature set for our algorithm (or, possibly, a few feature sets to choose from). The algorithm we'll generate this way shall be compared against several state-of-the-art extractive text summarization algorithms, both with and without PSO. The latter are mainly to provide the context for achieved scores.

Moreover, we can use the output of the best extractive method with PSO as an input to a suitable abstractive text summarization algorithm, creating a so-called hybrid text summarization. As there are no hybrid approaches known to us that utilize PSO, this could answer the question whether PSO would work better for hybrid algorithms than the existing implementations.

5 Model description

Our model consists of a sentence transformer and a PSO module that optimizes over sentence space.

The sentence transformer takes articles as an input and splits them into separate sentences. Then, each sentence is embedded with a transformer, giving a vector of numbers describing sentence meaning. The transformer component itself is changeable, allowing us to use different transformer algorithms, such as those trained on specific topic as bioinformatics or economics – which would supposedly perform better on articles in the respective field.

We denote an article (document) as D . An article is divided into sentences $D = \{s_1, s_2, \dots, s_{|D|}\}$. Embedding of a sentence s_i is denoted as $e(s_i)$.

These embeddings are then compared using cosine distance, giving a matrix of distances between sentences, immediately transformed to a matrix of similarities using simple $sim(e(s_i), e(s_j)) = 1 - dist(e(s_i), e(s_j))$ transformation. Cosine distance for two sentences is defined as follows:

$$dist(s_i, s_j) = 1 - \frac{e(s_i) \cdot e(s_j)}{\|e(s_i)\|_2 \|e(s_j)\|_2} \quad (3)$$

We used `pdist` function from `scipy` to calculate cosine distance more effectively.

Furthermore, all sentences are compared to the text of the whole article with cosine distance as well, approximating the similarity of a sentence to the article topic.

Then, PSO takes these similarity scores and uses them during optimization process. PSO optimizes the following target function:

$$\text{maximize } f = \text{capacity} \cdot g - (1 - \text{capacity}) \cdot h \quad (4)$$

$$g = \sum_{i=1}^{n-1} \sum_{j=i+1}^n [sim(e(D), e(s_i)) + sim(e(D), e(s_j)) - sim(e(s_i), e(s_j))] \cdot x_{ij} \quad (5)$$

$$h = \min(L - \sum_{i=1}^{n-1} \sum_{j=i+1}^n x_{ij}, 0) \quad (6)$$

$$x_{ij} \in \{0, 1\}, \forall i, j \quad (7)$$

$$L - \text{max sentence count limit} \quad (8)$$

As a result, PSO chooses an optimal subset of sentences from the article. These sentences are then concatenated and returned by a summarization function.

6 Initial results

We compared our initial implementation of PSO against MatchSum model [19], using various metrics in ROUGE family. In Table 3 there are average F scores as computed by `pyrouge`.

Table 3: Comparison of initial PSO implementation and MatchSum model.

Metric	PSO score	MatchSum score
ROUGE-1	0.34204	0.43960
ROUGE-2	0.12963	0.20497
ROUGE-3	0.06942	0.12028
ROUGE-4	0.04475	0.08070
ROUGE-L	0.27855	0.40172
ROUGE-W-1.2	0.16078	0.22755
ROUGE-S*	0.09735	0.16051
ROUGE-SU*	0.10382	0.16949

Unfortunately, our PSO implementation scored much worse than MatchSum in each metric. However, there’s still a large room for improvement.

6.1 Example summaries

6.1.1 Turkish siege image block

This is the comparison of summaries generated by our PSO and MatchSum on one of the articles. Due to the length of the article, we don’t cite it here. Instead, we cite the reference summary:

turkish court imposed blocks as images of siege shared on social media.
images 'deeply upset' wife and children of hostage mehmet selim kiraz.
prosecutor, 46, died in hospital after hostages stormed a courthouse.
two of his captors were killed when security forces took back the building.

This is the summary of MatchSum, consisting of three longer sentences; the model required heavy modification of input sentences to make it more readable for the machine:

turkey has blocked access to twitter and youtube after they refused a request to remove pictures of a prosecutor held during an armed siege last week.
a turkish court imposed the blocks because images of the deadly siege were being shared on social media and 'deeply upset' the wife and children of mehmet selim kiraz, the hostage who was killed.
the 46-year-old turkish prosecutor died in hospital when members of the revolutionary people’s liberation party-front (dhkp-c) stormed a courthouse and took him hostage.

This, for the comparison, is the summary of our PSO. Formatting issues are due to the original text being already transformed; PSO doesn’t change formatting of the input sentences and could work with original sentences.

turkey has blocked access to twitter and youtube after they refused a request to remove pictures of a prosecutor held during an armed siege last week.
mr kiraz, a father-of-two married to a judge who also worked at the courthouse, was targeted for his part in an investigation into the death of berkin elvan.
gathering: prosecutors, lawyers and judges stand near a statue of lady justice during the funeral ceremony a british national, of polish origin but who has not been named, was arrested on saturday as part of an operation against the revolutionary people’s liberation party-front, according to reports.
a foreign office spokeswoman said this morning: 'i can confirm that a british national has been arrested in turkey and that we are offering consular assistance'

6.1.2 Dragging a dog behind a car

We can compare the outputs of compared models with a human-friendly target summary, as shown below:

WARNING: GRAPHIC CONTENT - Deborah Fuller dragged her one-year-old dog behind her car for 400 metres at a speed of around 30mph.
Animal was left with wounds to paws and elbow and grazes on his stomach.
Fuller, a breeder, failed to quickly take the animal to vet which could have reduced the 'unnecessary suffering' of the Rhodesian ridgeback.
She was banned from keeping animals for five years and given a curfew.

MatchSum required de-formatting the text and returned these lowercase sentences (with spaces on both sides of all punctuation, which we removed for readability):

deborah fuller, 56, dragged her dog tango for 400 metres behind her car as she drove along the b1066 near long melford, essex.
the rhodesian ridgeback was left with injuries to all four paws as well as grazing to his chest and a deep wound on his elbow.
fuller has also been banned from keeping animals for five years and had her 27 other dogs confiscated.

In case the input isn't needlessly transformed, our implementation of PSO returns sentences in original format, rendering the summary much more readable.

Tree surgeons working nearby spotted what had happened and tried to get Fuller, who is a Rhodesian Ridgeback breeder and exhibitor, to stop her car.
RSPCA inspector Sam Garvey said after the sentencing: 'Eyewitnesses report they saw the dog being dragged about 400 metres at a speed of 30mph.
Judges ruled that the dog's suffering could have been avoided if Fuller had taken him to the vets earlier.

Further tests shall verify whether difference in formatting influences the scores given by ROUGE metrics.

7 Grid search results

In order to search for the best results, our model was tested with several transformers for text embedding and several values of `capacity` parameter. The following transformers were checked:

- MiniLM-L6-v2 (Table 4)
- MiniLM-L12-v2 (Table 5)
- distilroberta-v1 (Table 6)
- mpnet-base-v2 (Table 7)

Table 4: Comparison of different `capacity` values for MiniLM-L6-v2 transformer.

capacity	0.1	0.3	0.5
ROUGE-1	0.34204	0.31892	0.32121
ROUGE-2	0.12963	0.11737	0.12780
ROUGE-3	0.06942	0.06205	0.06899
ROUGE-4	0.04475	0.03992	0.04469
ROUGE-L	0.27855	0.25861	0.26169
ROUGE-W-1.2	0.16078	0.15036	0.15536
ROUGE-S*	0.09735	0.08526	0.08484
ROUGE-SU*	0.10382	0.09098	0.08958

Table 5: Comparison of different `capacity` values for MiniLM-L12-v2 transformer.

capacity	0.1	0.3	0.5
ROUGE-1	0.32183	0.32312	0.32384
ROUGE-2	0.12062	0.12122	0.13002
ROUGE-3	0.06430	0.06470	0.07117
ROUGE-4	0.04138	0.04157	0.04621
ROUGE-L	0.26250	0.26319	0.26491
ROUGE-W-1.2	0.15272	0.15315	0.15753
ROUGE-S*	0.08704	0.08755	0.08639
ROUGE-SU*	0.09286	0.09338	0.09125

Table 6: Comparison of different `capacity` values for distilroberta-v1 transformer.

capacity	0.1	0.3	0.5
ROUGE-1	0.31161	0.31231	0.31300
ROUGE-2	0.11147	0.11206	0.11891
ROUGE-3	0.05801	0.05861	0.06302
ROUGE-4	0.03688	0.03769	0.04053
ROUGE-L	0.25233	0.25286	0.25319
ROUGE-W-1.2	0.14661	0.14692	0.14993
ROUGE-S*	0.08101	0.08128	0.08009
ROUGE-SU*	0.08657	0.08686	0.08472

Table 8 shows results of the model for each transformer with the best capacity value. The model with transformer MiniLM-L6-v2 had the best ROUGE-1 and ROUGE-L scores, which are usually the most representative as summary quality indicator. Interestingly, transformer MiniLM-L12-v2 had better results for ROUGE-2, ROUGE-3 and ROUGE-4.

Table 7: Comparison of different capacity values for mpnet-base-v2 transformer.

capacity	0.1	0.3	0.5
ROUGE-1	0.31423	0.31451	0.31598
ROUGE-2	0.11310	0.11378	0.12263
ROUGE-3	0.05930	0.05960	0.06564
ROUGE-4	0.03789	0.03835	0.04233
ROUGE-L	0.25458	0.25491	0.25614
ROUGE-W-1.2	0.14784	0.14806	0.15209
ROUGE-S*	0.08249	0.08266	0.08103
ROUGE-SU*	0.08813	0.08827	0.08566

Table 8: Comparison of the best capacity values for different transformers.

transformer capacity	MiniLM-L6-v2 0.1	MiniLM-L12-v2 0.5	distilroberta-v1 0.5	mpnet-base-v2 0.5
ROUGE-1	0.34204	0.32121	0.31300	0.31598
ROUGE-2	0.12963	0.13002	0.11891	0.12263
ROUGE-3	0.06942	0.07117	0.06302	0.06564
ROUGE-4	0.04475	0.04621	0.04053	0.04233
ROUGE-L	0.27855	0.26491	0.25319	0.25614
ROUGE-W-1.2	0.16078	0.15753	0.14993	0.15209
ROUGE-S*	0.09735	0.08639	0.08009	0.08103
ROUGE-SU*	0.10382	0.09125	0.08472	0.08566

8 Deployment

Our model is deployed in the form of a web application inside a Docker container. Web application can be run locally on port 8000 and its layout is presented on Figure 7. User can paste text of an article in the upper text area, click *Summarize* button and summary of this article will appear in the lower text area.

8.1 Web application

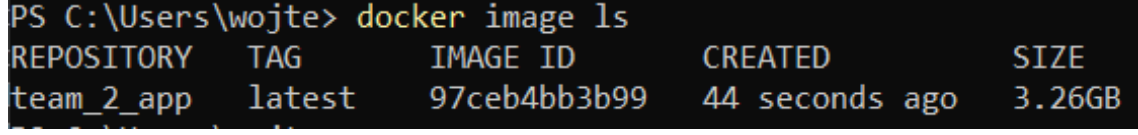
Web application is build with FastAPI and uses created model for text summarization with PSO.



Figure 7: Layout of the web application.

8.2 Containerization

Web application is containerized using Docker and docker-compose. Image can be build by simply running command `docker-compose up` in project directory.

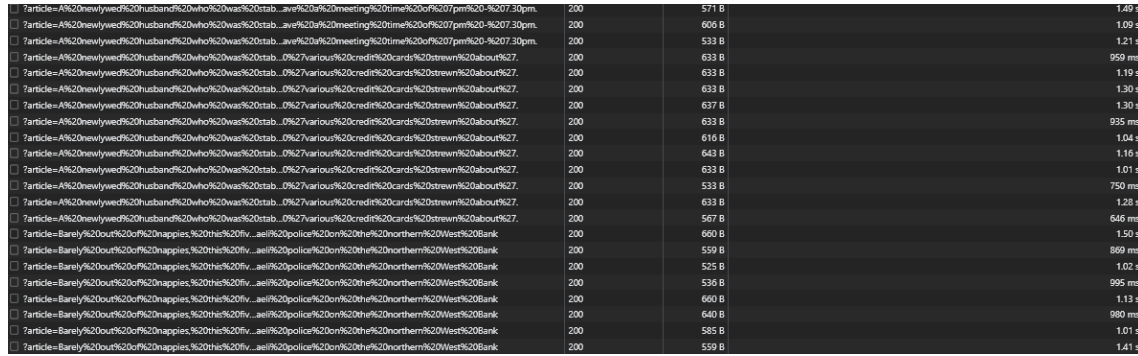


REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
team_2_app	latest	97ceb4bb3b99	44 seconds ago	3.26GB

Figure 8: Size of Docker image.

8.3 Performance tests

Performance tests are presented on Figure 9. As it is shown, requests usually take around 1-1.5 seconds to return a summarization for a medium-sized article.



?article=AF%20newlywed%20husband%20who%20was%20stab...ave%20a%20meeting%20time%20of%207pm%20-%207.30pm.	200	571 B	1.49 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...ave%20a%20meeting%20time%20of%207pm%20-%207.30pm.	200	606 B	1.09 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...ave%20a%20meeting%20time%20of%207pm%20-%207.30pm.	200	533 B	1.21 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	633 B	959 ms	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	633 B	1.19 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	633 B	1.30 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	637 B	1.30 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	633 B	935 ms	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	616 B	1.04 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	643 B	1.16 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	633 B	1.01 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	533 B	750 ms	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	633 B	1.28 s	
?article=AF%20newlywed%20husband%20who%20was%20stab...%27various%20credit%20cards%20strewn%20about%27.	200	567 B	646 ms	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	660 B	1.50 s	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	559 B	869 ms	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	525 B	1.02 s	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	536 B	995 ms	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	660 B	1.13 s	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	640 B	980 ms	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	585 B	1.01 s	
?article=Barely%20our%20of%20nappies,%20this%20f...ae%20police%20on%20thef%20northern%20West%20Bank	200	559 B	1.41 s	

Figure 9: Performance test for example summarization.

9 Summary

Text summarization with PSO is possible. It works quite well, but is outperformed by modern NLP algorithms. Improvements are for sure possible. One way is to enhance the optimization capabilities of PSO. There are many versions of this algorithm, which are more complex but may produce better results. These experiments employed quite simple PSO version, so trying out the latest modifications could be promising. Another way is to redefine the target function. Although, the idea of using embeddings is quite novel for the PSO text summarization, it needs more testing and tweaking to assure its quality. Summing up, the PSO is not very promising, compared to modern text summarization methods, but when constructed properly can still be useful in some use cases.

References

- [1] Rasim Alguliyev et al. “MCMR: Maximum coverage and minimum redundant text summarization model”. In: *Expert Systems with Applications* 38 (Nov. 2011), pp. 14514–14522. DOI: 10.1016/j.eswa.2011.05.033.
- [2] Enrique Amigó et al. “QARLA: A Framework for the Evaluation of Text Summarization Systems”. In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. ACL ’05. Ann Arbor, Michigan: Association for Computational Linguistics, 2005, pp. 280–289. DOI: 10.3115/1219840.1219875. URL: <https://doi.org/10.3115/1219840.1219875>.
- [3] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [4] George Doddington. “Automatic Evaluation of Machine Translation Quality Using N-Gram Co-Occurrence Statistics”. In: *Proceedings of the Second International Conference on Human Language Technology Research*. HLT ’02. San Diego, California: Morgan Kaufmann Publishers Inc., 2002, pp. 138–145.
- [5] Ruipeng Jia et al. “Neural extractive summarization with hierarchical attentive heterogeneous graph network”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 3622–3631.
- [6] Wafaa S. El-Kassas et al. “Automatic text summarization: A comprehensive survey”. In: *Expert Systems with Applications* 165 (2021), p. 113679. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2020.113679>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417420305030>.
- [7] Jingun Kwon et al. “Considering Nested Tree Structure in Sentence Extractive Summarization with Pre-trained Transformer”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021, pp. 4039–4044.
- [8] Chin-Yew Lin. “ROUGE: A Package for Automatic Evaluation of Summaries”. In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 74–81. URL: <https://aclanthology.org/W04-1013>.
- [9] Yang Liu and Mirella Lapata. “Text summarization with pretrained encoders”. In: *arXiv preprint arXiv:1908.08345* (2019).
- [10] Derek Miller. “Leveraging BERT for extractive text summarization on lectures”. In: *arXiv preprint arXiv:1906.04165* (2019).
- [11] Mohamed Atef Mosa. “A novel hybrid particle swarm optimization and gravitational search algorithm for multi-objective optimization of text mining”. In: *Applied Soft Computing* 90 (2020), p. 106189. ISSN: 1568-4946. DOI: <https://doi.org/10.1016/j.asoc.2020.106189>. URL: <https://www.sciencedirect.com/science/article/pii/S1568494620301290>.
- [12] Mohamed Atef Mosa, Arshad Syed Anwar, and Alaa Hamouda. “A survey of multiple types of text summarization with their satellite contents based on swarm intelligence optimization algorithms”. In: *Knowledge-Based Systems* 163 (2019), pp. 518–532. ISSN: 0950-7051. DOI: <https://doi.org/10.1016/j.knosys.2018.09.008>. URL: <https://www.sciencedirect.com/science/article/pii/S0950705118304593>.
- [13] Gowri Shankar P. *CNN-DailyMail News Text Summarization — Kaggle*. URL: <https://www.kaggle.com/gowrishankarp/newspaper-text-summarization-cnn-dailymail>. (accessed: 22.03.2022).
- [14] Kishore Papineni et al. “Bleu: a Method for Automatic Evaluation of Machine Translation”. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: <https://aclanthology.org/P02-1040>.
- [15] Dragomir Radev, Eduard Hovy, and Kathleen McKeown. “Introduction to the special issue on summarization”. In: *Computational linguistics* 28.4 (2002), pp. 399–408.

- [16] Rasmita Rautray and Rakesh Chandra Balabantaray. “Cat swarm optimization based evolutionary framework for multi document summarization”. In: *Physica A: Statistical Mechanics and its Applications* 477 (2017), pp. 174–186. ISSN: 0378-4371. DOI: <https://doi.org/10.1016/j.physa.2017.02.056>. URL: <https://www.sciencedirect.com/science/article/pii/S0378437117302121>.
- [17] Jesus M. Sanchez-Gomez, Miguel A. Vega-Rodríguez, and Carlos J. Pérez. “Extractive multi-document text summarization using a multi-objective artificial bee colony optimization approach”. In: *Knowledge-Based Systems* 159 (2018), pp. 1–8. ISSN: 0950-7051. DOI: <https://doi.org/10.1016/j.knosys.2017.11.029>. URL: <https://www.sciencedirect.com/science/article/pii/S0950705117305609>.
- [18] Minakshi Tomer and Manoj Kumar. “Multi-document extractive text summarization based on firefly algorithm”. In: *Journal of King Saud University - Computer and Information Sciences* (2021). ISSN: 1319-1578. DOI: <https://doi.org/10.1016/j.jksuci.2021.04.004>. URL: <https://www.sciencedirect.com/science/article/pii/S1319157821000902>.
- [19] Ming Zhong et al. “Extractive summarization as text matching”. In: *arXiv preprint arXiv:2004.08795* (2020).