ANALYTICAL PLATFORM

Jan Přichystal[1], Roman Valovič[1]

[1]Department of Informatics, Faculty of Business and Economics, Mendel University in Brno, Zemědělská 1, 613 00 Brno, Czech Republic

ABSTRACT

The Analytical Platform is an online platform that simplifies work of analytical department, front-office, back-office and compliance for investment firms. It is using an artificial intelligence, data analytics and machine learning in finance and investments. Analytical Platform helps investment funds, securities traders and professional investors gain higher alpha, and lower beta and get flawless records of investment decisions. It offers various specialized tools and services in a flexible and personalized way to fit investor's needs. This article presents the components of the platform, the technologies used for the development and approaches for sentiment analysis and summarization of financial articles as a source of key information for decision support.

Keywords: data mining, financial markets, natural language processing, machine learning, sentiment analysis, text summarization, web application

JEL Code: L86

1 INTRODUCTION

Analytical Platform is a multifunctional platform combining sets of functions for performing data analysis and text analysis. The goal of the platform is to remove information asymmetry and bring added value to users in the form of reducing transaction costs, streamlining work processes, and uncovering new business opportunities using unique software combining data and text analytics capabilities.

The logic of the platform is built on a multi-layered architecture. The first layer is the development environment (DPS Lab), which allows you to combine tools in the field of data and text analytics and build your own scenarios to obtain the desired outputs. Verified combinations with market potential for the selected domain are then displayed within the end applications (DPS Applications). A communication interface (DPS Result sharing) is also available.

The project aimed to create a platform that can be used through microservices as part of larger units and information flows of the enterprise architecture of communication and computing systems. The goal was achieved through the creation of a modular system that:

1. provides functionality for data import and basic data/text analytics (DPS Lab),
2. allows you to connect and combine individual tools to achieve the desired result (DPS Lab),
3. provides end applications for use in the selected field (DPS Applications),

https://doi.org/10.11118/978-80-7509-990-7-0127

4. enables sharing and discussion of results within working analytical teams (DPS Result sharing) and at the same time enables differentiated access to applications (API, web application, analyses).

Users can work with the platform and independently define the functions they will use to process the input data. Alternatively, they will use an industry package prepared for their work role within a specific market segment and thus use the industry experience implemented directly in the platform. The aim of this article is to introduce the Analytical platform as a new decision support tool in the field of securities investment, which uses artificial intelligence and machine learning tools to identify key parameters when building individual investors' portfolios.

2 SYSTEM COMPONENTS

The platform consists of several relatively separate components that the user can use independently or share the results from these individual components and use them for complex analyses.

2.1 DPS Lab

The data platform (DPS Lab) is composed of several basic technological objects that enable the creation of advanced data processing scenarios. The input object is a set of functions for importing data of various formats. Data input accepts machine-readable files such as XML, CSV, JSON and other data formats (for example, links to YouTube videos from which information is extracted). The range of data used varies from open data of the public sector (US Statements/EDI), through data downloaded by web crawlers (for example from the information websites Yahoo Finance or Seeking Alpha), through paid APIs (for example RapidAPI), to private data that users they upload to the platform themselves.

The platform enables the input of diverse data from different fields, it is possible to apply functions and methods specifically developed for a specific field cross-field. For example, the sentiment analysis of financial market articles. This makes it possible to achieve innovative, experimental data results that can be repeatedly compared with each other. This methodical procedure of systematic work with data finds application in data-intensive fields in which understanding data is a key competence for gaining or maintaining a competitive advantage.

The application allows, for example, to analyze videos devoted to product reviews. In the domain case of investment research, the analyst will use them for decision support, when he expects that the high-quality products of the company being evaluated will likely contribute to the growth of the stock price. In the same way, the information obtained helps buyers in the supply chain for stockpiling and other strategic decisions. In the field of construction and project preparation, testing is taking place, for example, with crowdfunding platforms for scoring developers, i.e. companies coming in with a need to finance their projects. Separate article (Přichystal et al., 2023) deals with this module in more detail.

2.2 Portfolio Manager

This component enables the management of stock portfolios according to the expected return, volatility or Sharpe ratio, according to the procedures of Harry Markowitz, who won the Nobel Prize for this theory.

The application combines a theory that investment company analysts have not yet been able to apply to robust portfolios, at the same time they did not have sufficiently high-quality pre-processed input data for it, which are built on DPS Lab (obtaining price data from the

Rapid API or scraping from Yahoo Finance, preprocessing data and subsequent determination restrictions for choosing the length of the time series) and subsequently enriched in the application with specific functions that interest end users. This is, for example, the possibility of exchanging a specific number of shares, or the choice of a specific stock pool, so that both effective results are achieved according to theory and at the same time the principles of the specific analyst who uses the tool are observed.

2.3 Stock Picking Lab

The application provides the management of stock portfolios according to multi-factor analysis in combination with the use of machine learning for the selection and operation of models. Multifactor analysis means that individual indicators that are important for analysts are calculated based on the input data. SP Lab builds on the activities and knowledge acquired before the start of the project implementation, when the consortium already had a functional model in the Matlab program generating predictions on a monthly basis and a linear model for evaluating shares, including the basic decomposition of scores, these outputs were displayed in a simple web application.

As part of the follow-up activities of the project, knowledge was significantly expanded and the DPS Lab and StockPicking Lab applications now provide users with the possibility of training their own model for valuing stocks, a view of the decomposition of individual indicators (displaying the results of individual indicators + weights that the model selected and trained for them in a given prediction assigned), which provides analysts with new opportunities to understand data and make better investment decisions.

2.4 Summary of Financial Articles

Summary of Financial Articles is a system designed to speed up the process of obtaining and processing information for these experts and thus facilitate their work. This system obtains articles, news or tweets from selected sources, especially in the field of financial markets, and analyzes their content. The aim is to offer a summarized view of individual documents in the form of several sentences summarizing the main ideas, to determine their sentiment and to provide aggregated data for specified time periods. A more detailed description of this module is available in Jakúbek et al. (2023).

2.5 DPS Result sharing

The output of the project is implemented in the form of a web application, which allows the use of the platform without the need for installation and complicated configuration via a web browser. The advantage of this solution is independence from the used technical equipment (personal computer, tablet, mobile phone) and operating system (MS Windows, MacOS, GNU Linux, ...).

The advantage of the web application is also the easy possibility of teamwork of several users on joint projects, sharing of solutions in progress and team communication provided by the DPS Result sharing application, within which users can share results and work in progress.

3 SYSTEM DESIGN

3.1 Frontend implementation

The frontend and backend of the system are two standalone applications that communicate via thin REST API on the web server. Heavy frontend is delivered to client browser just once and the content of the website is then partially updated, in places that require change, directly in the browser. This concept is called Single Page Application (SPA) and can be implemented in various JavaScript libraries or frameworks. In our case, to deal with view layer in MVC software architecture, we chose ReactJS library.

3.1.1 Used technologies

ReactJS is still a popular open-source library that is used for building web (and mobile) composable user interfaces. In ReactJS, the user interface is broken down in a large number of reusable components. These components manage their states to handle both the logic and the view itself. Their hierarchical structure clearly defines unidirectional flow of data in the application and their APIs and lifecycle hooks describe the behavior.

The state of the component is changed after data mutation or by executing a user action and this update triggers a smart and controlled rendering process. In the background, ReactJS engine tries to improve performance by minimizing expensive manipulation with real DOM as much as possible, so it first renders the changes to the virtual DOM represented in JavaScript. Then its job is to reconcile the recent changes against the old one and only the necessary ones applied to the real DOM.

To get the best of ReactJS world, we used JSX (syntax extensions to JavaScript), TypeScript, functional components and React hooks. To manage application-level state we injected Redux into application. Redux is a predictable state container which plays a helping hand while developing large scale JavaScript application. To keep a similar look and feel to our previous apps, we selected Ant Design as the UI framework. This framework is known mostly in the East and does not follow ubiquitous and shabby material design.

3.2 System backend

The technological complexity and computing power requirements vary dramatically between applications; for example, the SFA Rest API, which responds to a client request with a preprocessed result stored in a database, has only minimal performance requirements on the machine on which the application runs. SFA Sentiment Analysis, on the other hand, provides outputs based on the computations of a complex BERT neural network.

However, the workload of each application varies throughout the day. While the Rest API is queried virtually constantly during normal business hours (8AM–4PM), Sentiment Analysis classifies a new batch of articles every hour. Not only from the point of view of application sustainability and robustness, but also from the point of view of cost-effectiveness, we did not implement the SFA system as a single monolith but opted for a “micro services” architecture. Although there is no strict definition of the term micro service, but we can talk about the framework that this architecture should fulfill:

- services communicate via HTTP (Fowler, 2015),
- services are independent of each other (Nadareishvili, 2016),
- services can be implemented using different programming languages, databases, environments and technologies, depending on the task they are supposed to solve (Chen, 2018),
- services are small, communicate only in a bounded context, autonomous, built and deployed by an automated process (Nadareishvili, 2016).

We implemented the fulfillment of these requirements using Docker technology. In Docker jargon, each service represents a so-called container. The container is a small simple standalone software package and as a monolith contains everything needed to run the application – runtime environment code, system libraries and configuration. Containers isolate software and the environment, ensuring platform independence. Containers can be shut down, restarted and changed independently without directly affecting the running of any other service.

It also allows us to run each service individually and thus efficiently optimize costs – e.g. by scaling performance or shutting down servers completely when the service is not in use. This would not be possible with a monolith architecture.

We provided the automated build requirement for the deployment using Github CI/CD, which provides tools to automate this process in addition to versioning the code. Thus, the developer's only concern is to upload code changes to the main repository. To minimize errors in the code, automated tests are run before each deployment to prevent changes from being introduced to the code if they fail.

3.3 Communication interface

The Flask framework[1] was used for development, which implements the usual best-practices in the field of web application development. It also enables fast deployment thanks to the WSGI communication interface, which is supported by an Apache web server. The framework works as a REST-API, i.e. that, based on the URI and HTTP method of the request performs action and returns the appropriate response, usually structured in JSON format. As a result, a single backend implementation can handle both web and mobile client requests.

During the development we put emphasis on the creation of documentation, which is often neglected resulting in a mismatch between the actual and documented functionality. To avoid this, we used tools that document individual API endpoints automatically. The result is the Swagger framework user interface, which clearly specifies how one can interact with our REST API.

3.4 Data storage

Two different repositories are used for the storage of persistent data. Articles are automatically retrieved by scraping from various websites. Unfortunately, each site has a different structure, which also changes over time, so we cannot rely on the consistency of the data obtained. Therefore, in the first phase, the data is stored in the non-relational database MongoDB. Non-relational databases do not store data in tables, but as so-called documents. Each document then stores the data in key-value form, but the internal structure of the documents may not be uniform (i.e., each document may contain different keys).

In the second phase, the data is normalized and stored in the traditional PostgreSQL relational database. In this database, we store structured information about each article, such as the title, date of publication, author abstract and, of course, the textual content of the article.

To avoid having to modify the scraper every time the site structure changes, we implemented an intermediate step where we transform data from a non-relational MongoDB database into a relational PostgreSQL database. In addition to changing the structure, this intermediate step allows us to keep track of changes to the article content itself that may have occurred over time.

[1] https://flask.palletsprojects.com/

3.5 Text processing

An important part of the Analytical platform is the analysis of published articles and news on portals dealing with the area of financial markets. This information is used in the Analytical platform as a basis for decision support when building an investment portfolio. When analyzing the content of articles, we mainly focus on:

- Summarization – the original long text can be shortened by the extractive or abstractive summarization technique to a paragraph containing a few sentences. The extractive technique is based on identifying and selecting the most important parts from the original text, the abstractive will generate a text containing the main topics of the article.
- Sentiment analysis – for the summarized content of the article, the sentiment is evaluated into one of 3 classes: positive, negative or neutral. The sentiment of each news can be evaluated from different perspectives, for example, information about the decline in electric car sales will not please Tesla shareholders, but on the other hand, oil company shares will probably rise.

3.5.1 Text summarization

We analyzed several different approaches from the point of view of both abstractive and extractive summary creation. We compared outputs of our own solutions and commonly available libraries (e.g. Sumy[2], Python Text Summarizer[3]) in Chochula (2021).

The summary, which contains essential information from the whole article, was created in an extractive way. The Gensim library from the Python programming language was used to create the summary. The extraction process is based on the TextRank algorithm (Mihalcea and Tarau, 2004), which is a variant of the PageRank (Brin and Page, 1998) algorithm used by Google to evaluate the relevance of websites.

Due to dissatisfaction with the outputs obtained from the above-mentioned approaches, we decided to focus on newer algorithms based on large language models (LLM). As Deng (2023), Zhang (2023a), Zhang (2023b) or Sun (2023) state, the use of LLM for sentiment analysis in news and articles from the field of financial markets is possible. The same approach can be applied in the case of summarizing the content of reports and articles from this area. The advantage of using LLM is the possibility of generating abstract summaries, which allows us to achieve a more readable level of content compared to the extractive approach while maintaining the same information value. The use of LLM in the field of summarization is mentioned, for example, by the Laban (2023), Sarode (2023) or Zhang (2023c).

The goal is to get information about the content of the article summarized in a few sentences and structured in JSON format. This is important for further processing and presentation of the content. In order to get output from LLM in the required format, we need to create a suitable prompt. Preparing a prompt is not always simple and straightforward. We arrived at the desired result after several iterations and modifications. The key requirements that had to be included in the prompt were a list of individual parts that should appear in JSON format – a summary of the entire article and an array of strings in the form of bullet points. It was also important to state that the output should be formatted in accordance with the RFC8259 specification defining the structure of the JSON format. The last key point made in the prompt included a demonstration of how the result should be formatted. Based on these requirements, we subsequently obtained exactly the output we required.

[2] https://pypi.org/project/sumy/

[3] https://www.geeksforgeeks.org/python-text-summarizer/

```
  {
  “summary”: “Apple is rumored to bring the Action button, a capacitive and
pressure-sensitive feature, to all iPhone models in the iPhone 16 lineup in
2024. Previously, this feature was only planned for high-end Pro models. Sources
suggest Apple intends to provide consistency for developers across its entire
product line.”,
  “bulletpoints”: [“Apple is planning to make Action button capacitive and
pressure sensitive”, “Action button will be available on all iPhone models in
iPhone 16 lineup”, “Consistency is the reason behind bringing Action button to
entire lineup”]
  }
```


3.5.2 Sentiment analysis

For sentiment analysis, we used several deep learning algorithms according to the state of the art in this field, such as BERT (and its derivatives) or LSTM (Šťastná, 2022). In general, the drawback of machine learning is the need for a large amount of labeled data to train and build the model. Sentiment analysis is a domain-oriented task; a model trained with data from a certain domain (e.g., movie reviews) usually performs poorly in another domain (e.g., financial news). On the other hand, data from the financial domain is limited and manual labeling is expensive and time-consuming.

Therefore, we collect data from available sources on the Internet. Specifically, we use the web scraper tool to obtain data from seekingalpha.com. The articles on this site are categorized into “Bearish” (market declines) and “Bullish” (market rising) (see Fig. 1). In this terminology, we consider bullish news to be positive and bearish news to be negative. In total, we have collected more than 8.5 thousand articles that we can use to train the model.

We compared several algorithms and inputs for this task and tried to uncover how article length affects the accuracy of sentiment prediction. We attempted to classify sentiment for three types of inputs – article title only, article summary, and full article text. As shown in Fig. 2, all methods achieved the highest F1 score for article summaries, which suggests that the article title does not contain enough information; on the other hand, the raw article text is likely to contain unusable features that are not relevant for sentiment analysis.

Due to the excellent results, we obtained when using LLM in the area of summarizing the content of articles, we decided to use this modern approach in the area of sentiment identification as well. A big advantage is that it is possible to determine the sentiment not only for the entire article, but also for individual tickers that appear in the text, including the justification for why the rating is positive or negative. The output is again formatted in JSON. Part of

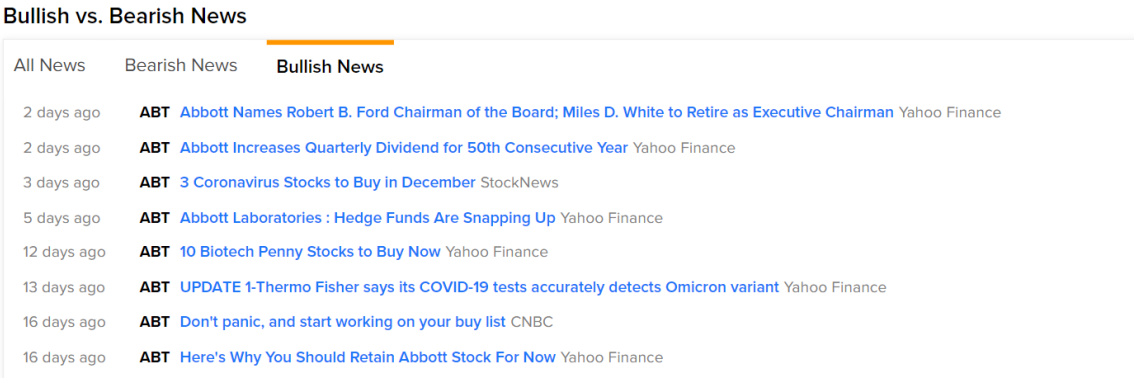

Fig. 1: Bearish and bullish news on Seeking alpha website

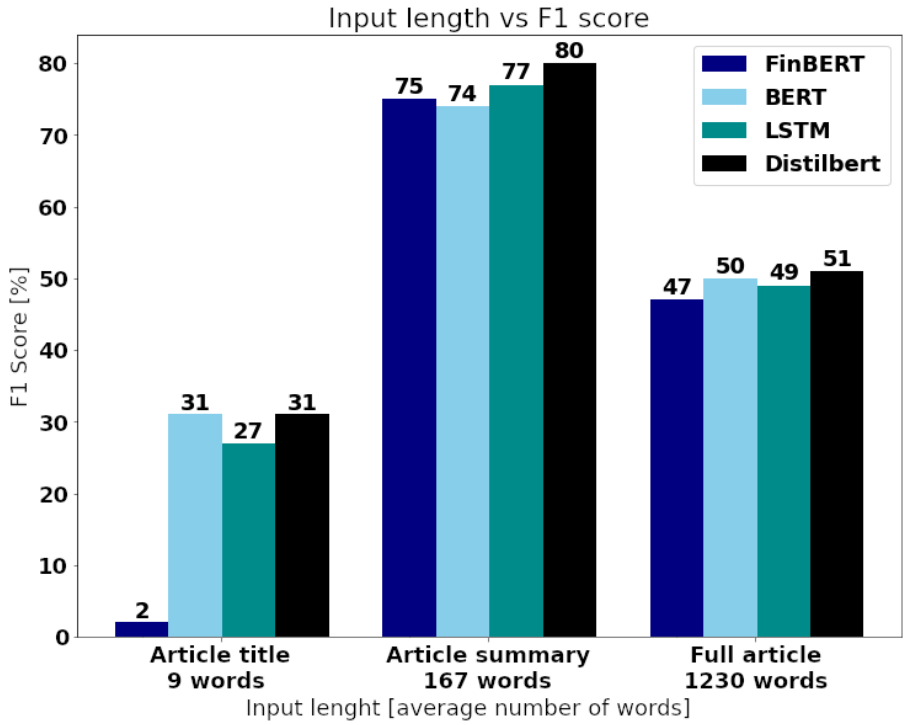

Fig. 2: Performance of methods depending on input length

the prompt again, as in the case of summarization, is the exact specification of the required structure of the JSON output format and the presentation of an example of how the output should look. In the prompt, there is a request to determine the sentiment for the entire article and individual tickets separately.

```
  { “general”: {
          “sentiment”: “negative”,
          “reasoning”: “The text mentions that multiple companies are
struggling, with Tesla reporting a miss on both top and bottom lines,
IBM’s sales coming in just shy of expectations, and Alcoa experiencing a decline
in sales.”
    }
    “companies”: [
          {
          “id”: 5011, “symbol”: “TSLA”, “sentiment”: “negative”,
          “reasoning”: “The text mentions that Tesla is struggling with
declining sales and profit margins, and the company has cut prices multiple times
this year, indicating a negative sentiment.”
          },
          {
          “id”: 4684, “symbol”: “IBM”, “sentiment”: “neutral”,
          “reasoning”: “The text mentions IBM’s mixed results with profit
beating expectations but sales coming in just a bit short of what analysts were
looking for.” } ] }
```

For the reasons stated above, we decided to replace the summarization of articles using the Gensim tool as well as the sentiment classification implemented until now using the BERT tool with a language model that handles both of these tasks.

4 DISCUSSION AND CONCLUSIONS

The presented system is currently used by Cyrrus Corporate Finance, a.s. for their daily usage. Based on client requirements, additional information sources are regularly added and the portfolio of analyzed articles is still expanding. The list of long-term analyzes presented in the form of graphs is also under development. The goal is also to integrate the created API into a complex application. This will allow regular users to create their own business strategies, where one of the key attributes may be the sentiment of published articles

There are a number of similar projects available on the Internet that offer analysis of articles in the area of financial markets and securities trading. Sentimentrader provides daily sentiment report and ad hoc reports if there is anything especially timely or unusual. The Daily Sentiment Report includes an overview of where short- and intermediate-term sentiment is each day, along with updates on indicator extremes or studies focused primarily on sentiment, breadth and price action (Sentimentrader, 2021).

Sentdex (2021) is an application providing sentiment analysis in Finance, Politics and Geographical area. The Financial analysis shows the sentiment for selected companies and long-term analysis of S&P 500 Index. The analysis is available for 7 days, 30 days, 6 months and 1 year interval. The stock price is available in the presented chart too.

Stockgeist is an interactive platform for monitoring the current popularity of 2200+ publicly traded companies. It provides ranking based on the number of published messages about companies, stock overview, aggregated news from various websites, stock sentiment analysis based upon social media messages, wordcloud with the most popular topics the crowd is talking about, fundamentals such as market capitalization and P/E ratio, short story on the selected company to give a general idea of what they do and charts that display historical data about social media messages and positivity. Stockgeist (2021)

Stocktwits is the platform focusing on the same problem. It shows what actual investors and traders are saying in real time about stocks, crypto, futures and forex. (Stocktwits, 2021)

We can see from this overview the sentiment analysis is a very important task in natural language processing and the financial analysts are really interested in information published in articles and news. We know from a discussion with our customer that even a summarized view of published texts is very important. However, other applications do not provide this functionality, which we consider an important advantage of our solution. Our application also offers sentiment analysis related to individual tickers mentioned in the article, which other applications do not allow. The disadvantage of Analytical platforms is the absence of information related to cryptocurrency trading

Our research and experiments show that using LLM for content summarization and sentiment analysis provides better results than procedures based on traditional approaches. LLMs enable not only the creation of higher-quality outputs, for example in the form of abstract summarization of the text, but also more advanced analyzes that were complicated using traditional approaches, such as aspect-based sentiment analysis. Therefore, the tools offered by the Analytical Platform are now based on this technology. The software is available at https://test.analyticalplatform.com/dps/login.

Acknowledgements

This paper was supported by the project CZ.02.1.01/0.0/0.0/16_017/0002334 Research Infrastructure for Young Scientists, this is co-financed from Operational Programme Research, Development and Education and by the project EG17_176/0015722 – Data processing software (2020–2023, MPO/EG).

REFERENCES

BRIN, S. and PAGE, L. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7), 107–117.

CHEN, L. 2018. Microservices: architecting for continuous delivery and DevOps. In: *2018 IEEE International conference on software architecture (ICSA),* pp. 39–397. IEEE.

CHOCHULA, P. 2021. Sumarizace dokumentů z oblasti finančních trhů, Final thesis. Mendel University in Brno. https://is.mendelu.cz/lide/clovek.pl?id=71901;zalozka=7;zp=69217;studium=107183

DENG, X., BASHLOVKINA, V., HAN, F., BAUMGARTNER, S. and BENDERSKY, M. 2023. What do llms know about financial markets? A case study on reddit market sentiment analysis. In: *Companion Proceedings of the ACM Web Conference 2023*, pp. 107–110.

FOWLER, M. and LEWIS, J. 2015. Microservices: Nur ein weiteres Konzept in der Softwarearchitektur oder mehr. *Objektspektrum*, 1(2015), 14–20.

JAKÚBEK, T., VALOVIČ, R. and PŘICHYSTAL, J. 2022. Summary of Financial Articles. In: *24th International Conference Economic Competitiveness and Sustainability: Abstracts. 2022*. Brno: Mendelova univerzita v Brně, 18. ISBN 978-80-7509-826-9

LABAN, P., KRYŚCIŃSKI, W., AGARWAL, D., FABBRI, A. R., XIONG, C., JOTY, S. and WU, C. S. 2023. SummEdits: Measuring LLM Ability at Factual Reasoning Through the Lens of Summarization. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9662-9676.

MIHALCEA, R. and TARAU, P. 2004. Textrank: Bringing order into text. In: *Proceedings of the 2004 conference on empirical methods in natural language processing*, pp. 404–411.

NADAREISHVILI, I., MITRA, R., MCLARTY, M. and AMUNDSEN, M. 2016. *Microservice architecture: aligning principles, practices, and culture*. O'Reilly Media, Inc.

PŘICHYSTAL, J., JAKÚBEK, T. and VALOVIČ, R. 2023. NLP Studio. Working paper. In: *25th International Conference Economic Competitiveness and Sustainability: Abstracts.* Brno: Mendelova univerzita v Brně, 2023. 45. ISBN 978-80-7509-912-9. https://akela.mendelu.cz/~jprich/prichystal-NLPstudio.pdf

SARODE, K., JAVAJI, S. R. and KALAKONNAVAR, V. 2023. Question-focused Summarization by Decomposing Articles into Facts and Opinions and Retrieving Entities. arXiv preprint. *arXiv*: 2310.04880.

SENTDEX. 2021. *Sentdex* [online]. http://sentdex.com/.

SENTIMENTRADER. 2021. Sentimentrader [online]. https://sentimentrader.com/.

STOCKGEIST. 2021. *Stockgeist* [online]. https://www.stockgeist.ai/.

STOCKTWITS. 2021. *Stocktwits* [online]. https://stocktwits.com/.

SUN, X., LI, X., ZHANG, S., WANG, S., WU, F., LI, J., ... and WANG, G. 2023. Sentiment analysis through llm negotiations. arXiv preprint. *arXiv*: 2311.01876.

ŠŤASTNÁ, P. 2020. *Analýza sentimentu v oblasti financí a obchodování s cennými papíry*. Final thesis. Mendel University in Brno. https://is.mendelu.cz/auth/lide/clovek.pl?id=68274;zalozka=7;zp=71400;studium=107655

ZHANG, B., YANG, H., ZHOU, T., ALI BABAR, M. and LIU, X. Y. 2023a. Enhancing financial sentiment analysis via retrieval augmented large language models. In: *Proceedings of the Fourth ACM International Conference on AI in Finance*, pp. 349-356.

ZHANG, W., DENG, Y., LIU, B., PAN, S. J. and BING, L. 2023b. Sentiment Analysis in the Era of Large Language Models: A Reality Check. arXiv preprint. *arXiv*: 2305.15005.

ZHANG, B., YANG, H., ZHOU, T., ALI BABAR, M. and LIU, X. Y. 2023c. Enhancing financial sentiment analysis via retrieval augmented large language models. In: *Proceedings of the Fourth ACM International Conference on AI in Finance,* pp. 349–356.

Contact information

Jan Přichystal: email: jan.prichystal@mendelu.cz

Roman Valovič: email: roman.valovic@mendelu.cz