ПРОГРЕСИВНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

PROGRESSIVE INFORMATION TECHNOLOGIES

UDC 004.93, 004.8

METHOD OF IMPERATIVE VARIABLES FOR SEARCH AUTOMATION OF TEXTUAL CONTENT IN UNSTRUCTURED DOCUMENTS

Boiko V. O. – Assistant of the Department of Software Engineering, Khmelnytskyi National University, Khmelnytskyi, Ukraine.

ABSTRACT

Context. Currently, there are a lot of approaches that are used for textual search. Nowadays, methods such as pattern-matching and optical character recognition are highly used for retrieving preferred information from documents with proven effectiveness. However, they work with a common or predictive document structure, while unstructured documents are neglected. The problem – is automating the textual search in documents with unstructured content. The object of the study was to develop a method and implement it into an efficient model for searching the content in unstructured textual information.

Objective. The goal of the work is the implementation of a rule-based textual search method and a model for seeking and retrieving information from documents with unstructured text content.

Method. To achieve the purpose of the research, the method of rule-based textual search in heterogenous content was developed and applied in the appropriately designed model. It is based on natural language processing that has been improved in recent years along with a new generative artificial intelligence becoming more available.

Results. The method has been implemented in a designed model that represents a pattern or a framework of unstructured textual search for software engineers. The application programming interface has been implemented.

Conclusions. The conducted experiments have confirmed the proposed software's operability and allow recommendations for use in practice for solving the problems of textual search in unstructured documents. The prospects for further research may include the improvement of the performance using multithreading or parallelization for large textual documents along with the optimization approaches to minimize the impact of OpenAI application programming interface content processing limitations. Furthermore, additional investigation might incorporate extending the area of imperative variables usage in programming and software development.

KEYWORDS: textual search, unstructured text documents, natural language processing, rule-based search, generative artificial intelligence, imperative variables.

ABBREVIATIONS

OCR is an optical character recognition;
API is an application programming interface;
ZOCR is a zonal optical character recognition;
NLP is a natural language processing;
AI is an artificial intelligence;
GPT is a generative pre-trained transformer.

NOMENCLATURE

T is a set of unstructured text documents;

R is a set of search rules (prompts);

D is a description of a purpose for the model;

X is input data that consists of unstructured text documents, search rules, purpose descriptions, and sample document content;

Y' is output data that represents extracted data points from text documents;

f is a general representation of a function that performs data extraction based on input parameters;

Y is a sample response, ground truth;

$L(Y,Y')$ is a Cross-Entropy Loss function;

N is a number of variables or data points to be predicted in each example;

M is the size of the whole dataset;

$y'_{i,j}$ is a predicted value for the j-th data point in the i-th example;

$Y'_{i:i-1}$ is a previously generated set of tokens;

$P()$ is a probability distribution of the subsequent token;

θ is a model parameters;

$\nabla L(\theta)$ is a gradient of the loss function L with respect to the model parameters θ;

$g^{(t)}$ is a gradient of the loss function evaluated at time step t;

$m^{(t)}$ is a first-moment estimate at time step t;

β_1 is a decay rate for the first-moment estimate;

$v^{(t)}$ is a second-moment estimate at time step t;

DOI 10.15588/1607-3274-2024-2-12

β_2 is a decay rate for the second-moment estimate;

$\hat{m}^{(t)}$ is a correction of bias in the first-moment estimate at time step *t*;

$\hat{v}^{(t)}$ is a correction of bias in the second-moment estimate at time step *t*;

$\alpha^{(t)}$ is an adaptive learning rate;

ε is a constant to prevent dividing by zero;

$GPT()$ is a function that performs text generation based on input parameters using a GPT-3.5 Turbo model.

INTRODUCTION

Text searching is a widespread and basic operation for working with document content. The most popular text processing software such as Microsoft Word, PDF Reader, etc. incorporates the standard seeking algorithms into their search capabilities like a keyword or pattern-based search.

On the other hand, they are not able to work with pictures – search and retrieve information from them. In this case, the optical character recognition method could help find the appropriate text and additionally categorize it properly [1].

However, when unstructured documents are taken into account, the above-listed methods will not work, because, for keyword-based, pattern-matching search, or even OCR, the predefined document structure is required, otherwise, the results will be smooth and inaccurate.

The object of study is the process of textual search in documents with unstructured content.

The subject of study is the methods for searching and retrieving information from textual documents.

The known text search approaches and algorithms, described by authors and outlined as a part of different areas of implementation [2, 3] and [5, 6] are inappropriate and not suitable for unstructured textual document processing.

However, several studies [7–12] outline the approach based on NLP to seek appropriate data in documents related to specific areas, but these approaches are not described as a general method of searching data in unstructured documents.

The purpose of the work is to develop a method and incorporate it into an efficient and generic model for textual search in documents without a predefined structure that would be possible to use by software engineers as a framework.

1 PROBLEM STATEMENT

Suppose we have a set of unstructured text documents T, a list of search rules R, and a description D of a field that represents a user context where to find the appropriate information. Text information from documents, descriptions, and criteria are considered a set of tokens which means it could be a different type of textual content, such as a sequence of symbols, words, or sentences.

The task is to develop a method that will perform an accurate rule-based search to find an appropriate set of data points Y' in unstructured documents. The quality of response and performance should not depend on the number of rules and documents that should be processed. This can be represented by the following model:

$$\begin{aligned} &X = \{T, R, D\} \\ &f : X \to Y' \end{aligned}, \qquad (1)$$

where the function f from input X generates an output Y' which represents the found data.

Additionally, a generic search model should be designed and the method of rule-based search should be implemented in the model that will be used to construct a convenient API.

2 REVIEW OF THE LITERATURE

A standard keyword search is usually performed using algorithms like Rabin-Karp or Knuth-Morris-Pratt. These algorithms are often utilized to develop frameworks that detect plagiarism in text documents as described in the study [2]. However, they are not effective in rule-based search, as they can only identify specific patterns of text based on explicitly specified key phrases. Therefore, these algorithms are not suitable for tasks that require more advanced search techniques.

Pattern Matching Search, also known as Regex Search, is a powerful tool that allows for flexible string matching by describing complex patterns. It is widely supported across different programming languages, as it is built into text processing libraries. In a certain publication [3], the authors combined regular expressions with keyword searches to improve web search results. By using keywords as criteria or rules, fragments of information found through pattern matching can be considered as either the criteria value or as a result of the defined criteria. This method provides an effective criteria-based search.

The methods mentioned earlier are useful only if the documents contain text information. However, they cannot be employed for extracting text from images or documents that have only images with text (for instance, scanned copies of pages in PDF format). In such circumstances, the OCR technique serves as a viable alternative for seeking and extracting textual information.

In recent years, many services have emerged that provide the ability to extract textual information from images through API. One such service is Azure OCR, which has gained popularity due to its capability to recognize both printed and handwritten text from images and to distribute information based on the contextual understanding of the document [4]. Other services, including Google Cloud Vision, Amazon Textract, and Tesseract OCR, also offer similar functionality for extracting textual information from images. Furthermore, recent research studies have highlighted the significance of word processing via OCR for historical documents [5]. These studies have demon-

© Boiko V. O., 2024
DOI 10.15588/1607-3274-2024-2-12

strated that post-processing techniques can be applied to improve the accuracy and reliability of the OCR results.

The effectiveness of the OCR method for document processing is dependent on the selection of relevant criteria for data extraction. In scenarios where the criteria yield only a small amount of data while a document is voluminous, the OCR approach becomes ineffective, and the processing of the document may require significant memory or technical solutions aimed at reducing the system load. To address this challenge, the Zonal OCR approach has been developed. Unlike the standard OCR, which processes the entire document, ZOCR narrows the areas of text recognition to specific fragments where data extraction is required, thus avoiding the need to process all the text information in the document. In the paper [6], the underlying principles of ZOCR's smart parsing of documents are described. Modern OCR services, discussed earlier, have ZOCR support, making them effective tools for zonal character recognition.

These methods for finding textual information are useful and effective when predefined patterns and criteria are present. Text search based on regular expressions can identify similar character sequences, while optical character recognition is able to recognize characters and categorize text into appropriate groups using automated algorithms.

Both methods have common issues that become apparent when modifications are made to the current implementation. In the first case, developers must incorporate pattern-matching logic based on new business requirements, which may entail defining new or modifying existing search criteria. In the case of regular expressions, some selection rules may not be implementable through Regex. Therefore, an additional search logic alongside the existing one may be required. When considering this issue in the context of optical character recognition, introducing new search criteria may raise the question of retraining the existing model. Instead, a standardized approach may involve scanning the entire document for textual information and applying a pattern-matching search, which again brings us back to the above problem.

In a study [7], a solution encountered in extracting complex text structures, tabular information, and text located in different places was proposed by using NLP. The authors indicated that the ZOCR method was insufficient in extracting such information, and thus, they utilized the spaCy library which provides linguistically complex models for searching for necessary text information. Recent studies have shown that NLP is an important and effective approach in solving problems and offers new opportunities for improving information retrieval processes in text documents, resulting in more accurate and complete results.

After analyzing recent research on NLP [7, 8], it can be concluded that this approach is significant and effective in overcoming the limitations found in previously analyzed methods. The use of NLP can provide more accurate and complete results, simplify the recognition of complex structures, and improve the quality of textual information retrieval. For example, in the study [9] authors implemented an NLP algorithm to extract the presence of social factors from clinical text, which is considered as an unstructured document. The paper [10] also shows the usage of rule-based NLP search for unstructured data in electronic health records. The publication [11] outlines the NLP text extraction from unstructured geoscience reports.

3 MATERIALS AND METHODS

Generative AI refers to a class of AI systems that are specifically designed to generate new and original content, rather than just analyzing existing data or making predictions based on learned patterns. These systems employ various techniques, such as machine learning and deep learning, to produce fresh content that is often comparable, if not identical, to what a human can produce.

The development of generative AI has been significantly advanced by the contribution made by the OpenAI company [12]. Among their notable accomplishments is the ChatGPT model, which has been continuously evolving. OpenAI has made interaction with GPT models accessible through the public OpenAI API, which is widely used in various applications, including chatbots, content creation, and natural language understanding tasks. According to research [13], ChatGPT showcases significant progress in the field of natural language processing and has the potential to revolutionize the way we interact with machines and process natural language data. The model is also capable of maintaining the context of conversations, enabling it to refer to previous messages to provide relevant responses.

The potential of ChatGPT in the field of natural language processing (NLP) and the search approach outlined in a publication [6] have paved the way for leveraging modern generative artificial intelligence (AI) capabilities in the process of rule-based search for textual information. Today NLP mechanisms and capabilities can enable a more efficient and effective search of unstructured textual data.

When communicating with ChatGPT, prompts are used to provide information – input data given to the model for generating responses and can be messages, questions, or any text that provides context or instruction. Users interact with ChatGPT by sending prompts, and the model generates text responses based on the input. It is important to note that the quality and relevance of the responses generated by ChatGPT depend on the clarity and specificity of the prompts. A prompt is a crucial component in interacting with the context-forming model. In the paper [14], the main approaches to prompt engineering, which is the process of forming and correcting prompts, are considered and highlighted. Therefore, by using the correct approaches to form accurate AI queries, it is possible to search for information more effectively. The clearer and more specific the prompt is, the better the search results will be.

Using GPT models can be effective for searching and extracting text from text documents, including unstruc-

© Boiko V. O., 2024
DOI 10.15588/1607-3274-2024-2-12

tured ones. Prompts are used to retrieve specific textual information. Correction of output results can be done by providing sample results. To distinguish the extracted information, prompts with specific example results can be marked by a unique identifier or name. All these parts can be represented as variables, that are denoted as rules in formula (1) and are a part of model input data *X*. Since prompts are often used in an imperative form, the variables can be called imperative. Fig. 1. illustrates the general structure of an imperative variable.

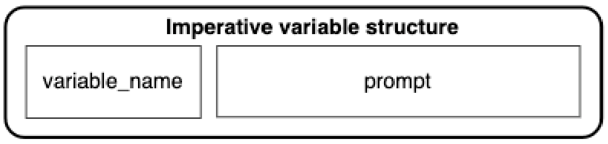


Figure 1 – Imperative variable structure

These variables contain prompts that can represent different rules for filtering, searching data, and even descriptions of how to format the output result which are saved in result variables. To make a model better understand what kind of data it should extract from documents, the sample response for each imperative variable should be defined. This type of data can be called a sample variable as it represents an example of an output for the appropriate imperative variable. The structure of a sample variable is shown in Fig. 2.

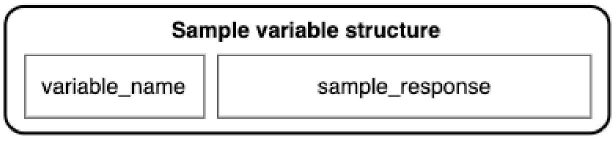
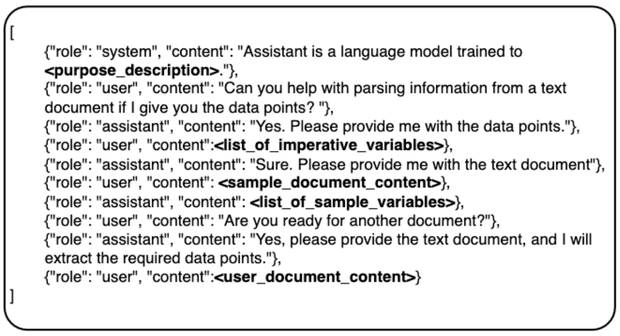


Figure 2 – Sample variable structure

After defining imperative and sample variables, there is a need to set up the chat model, which is described in this instruction [15]. Additionally, a chat assistant should have contextual information that briefly describes an area in which a user works. Imperative variables are included in the user-side messages along with the sample responses for assistance. For more accurate results the sample document can be provided. This document contains an example content to show the assistant what kind of documents it will deal with in case of user-provided documents. These settings are demonstrated in Fig 3.



```
[
    {"role": "system", "content": "Assistant is a language model trained to
    <purpose_description>."},
    {"role": "user", "content": "Can you help with parsing information from a text
    document if I give you the data points? "},
    {"role": "assistant", "content": "Yes. Please provide me with the data points."},
    {"role": "user", "content":<list_of_imperative_variables>},
    {"role": "assistant", "content": "Sure. Please provide me with the text document"},
    {"role": "user", "content": <sample_document_content>},
    {"role": "assistant", "content": <list_of_sample_variables>},
    {"role": "user", "content": "Are you ready for another document?"},
    {"role": "assistant", "content": "Yes, please provide the text document, and I will
    extract the required data points."},
    {"role": "user", "content":<user_document_content>}
]
```

Figure 3 – Messages to set up the assistant

In GPT models during the training stage, several common operations are typically performed, including the specification of a loss function and its minimization through optimization algorithms. In the case of an unstructured document text search task, the loss function can be defined to measure the discrepancy between the model's predictions – the search results and the ground truth relevant information that the model is expected to retrieve from the documents. In the case of text generation tasks the Cross-Entropy Loss function can be used. It is represented by the following formula (2):

$$L(Y,Y') = -\sum_{i=1}^{N} \ln(P(y'_i \mid X, Y'_{i:i-1})) . \qquad (2)$$

It's important to note that function (2) works only with a single sequence of data, but the more effective way will be enhancing the existing function with the possibility of batch processing, so multiple sequences can be used simultaneously. In the context of batch processing, where multiple sequences are processed simultaneously, the loss function computes the discrepancy between the model's predictions and the ground truth for all sequences in the dataset. To obtain a representative measure of the average discrepancy per sequence in the batch, $1/M$ a coefficient is included. This factor ensures that the loss value is normalized by the number of sequences. Updated formula looks like this (3):

$$L(Y,Y') = -\frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{N} \ln(P(y'_{i,j} \mid X, Y'_{i:i-1})) . \qquad (3)$$

In (2) the double summation is used since there is a need for batch processing to sum over all tokens in the output sequence Y' and all possible tokens in the vocabulary (the inner summation) to compute the overall loss. This ensures that the discrepancy is considered for each token in the predicted sequence compared to all possible tokens in the vocabulary [16].

After the loss function is defined there is a need to optimize it, because the primary goal during training is to make the model learn to perform more accurate predictions or generate more relevant outputs. Several optimization techniques can be used, but the most effective is the adaptive moment estimation algorithm (Adam) and its variations that are used in GPT models. Adam optimization is performed in several steps [17]. Firstly, the gradient function should be defined. For the current case, it can be represented by the following formula (4):

$$\nabla L(\theta) = -\frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{N} \frac{\partial}{\partial\theta} \ln(P(y_{i,j} \mid X, y_{i:i-1})) . \qquad (4)$$

It computes the partial derivative of the logarithm of the predicted probability of each token in the output sequence given the input and previous tokens, summed over all training examples (documents) and tokens within each

DOI 10.15588/1607-3274-2024-2-12

document. The gradient function with respect to the time step is represented by the following formula (5):

$$g^{(t)} = \nabla L(\theta^{(t-1)}) . \tag{5}$$

Then, update the first (6) and second (7) moment estimation happens:

$$m^{(t)} = \beta_1 m^{(t-1)} + (1-\beta_1) g^{(t)} , \tag{6}$$

$$v^{(t)} = \beta_2 v^{(t-1)} + (1-\beta_2)(g^{(t)} \circ g^{(t)}) . \tag{7}$$

After that, a bias correction for each updated moment estimate is computed (8, 9):

$$\hat{m}^{(t)} = \frac{m^{(t)}}{1-\beta_1{}^t} , \tag{8}$$

$$\hat{v}^{(t)} = \frac{v^{(t)}}{1-\beta_2{}^t} . \tag{9}$$

The final stage is to update the model parameters based on calculated adaptive learning rate and bias-corrected moment estimates (10):

$$\theta^{(t)} = \theta^{(t-1)} - \alpha^{(t)} \frac{\hat{m}^{(t)}}{\sqrt{\hat{v}^{(t)}} + \varepsilon} . \tag{10}$$

The algorithm which includes operations (4–10) is performed several times. The loop works until either the model converges or the maximum number of iterations is reached, whichever comes first.

The mentioned algorithms used in the model training process today are incorporated into GPT language models, like GPT-3.5 Turbo from OpenAI. However, this model does not disclose its optimization techniques, but according to several types of research, the methods shown in this paper are used by some GPT models. Instead of building a custom model that can last long and take a big amount of computing resources to maintain training and deployment processes, the most stable GPT-3.5 Turbo model can be used effectively for tasks such as a rule-based search. Thus, the final formula for getting the search result Y' using the GPT-3.5 Turbo model can be represented as the following (11):

$$Y' = GPT(X) . \tag{11}$$

The imperative variables method can be applied to different documents and to make it generic the appropriate search model has been developed. According to this model, imperative variables along with sample responses are saved in data storage. This model allows us to build custom and flexible templates based on imperative variables for different areas and apply them for rule-based textual search in documents, including unstructured ones. Templates consist of imperative and sample variables with one sample document. Fig 4 illustrates this model which is represented by a sequence diagram. This is a generic approach to serve the NLP-based textual search. Users can define their templates according to their business area, specify the purpose description, and manage templates that contain imperative variables. Additionally, the model can be used to implement the appropriate API.

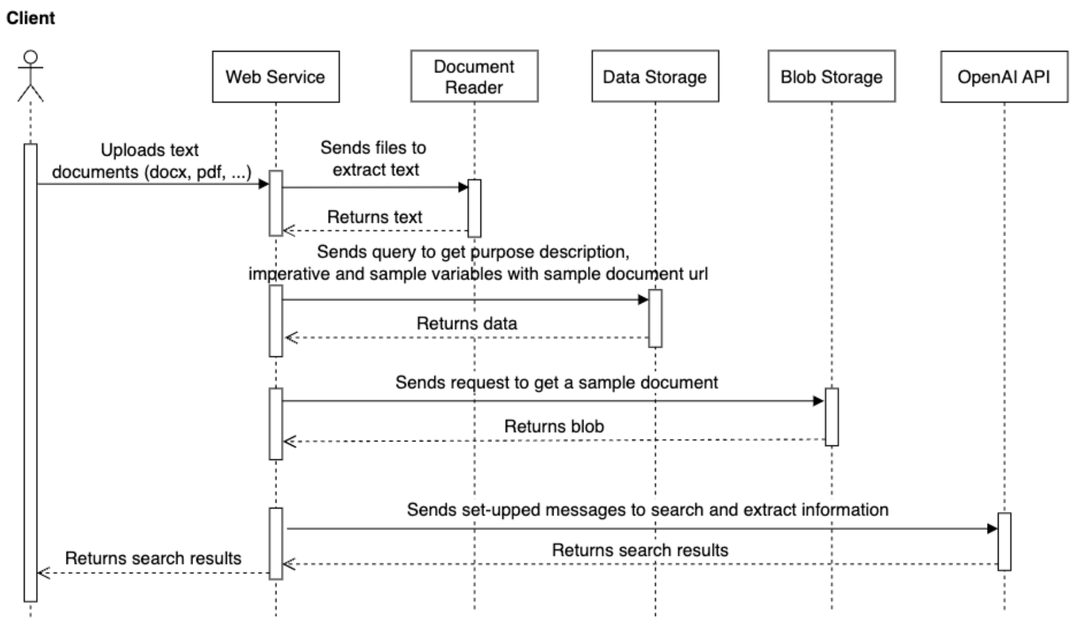

Client
Web Service
Document Reader
Data Storage
Blob Storage
OpenAI API
Uploads text documents (docx, pdf, ...)
Sends files to extract text
Returns text
Sends query to get purpose description, imperative and sample variables with sample document url
Returns data
Sends request to get a sample document
Returns blob
Sends set-upped messages to search and extract information
Returns search results
Returns search results

Figure 4 – Sequence diagram of a search model using the imperative variables method

DOI 10.15588/1607-3274-2024-2-12

Compared to the pattern-matching search, the imperative variables search model has several advantages:

1) The model requires only properly described prompts using natural language and sample responses while pattern-matching uses Regular Expression syntax to seek data that is not well-known to common users.

2) Along with a description of what the user needs to find, it can be formally extended by the flexible set of additional requirements, for example, of how it should be formatted in the output or conditions of what to do if the required data is not found without any additional programmed logic.

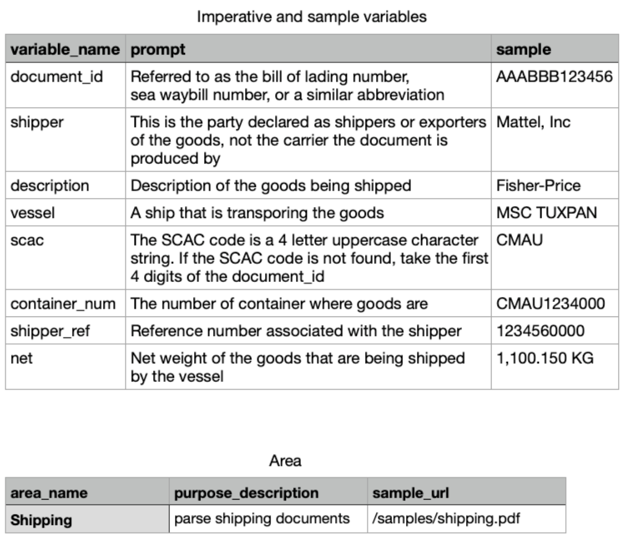
However, with a large number of imperative variables or large documents, a search can be slower, thus the model needs to be optimized using multithreading or parallelization mechanisms which is a part of further studies.

4 EXPERIMENTS

The model has been implemented to show how the method works. Since it is generic, any area is suitable for this model. So, the Shipping area was selected to conduct the experiments.

According to the model – several imperative and response variables were created in the MySQL database.
Also, to properly set up the Open AI assistant, the purpose description was initialized.

For a convenient view data was converted to CSV format. The area information along with imperative and sample variables are listed in Figure 5.



Imperative and sample variables

variable_name	prompt	sample
document_id	Referred to as the bill of lading number, sea waybill number, or a similar abbreviation	AAABBB123456
shipper	This is the party declared as shippers or exporters of the goods, not the carrier the document is produced by	Mattel, Inc
description	Description of the goods being shipped	Fisher-Price
vessel	A ship that is transporing the goods	MSC TUXPAN
scac	The SCAC code is a 4 letter uppercase character string. If the SCAC code is not found, take the first 4 digits of the document_id	CMAU
container_num	The number of container where goods are	CMAU1234000
shipper_ref	Reference number associated with the shipper	1234560000
net	Net weight of the goods that are being shipped by the vessel	1,100.150 KG

Area

area_name	purpose_description	sample_url
Shipping	parse shipping documents	/samples/shipping.pdf

Figure 5 – Variables for text extraction

For better results, there was created a sample shipping document which contains example data for the GPT model. It is saved on Azure blob storage.

To conduct the experiments .NET environment was used and for simplicity there was developed a console application that accepts text documents and on output generates the CSV file with a response.

There are several steps were performed to parse the documents:

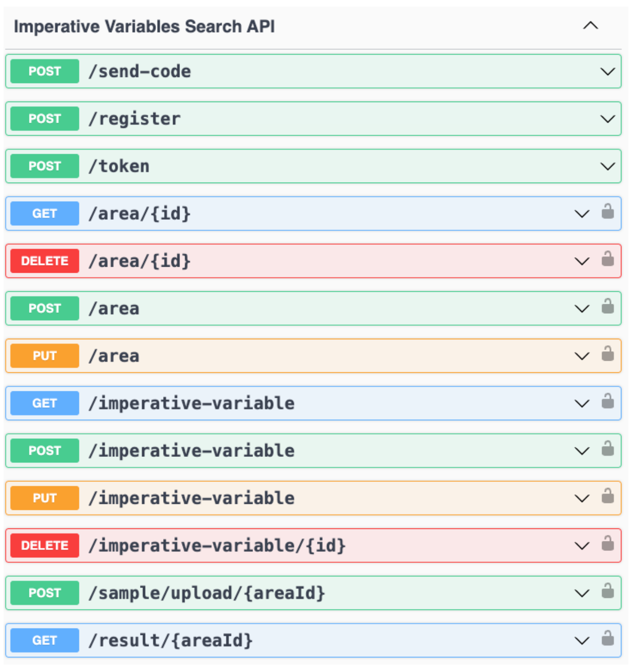
1) Application field information and variables are retrieved from the MySQL database.

2) A sample document is downloaded from the Azure Blob storage.

3) The messages are set up and sent to the OpenAI API for processing.

4) The result is obtained in a JSON format and saved to a CSV file.

The search model was implemented in ASP.NET-based Imperative Variable Search Web API which is a general point to access the rule-based search. The list of implemented endpoints is illustrated in Fig. 6.



Imperative Variables Search API

POST	/send-code
POST	/register
POST	/token
GET	/area/{id}
DELETE	/area/{id}
POST	/area
PUT	/area
GET	/imperative-variable
POST	/imperative-variable
PUT	/imperative-variable
DELETE	/imperative-variable/{id}
POST	/sample/upload/{areaId}
GET	/result/{areaId}

Figure 6 – Imperative Variable Search API endpoints

Endpoints accept user requests and perform the main functionality of the API. A user first needs to register an account, then create an area with an appropriate name and purpose, populate the set of imperative variables along with sample responses that are related to the area, upload a sample file, and then upload files to perform the rule-based search and obtain the results. The API has a convenient way of interacting using Swagger and can be integrated into different business solutions.

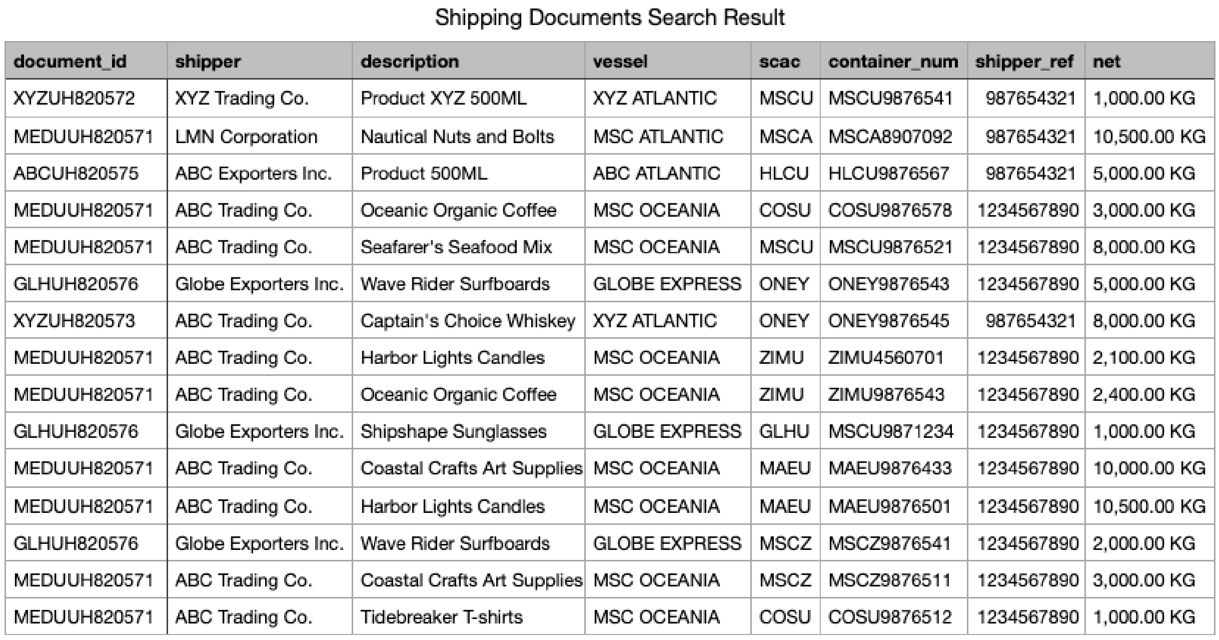
5 RESULTS

15 unstructured shipping documents were selected and processed (18 KB each) with 8 variables. No parsing errors occurred. There are conducted 5 attempts of execution to determine the average time. Time calculation includes retrieving imperative variables and other information from the database, extracting textual content from documents, and batch-sending requests to the OpenAI API. The result of processing documents was saved into a CSV file and illustrated in Fig. 7.

According to time measurement results the average time spent on requests to OpenAI API and the time of overall program execution is approximately 3 seconds which can be considered as an acceptable result (Fig. 8).

DOI 10.15588/1607-3274-2024-2-12

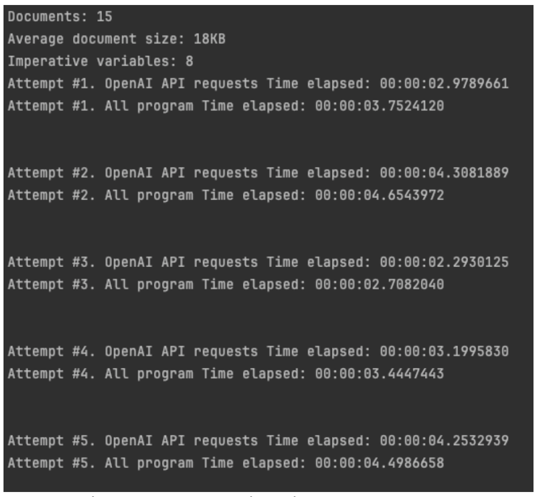
However, OpenAI API has a rate limit that depends on the Tier subscription [18]. The experiment was conducted on Tier 1 which has a limit of 60000 tokens that can be sent per 1 minute, which also keeps the limit of the number of documents that can be processed per this time. Thus, the queue should be used in case of a large number of documents. Additionally, the content length has limits too – for GPT-3.5 Turbo model is 16385 tokens [19], thus larger documents should be split into smaller parts before processing.



Shipping Documents Search Result

document_id	shipper	description	vessel	scac	container_num	shipper_ref	net
XYZUH820572	XYZ Trading Co.	Product XYZ 500ML	XYZ ATLANTIC	MSCU	MSCU9876541	987654321	1,000.00 KG
MEDUUH820571	LMN Corporation	Nautical Nuts and Bolts	MSC ATLANTIC	MSCA	MSCA8907092	987654321	10,500.00 KG
ABCUH820575	ABC Exporters Inc.	Product 500ML	ABC ATLANTIC	HLCU	HLCU9876567	987654321	5,000.00 KG
MEDUUH820571	ABC Trading Co.	Oceanic Organic Coffee	MSC OCEANIA	COSU	COSU9876578	1234567890	3,000.00 KG
MEDUUH820571	ABC Trading Co.	Seafarer's Seafood Mix	MSC OCEANIA	MSCU	MSCU9876521	1234567890	8,000.00 KG
GLHUH820576	Globe Exporters Inc.	Wave Rider Surfboards	GLOBE EXPRESS	ONEY	ONEY9876543	1234567890	5,000.00 KG
XYZUH820573	ABC Trading Co.	Captain's Choice Whiskey	XYZ ATLANTIC	ONEY	ONEY9876545	987654321	8,000.00 KG
MEDUUH820571	ABC Trading Co.	Harbor Lights Candles	MSC OCEANIA	ZIMU	ZIMU4560701	1234567890	2,100.00 KG
MEDUUH820571	ABC Trading Co.	Oceanic Organic Coffee	MSC OCEANIA	ZIMU	ZIMU9876543	1234567890	2,400.00 KG
GLHUH820576	Globe Exporters Inc.	Shipshape Sunglasses	GLOBE EXPRESS	GLHU	MSCU9871234	1234567890	1,000.00 KG
MEDUUH820571	ABC Trading Co.	Coastal Crafts Art Supplies	MSC OCEANIA	MAEU	MAEU9876433	1234567890	10,000.00 KG
MEDUUH820571	ABC Trading Co.	Harbor Lights Candles	MSC OCEANIA	MAEU	MAEU9876501	1234567890	10,500.00 KG
GLHUH820576	Globe Exporters Inc.	Wave Rider Surfboards	GLOBE EXPRESS	MSCZ	MSCZ9876541	1234567890	2,000.00 KG
MEDUUH820571	ABC Trading Co.	Coastal Crafts Art Supplies	MSC OCEANIA	MSCZ	MSCZ9876511	1234567890	3,000.00 KG
MEDUUH820571	ABC Trading Co.	Tidebreaker T-shirts	MSC OCEANIA	COSU	COSU9876512	1234567890	1,000.00 KG

Figure 7 – Search results for 15 shipping documents



```
Documents: 15
Average document size: 18KB
Imperative variables: 8
Attempt #1. OpenAI API requests Time elapsed: 00:00:02.9789661
Attempt #1. All program Time elapsed: 00:00:03.7524120


Attempt #2. OpenAI API requests Time elapsed: 00:00:04.3081889
Attempt #2. All program Time elapsed: 00:00:04.6543972


Attempt #3. OpenAI API requests Time elapsed: 00:00:02.2930125
Attempt #3. All program Time elapsed: 00:00:02.7082040


Attempt #4. OpenAI API requests Time elapsed: 00:00:03.1995830
Attempt #4. All program Time elapsed: 00:00:03.4447443


Attempt #5. OpenAI API requests Time elapsed: 00:00:04.2532939
Attempt #5. All program Time elapsed: 00:00:04.4986658
```

Figure 8 – Execution time measurement

6 DISCUSSION

The method of imperative variables generally shows applicable results. Compared to research that was performed and resulted in [7–11] the developed method has an easier implementation, so the API that can be built on this model, will be developed without any extra spending time for developing custom NLP models, as it was done and outlined in those studies. Therefore, custom-trained NLP models are often trained on a set of data that is related to a specific area. This approach has advantages in that this model better interacts and produces more accurate results since it is trained according to the specific field. However, the disadvantage could be the absence of enough flexibility, so these models will need to be additionally trained in case when they are used in another field.

The significant advantage of the method is that it uses a modern and well-trained GPT model and an open-source API for processing queries. According to experiment results this method is relatively fast and produces accurate search results. Also, this method is flexible and has relatively small-time expenses.

However, since it is based on OpenAI GPT models which have access to facilities based on a chosen subscription, API has several limitations that lead to the inability to parse large sets of data. Due to them, there is a need to perform several optimizations. For example, if the model deals with a large document, it can be divided into smaller parts which are acceptable by OpenAI API.

This method can be extended by combining other methods that are used to narrow the search range, like ZOCR. With these optimizations, OpenAI API will not be overloaded, but it will not be suitable for all unstructured documents since specific cases can take place when the document contains the desired textual information in different places. Overall, the combination of advanced language models with OCR technologies represents a promising direction for improving document processing and information retrieval tasks. By leveraging the strengths of both approaches, it's possible to create more robust and versatile solutions capable of handling a wide range of document formats and sources.

Despite these limitations, the method remains a powerful tool for natural language processing tasks, offering a balance between performance and accessibility. Continued improvements in the underlying GPT models and enhancements in the API capabilities may further mitigate these limitations in the future.

The method's integration with existing systems and workflows is relatively straightforward, because of its API-oriented model. This allows developers to seamlessly incorporate its capabilities into their applications without significant overhead.

CONCLUSIONS

The generic rule-based unstructured data search method was developed and incorporated into an API-oriented model.

The scientific novelty of the obtained results is that the generic imperative variables method based on OpenAI GPT models is firstly proposed. It outlines the approach of a rule-based search in unstructured documents based on GPT models. The flexible field-independent search model is designed based on the method. This allows to automate finding the information in documents with no predefined structure, build requests, and criteria in different forms for search and form the desired output results.

The practical significance of the obtained results is that the method is able to automate information retrieval tasks, without the need for predefined structures or explicit rules, making it highly adaptable to diverse use cases. The developed flexible model enables organizations to streamline their document processing workflows, improve efficiency, and extract valuable insights from unstructured textual data. Implemented Web API allows users to build custom templates to perform a rule-based search.

Prospects for further research are to study the optimization approaches to minimize the impact of OpenAI API limitations and decrease the execution time. Additionally, further investigation could involve expanding the usage of imperative variables in programming and software development.

REFERENCES

1. Dutta H., Gupta A. PNRank: Unsupervised ranking of person name entities from noisy OCR text, *Decision support systems*, 2021, P. 113662.
2. Kumar V., Chinmay B., Varsha N. A framework for document plagiarism detection using Rabin Karp method, *International Journal of Innovative Research in Technology and Managemen,* 2021, Vol. 5, pp. 18–19.
3. Onyenwe I. et al. Developing Smart Web-Search using Regex, *International Journal on Natural Language Computing,* 2022,Vol. 11, No. 3, pp. 25–30.
4. OCR – optical character recognition – azure AI services [Electronic resource], *Microsoft Learn: Build skills that open doors in your career.* Mode of access: https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/overview-ocr (date of access: 23.03.2024). Title from screen.
5. Drobac S., Lindén K. Optical character recognition with neural networks and post-correction with finite state methods, *International journal on document analysis and recognition (IJDAR),* 2020, Vol. 23, No. 4, pp. 279–295.
6. Deshmukh M., Maheshwari S. Free form document based extraction using ML, *International journal of science and research (IJSR)*, 2019, Vol. 8, P. 1.
7. Kwabena A. E. et al. An automated method for developing search strategies for systematic review using natural language processing (NLP), *MethodsX,* 2022, P. 101935.
8. Just J. Natural language processing for innovation search – Reviewing an emerging non-human innovation intermediary, *Technovation*, 2024, Vol. 129, P. 102883.
9. Allen K. S. et al. Natural language processing-driven state machines to extract social factors from unstructured clinical documentation, *JAMIA open*, 2023, Vol. 6, No. 2.
10. Li I. et al. Neural natural language processing for unstructured data in electronic health records: A review, *Computer science review*, 2022, Vol. 46, P. 100511.
11. Qiu Q. et al. Automatic spatiotemporal and semantic information extraction from unstructured geoscience reports using text mining techniques, *Earth science informatics,* 2020, Vol. 13, No. 4, pp. 1393–1410.
12. Research [Electronic resource], *OpenAI.* Mode of access: https://openai.com/research/overview (date of access: 24.03.2024). Title from screen.
13. Koubaa A. et al. Exploring ChatGPT capabilities and limitations: A critical review of the NLP game changer. Riyadh. Preprints, 2023, 29 p. (Preprint / Prince Sultan University; 2023030438).
14. Ekin S. Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices. Texas City: TechRxiv, 2023, 12 p. (Preprint / Texas A&M University; 22683919).
15. Chat completions API [Electronic resource]. Mode of access: https://platform.openai.com/docs/guides/text-generation/chat-completions-api (date of access: 26.03.2024). – Title from screen.
16. Lee M. A mathematical investigation of hallucination and creativity in GPT models, *Mathematics,* 2023, Vol. 11, No. 10, P. 2320.
17. Kingma D. P., Ba J. Adam: A Method for Stochastic Optimization, *3rd International Conference for Learning Representations*, San Diego, 7–9 May 2015.
18. Usage tiers [Electronic resource]. Mode of access: https://platform.openai.com/docs/guides/rate-limits/usage-tiers?context=tier-one (date of access: 26.03.2024). Title from screen.
19. GPT-3.5 Turbo [Electronic resource]. Mode of access: https://platform.openai.com/docs/models/gpt-3-5-turbo (date of access: 26.03.2024). – Title from screen.

Received 04.03.2024.
Accepted 24.04.2024.

УДК 004.93, 004.8

МЕТОД ІМПЕРАТИВНИХ ЗМІННИХ ДЛЯ АВТОМАТИЗАЦІЇ ПОШУКУ ТЕКСТОВОЇ ІНФОРМАЦІЇ У НЕСТРУКТУРОВАНИХ ДОКУМЕНТАХ

Бойко В. О. – асистент кафедри інженерії програмного забезпечення Хмельницького національного університету, Хмельницький, Україна.

DOI 10.15588/1607-3274-2024-2-12

АНОТАЦІЯ

Актуальність. На сьогодні існує багато підходів для виконання ефективного текстового пошуку. Для отримання знаходження та вилучення фрагментів інформації з документів широко використовуються такі методи, як зіставлення з шаблоном і оптичне розпізнавання символів. Однак вони працюють із чітко визначеною структурою документа, тоді як неструктуровані документи не можуть бути оброблені такими методами. А тому проблема полягає в автоматизації текстового пошуку в документах з неструктурованим вмістом. Метою дослідження було розробити метод та реалізувати ефективну модель пошуку вмісту в неструктурованій текстовій інформації.

Мета роботи – реалізація методу та моделі текстового пошуку на основі правил для отримання інформації з документів з неструктурованим текстовим вмістом.

Метод. Для досягнення мети дослідження розроблено та застосовано у відповідній моделі метод критеріального текстового пошуку для знаходження інформації у різнорідному текстовому вмісті. Він заснований на обробці природної мови, яка була вдосконалена в останні роки разом із новим генеративним штучним інтелектом, який стає все більш доступним та продуктивним.

Результати. Метод реалізовано в розробленій моделі, яка представляє шаблон або структуру неструктурованого текстового пошуку для розробників програмного забезпечення. Розроблено прикладний програмний інтерфейс для взаємодії з моделлю.

Висновки. Проведені експерименти у вигляді реалізованого програмного забезпечення підтвердили працездатність запропонованого методу та доводять практичність його використання для вирішення задач текстового пошуку в неструктурованих документах. Перспективи подальших досліджень можуть включати покращення продуктивності за допомогою багатопотоковості або паралелізації для великих текстових документів, а також розробка підходів до оптимізації методу для мінімізації впливу обмежень обробки контенту прикладного програмного інтерфейсу OpenAI. Крім того, додаткові дослідження можуть включати розширення області використання імперативних змінних у програмуванні та розробці програмного забезпечення.

КЛЮЧОВІ СЛОВА: текстовий пошук, неструктуровані текстові документи, обробка природної мови, пошук на основі правил, генеративний штучний інтелект, імперативні змінні.

ЛІТЕРАТУРА

1. Dutta H. PNRank: Unsupervised ranking of person name entities from noisy OCR text / Haimonti Dutta, Aayushee Gupta // Decision support systems. – 2021. – P. 113662.
2. Kumar V. A framework for document plagiarism detection using Rabin Karp method / Vivek Kumar, Bhatt Chinmay, Namdeo Varsha // International Journal of Innovative Research in Technology and Managemen. – 2021. – Vol. 5. – P. 18–19.
3. Developing Smart Web-Search using Regex / Ikechukwu Onyenwe et al. // International Journal on Natural Language Computing. – 2022. – Vol. 11, No. 3. – P. 25–30.
4. OCR – optical character recognition – azure AI services [Electronic resource] // Microsoft Learn: Build skills that open doors in your career. – Mode of access: https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/overview-ocr (date of access: 23.03.2024). – Title from screen.
5. Drobac S. Optical character recognition with neural networks and post-correction with finite state methods / Senka Drobac, Krister Lindén // International journal on document analysis and recognition (IJDAR). – 2020. – Vol. 23, No. 4. – P. 279–295.
6. Deshmukh M. Free form document based extraction using ML / Mona Deshmukh, Shruti Maheshwari // International journal of science and research (IJSR). – 2019. – Vol. 8. – P. 1.
7. An automated method for developing search strategies for systematic review using natural language processing (NLP) / Antwi Effah Kwabena [et al.] // MethodsX. – 2022. – P. 101935.
8. Just J. Natural language processing for innovation search – Reviewing an emerging non-human innovation intermediary / Julian Just // Technovation. – 2024. – Vol. 129. – P. 102883.
9. Natural language processing-driven state machines to extract social factors from unstructured clinical documentation / Katie S. Allen et al. // JAMIA open. – 2023. – Vol. 6, No. 2.
10. Neural natural language processing for unstructured data in electronic health records: A review / Irene Li et al. // Computer science review. – 2022. – Vol. 46. – P. 100511.
11. Automatic spatiotemporal and semantic information extraction from unstructured geoscience reports using text mining techniques / Qinjun Qiu et al. // Earth science informatics. – 2020. – Vol. 13, No. 4. – P. 1393–1410.
12. Research [Electronic resource] // OpenAI. – Mode of access: https://openai.com/research/overview (date of access: 24.03.2024). – Title from screen.
13. Exploring ChatGPT capabilities and limitations: A critical review of the NLP game changer / Anis Koubaa et al. – Riyadh : Preprints, 2023. – 29 p. – (Preprint / Prince Sultan University; 2023030438).
14. Ekin S. Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices / Sabit Ekin. – Texas City: TechRxiv, 2023. – 12 p. – (Preprint / Texas A&M University; 22683919).
15. Chat completions API [Electronic resource]. – Mode of access: https://platform.openai.com/docs/guides/text-generation/chat-completions-api (date of access: 26.03.2024). – Title from screen.
16. Lee M. A mathematical investigation of hallucination and creativity in GPT models / Minhyeok Lee // Mathematics. – 2023. – Vol. 11, no. 10. – P. 2320.
17. Kingma D. P. Adam: A Method for Stochastic Optimization / Diederik P. Kingma, Jimmy Ba // 3rd International Conference for Learning Representations: International Conference, San Diego, 7–9 May 2015.
18. Usage tiers [Electronic resource]. – Mode of access: https://platform.openai.com/docs/guides/rate-limits/usage-tiers?context=tier-one (date of access: 26.03.2024). – Title from screen.
19. GPT-3.5 Turbo [Electronic resource]. – Mode of access: https://platform.openai.com/docs/models/gpt-3-5-turbo (date of access: 26.03.2024). – Title from screen.

DOI 10.15588/1607-3274-2024-2-12