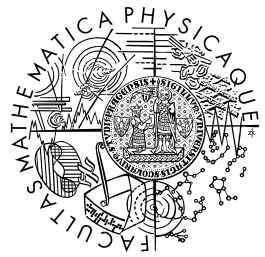

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Václav Stibor

Financial News Sentiment Analysis

Department of Software Engineering

Supervisor of the bachelor thesis: Supername Supersurname
Study programme: study programme
Study branch: study branch

Prague YEAR

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

Dedication. It is nice to say thanks to supervisors, friends, family, book authors and food providers.

Title: Financial News Sentiment Analysis

Author: Václav Stibor

Department: Department of Software Engineering

Supervisor: Supername Supersurname, department

Abstract: One key skill required to make good investments in the stock market is being able to correctly analyze news related to the finance and the business sector. Which company is diversifying its sectors or which company is showing signs of heading towards bankruptcy? You need to keep yourself updated with every little deal and fallout happening in the market. Financial news can be a little tricky to understand especially for those who are new to the financial world.

Keywords: key words

Contents

Introduction

In today's era of information explosion and constant flow of information, it becomes more time-consuming to keep track of associations and deeply understand the published content through media and online news, primarily when investing in a specific area. For instance, the investment in a company like Apple Inc. requires acquiring and processing a wide range of available information with significant effort and dedication in studying articles and other sources. At the same time, publicly available information resources such as news articles and tools like sentiment analysis allow us to transfer real-world context into the digital environment and use it for our benefit.

Sentiment analysis, the ability to identify and evaluate the emotional charge of content, has evolved into a crucial instrument for comprehending opinions, attitudes, and the general atmosphere surrounding various topics. This work focuses on developing an application that allows users to visualize connections between companies and news articles using a knowledge graph network and the impact of news sentiment on a company's stock price, even in real time.

Many experiments are currently being conducted based on historical data to examine the effect of sentiment, but not on current data, despite the rather promising results on datasets. The absence of such an application motivates this thesis. An application that extracts actual data from news articles for sentiment analysis and subsequently evaluates the future impact of that sentiment on a company's stock price.

This thesis will discuss the technical aspects of sentiment analysis and implementing an application that conveys this information to users as recently as possible. The aim is to provide users with a tool that allows them to actively monitor and analyze the flow of information about emotional overtones as one of the key identifiers in trading decisions. The thesis will be structured as follows. Chapter 1 will discuss the theoretical background behind the stock market. Chapter 2 will give an overview of data sources and the data itself. Chapter 3 will discuss the sentiment analysis and design of the application. Chapter 4 will discuss the implementation of the application. Chapter 5 will discuss the evaluation of the application. Chapter 6 will discuss the conclusion and future work.

1. Theoretical Background

Since the application's core is sentiment analysis, it is necessary to define the basic concepts. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus.

1.1 Sentiment Analysis Basics

Sentiment analysis or opinion mining is a subfield of Natural Language Processing (NLP) that aims to identify and extract opinions and emotions from a text. The goal is to determine the author's attitude towards a particular topic or the overall contextual polarity of various document levels. We measure the text's polarity using a numerical scale ranging from -1 to 1. The low-end score of the scale signifies a negative sentiment, zero represents neutrality, and the high-end score indicates a positive sentiment. This scale effectively estimates the degree of negativity or positivity in the text's tone.

The extraction of opinions and emotions has applications in various areas, from product reviews to political events. Hence, it is imperative to work in different domains (see Piryani et al., 2017). Because of cross-domain and cross-language, two of the most general issues in sentiment analysis, this thesis will focus only on the financial domain in English. Nevertheless, domain-specific sentiment analysis achieves remarkable accuracy while staying highly domain-sensitive, as shown in (Saunders, 2020). To delve deeper into cross issues, Liu provides further details in his book (Liu, 2022).

1.1.1 Levels of Sentiment Analysis

Sentiment analysis has been studied at several levels of granularity: Document-level, Sentence-level, Phrase-level, and Entity-level[1], as illustrated in Figure 1.1.

[1]Entities are sometimes referred to as targets, hence Target-level or Target-based sentiment analysis.

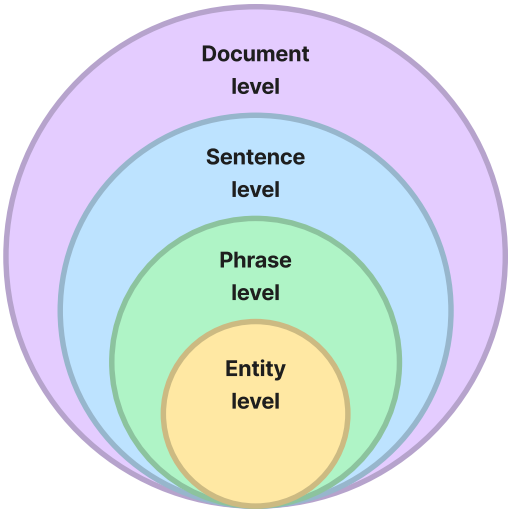

Figure 1.1 Levels of sentiment analysis (inspired by Wankhade et al., 2022).

Document-level

Document-level sentiment analysis is the most straight level. The task is to determine the overall emotional context of the entire document, such as a chapter, article, or review, whether or not involving a study of entities or aspects. This level gives us a general assessment of whether the content is more likely to be positive, negative, or neutral.

Sentence-level

Sentiment analysis at the sentence level focuses on individual sentences within the text. We observe the polarity of each sentence autonomously, employing the same methodologies utilized at the document level but with an increased volume of training data and enhanced processing resources. This level is more challenging than the document level because it requires a more in-depth understanding of the text.

Phrase-level

Phrase-level sentiment analysis examines sentiment within smaller linguistic units such as phrases or sentence members. Thus, it can better reveal the emotional charge in specific parts of sentences. Additionally, this level is more challenging than the sentence level because it requires a more detailed understanding of the text.

Entity-level

The most elaborative level is entity-level sentiment analysis, where we study sentiment associated with specific entities mentioned in the text. This level provides a detailed look at the expressed polarity of certain products, individuals, or organisations. One of the main tasks in this scope is the named entity recognition, which will be discussed later.

Some researchers classify the last level as the aspect-level, as noted by Wankhade et al., 2022, or a more detailed entity-level version called the feature-level proposed by Mary et al., 2017. While both approaches aim to evaluate sentiment towards specific aspects, they differ in their task approach. Relationships between these levels are illustrated in Figure 1.2.

In the first case, aspects are considered without directly mentioning entities in the text. We are not interested in the entities since the input textual data are commonly associated with them[2], such as reviews. The study conducted by Wang et al., 2019 analyzed sentiment at the aspect level within restaurant reviews. It primarily examines aspects such as food, price, service, and others. In the feature-based approach, aspects are commonly associated with an entity's features by connecting the entity and its aspects in text. To illustrate, consider the sentence:

"The battery life of this phone is excellent, but the camera is not good."

At the feature level, we identify *the battery life* and *camera* as specific features of entity *the phone*, allowing us to determine the polarity of each entity's feature.

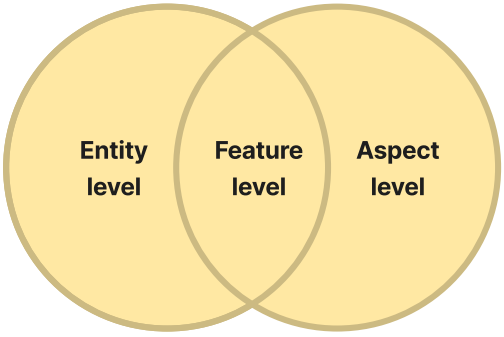

Figure 1.2 Comprehensive overview of the last level.

The term entity-level sentiment analysis is frequently employed in literature, and some studies consider it synonymous with targeted sentiment analysis,

[2]Entities are not handled in this case, but we provide them here for a better understanding.

as discussed Rønningstad et al., 2022 in the terminology review. For our purposes, entity-level sentiment analysis better captures the aggregate, document-wide approach, where a single entity can be associated with multiple targets in different sentences, discerning it from traditional target-level sentiment analysis.

However, this thesis primarily focuses on entity-level sentiment analysis, excluding consideration of the entity's features. This decision is motivated by treating the mentioned companies in news articles as entities rather than delving into their specific aspects. Additionally, entity and aspect extraction as separate tasks are complex and challenging, given that the methods and facets employed for recognition differ due to their distinct characteristics (Liu, 2015; Zhang et al., 2014).

TODO: Pokud nenarazím na článek, který by to vyvrátil. Navíc se zkoumáním aspektů by přibylo spousty práce.

1.2 Named Entity Recognition

Named entity recognition, or entity extraction, constitutes a fundamental component in NLP dedicated to identifying and classifying proper nouns into predefined semantic classes. These classes are unlimited since entities could be anything we can categorize using a tag, including the names of organisations, people, places, or other available information from unstructured textual data such as time, quantity, and money expressions. Someone well-versed in data analysis must have contemplated the possibility that some sources provide data in which the names of organisations are directly associated with their unique tags in text, but this is not our case (for more details, see Chapter 3). Understanding the importance of this task, we recognise its essential role in entity-level sentiment analysis, as it allows us to identify the emotion associated with specific entities mentioned in the text.

TODO: Vrátit se až budu mít všechny 3 zdroje a případně pozměnit na "né vždy náš případ."

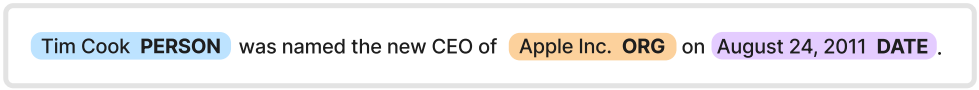

Figure 1.3 TODO: Named entities along with their associated label classes.

The example above illustrates that a single entity can contain more than one word. This challenge is addressed by token tagging formats outlined in the paper (Ramshaw et al., 1995). Individual words are referred to as tokens. The formats describe the position of each token in a named entity, such as the Beginning-Inside-Outside (BIO) format, also known as the Inside-Outside-Beginning (IOB) format as well as other derived names like Inside-Outside (IO) and Beginning-Middle-End-Whole-Outside (BMEWO). These formats discharge us from the constraints of entities, which are hard-coded or unreliably specified by regular expressions.

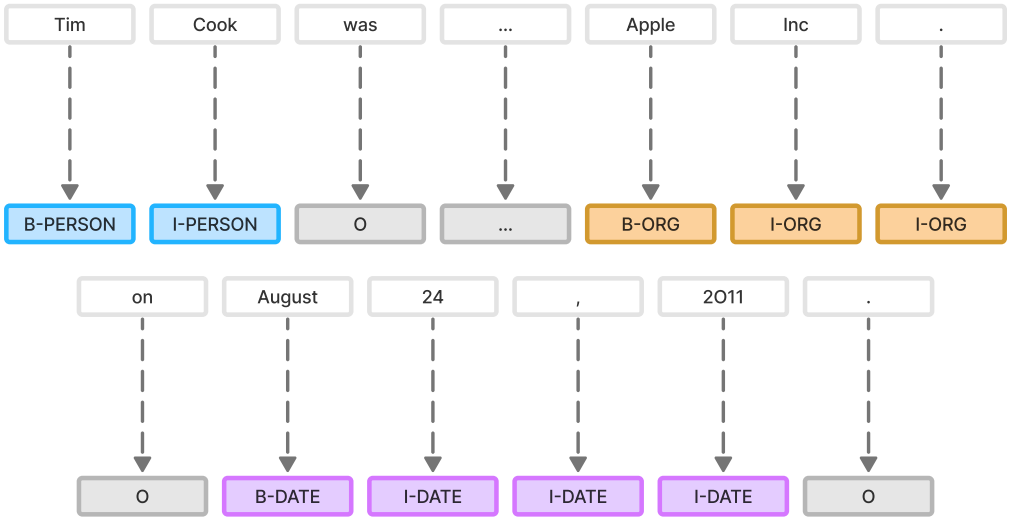

Figure 1.4 TODO - BIO2 format.

The tags that assign tokens to each entity class contain prefixes. The prefix I- indicates that the token is contained in the named entity, whereas O- indicates that the token is not contained in any named entity. The prefix B- is slightly more specific and indicates that the token is contained at the beginning of the named entity, followed immediately by a token not containing the O- prefix. A particular case of the BIO approach is the BIO2 format, denoting all tokens beginning with the prefix B-, regardless of whether a token with an O- prefix follows. A slightly more detailed approach is BMEWO, where the prefixes B- and O-, as in the previous ones, indicate the beginning and absence of the named entity occurrence, respectively. M- prefix symbolises the middle token between B- and E-, where the token with the prefix E- ends the named entity. Furthermore, the prefix W- indicates a single-token named entity. Table 1.1 below demonstrates how the mentioned sentence could be labeled with IO, BIO, BIO2, and BMEWO formats.

1.2.1 Techniques

TODO: Posunout k obrázku.

Approaches for entity extraction from unstructured data encompass a broad range of techniques, though they commonly converge into three principal categories. Specifically, the work (Keraghel et al., 2024) classifies them in the following way: Knowledge-based, Feature engineering, and Deep learning, as illustrated in Figure 1.5 below.

	Format			
Token	IO	BIO	BIO2	BMEWO
Tim	I-PERSON	I-PERSON	B-PERSON	B-PERSON
Cook	I-PERSON	I-PERSON	I-PERSON	E-PERSON
was	O	O	O	O
…	O	O	O	O
Apple	I-ORG	I-ORG	B-ORG	B-ORG
Inc	I-ORG	I-ORG	I-ORG	M-ORG
.	I-ORG	I-ORG	I-ORG	E-ORG
on	O	O	O	O
August	I-DATE	I-DATE	B-DATE	B-DATE
24	I-DATE	I-DATE	I-DATE	M-DATE
,	I-DATE	I-DATE	I-DATE	M-DATE
2011	I-DATE	I-DATE	I-DATE	E-DATE
.	O	O	O	O

Table 1.1 TODO

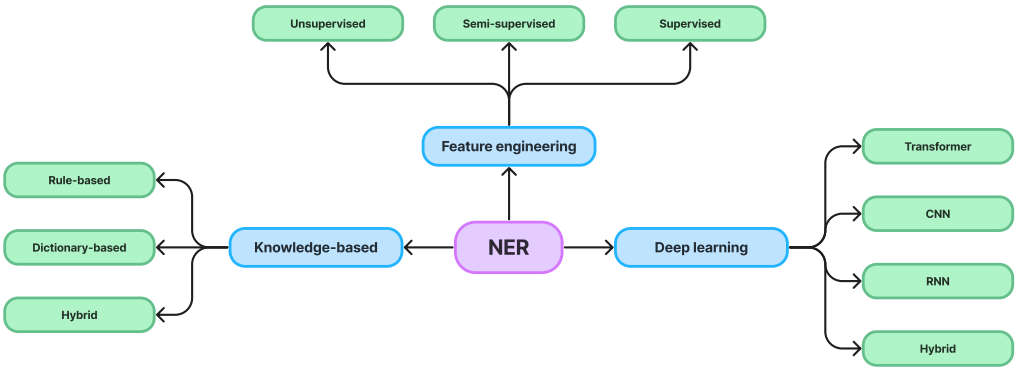

Figure 1.5 TODO: Named entity recognition main method approaches.

Knowledge-based

Knowledge-based approaches rely on predefined rules and dictionaries to identify entities. These rules and dictionaries, typically created by domain experts, recognise entities based on their characteristics. Rules can be given as regular expressions, denoting patterns to match character combinations in strings. Established on the chosen method, we separate this category into Rule-based and Dictionary-based or combine these approaches. The main advantage of knowledge-based approaches stems from their interpretability, as the rules and dictionaries can be easily understood and modified. Nevertheless, the princi-

pal disadvantage is frequent limitations to the entities in the manually created dictionaries and rules, as a result of which new entities cannot be recognised.

Feature engineering

Instead of manually creating a set of rules and a dictionary, feature engineering-based approaches, popularly identified as machine learning, use linguistic and statistical features to identify entities. These features are generally derived from the text and subsequently are used to train a machine learning model to recognise entities. The primary advantage of this approach is the ability to learn more about the data and discover patterns that may not be apparent at first. Additionally, their ability to recognise new entities differs from dictionaries in some cases. However, the main disadvantage is that these approaches require a large amount of labeled training data to be accurate.

Before exploring feature engineering-based methods in depth, we should clarify the data under discussion and the definition of labels. In the early introduction regarding named entity recognition, we referred to tags. These tags correspond to labels that serve as identifiers describing particular data that are consequently categorised according to the assigned label. To promote better comprehension, referring to the example illustrated in Figure 1.3, we have textual data in which labels are assigned as outlined in the following Table 1.2.

Data	Label
Tim Cook	**PERSON**
Apple Inc.	**ORG**ANISATION
June 8th, 2023	**DATE**

Table 1.2 TODO: Textual data with according labels.

Unsupervised learning The unsupervised learning method discovers patterns of entity occurrence in raw and unlabeled data. Consequently, the individual entities are split into groups based on their characteristic properties. The absence of pre-labeled data by human intervention in the training phase causes no supervisor to guide the model with information about the labels in training data, hence unsupervised learning. A conventional method employed in this approach is clustering, such as K-means clustering (Sinaga et al., 2020), which divides data[3] into groups based on similarity or dissimilarity.

[3]In our case, the data corresponds to tokens symbolising words.

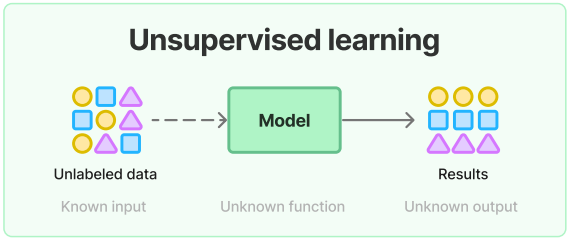

Figure 1.6 TODO: Unsupervised learning.

Semi-supervised learning The semi-supervised learning method combines labeled and unlabeled data, with the former comprising a slight portion of the dataset. As part of the classification training process, the unlabeled data learn the model's ability to generalise and represent the data in space using a statistical feature that better separates the classes. The algorithm aims to create the best decision boundary between classes based on a large amount of unlabeled data. Besides, the labeled data allows the model to determine the classification correctness for improvement, hence semi-supervised learning. Therefore, the model is partially supervised, hence semi-supervised learning. Semi-supervised learning is a preferred approach for model development because the labeled data is mainly expensive and time-consuming to acquire by requiring human intervention. In contrast, unlabeled data is more easily collectable.

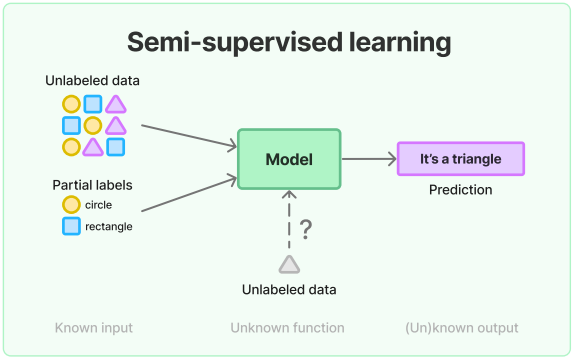

Figure 1.7 Semi-supervised learning.

Self-training, Co-training, and Multi-view learning are frequently employed subcategories of this learning method. Self-training involves using a small amount of labeled data and a more significant amount of unlabeled data,

repeatedly training the model on labeled data and using it to predict labels for unlabeled data. Co-training uses multiple independent models that communicate and share information to improve performance. Multi-view learning combines information from different sources or representations of data, such as combining textual data with metadata.

Supervised learning Unlike the semi-supervised learning method, the supervised learning approach exclusively utilises labeled data. Therefore, the training process is performed just on labeled data. We could perceive this approach as a function $f : X \mapsto Y$, denoted as f, mapping input X to output Y, where X and Y represent the input and output, respectively, known from the labeled data. Thus, as a supervisor guides the learning process, the model learns the most suitable way to map the input X to the input Y. The principal contrast between fully and semi-supervised learning is that in the former case, learning is done only over labeled data and in the latter case on a combination with unlabeled data. The most common algorithms utilised in supervised learning for named entity recognition include statistical models like Support Vector Machine (SVM) (Wang, 2005), Conditional Random Field (CRF) (Sutton et al., 2012), Maximum Entropy (ME) (Berger et al., 1996), and Hidden Markov Model (HMM) (Eddy, 1996).

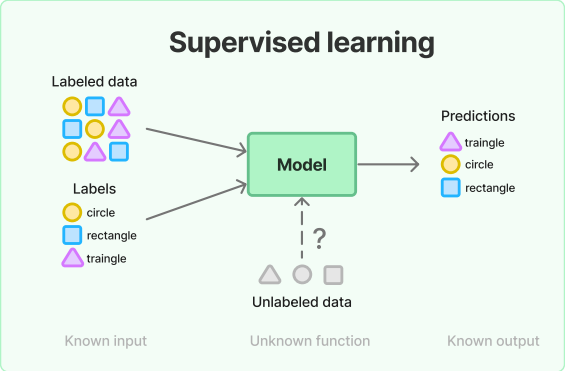

Figure 1.8 TODO: Supervised learning.

Deep learning

A given text could be extensive and contain many sentences or paragraphs mentioning entities in various forms. Therefore, deep learning comes into play due to its prowess in achieving better comprehension and learning contextual semantics. Its approaches overcome the primarily classification-focused methods introduced

thus far, leading to state-of-the-art results in entity recognition. To enhance elucidation of the semantic context, methods' consideration of words occurring before and after words possess a role. The overarching structure of this approach's most commonly employed methods, such as Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), and Transformer, is usually outlined into three fundamental stages: Data representation, Context encoding and Entity decoding.

1. **Data representation** is the first stage, where the input data are transformed into a format suitable for the model. We convert words into vector space to operate effectively in the following stages and maximise the potential of computational power. Transformations, collectively referred to as word embeddings, encompass techniques like One-Hot Encoding, Word2Vec (Mikolov et al., 2013), and Term Frequency-Inverse Document Frequency (TF-IDF) (Aizawa, 2003).

   (a) **One-Hot Encoding** consists of a vector representation of a word with a 1 occurring at its positional index in the dictionary and a 0 otherwise. The dictionary can be thought of as a sentence that is a sequence of words in a row.

   (b) **Word2Vec** projects words in a vector space that captures the meaning and relationships between other words. The projection into vector space allows the creation of a unique identification for each word in the corpus, with an emphasis on preserving syntactic and primarily semantic properties, making it possible to find synonyms after training on a large corpus.

   TODO?: Přidat zmínku o CBOW a Skip-gram?

   (c) **TF-IDF** as the name suggests, takes into account both the frequency of a word in a document and its occurrence across all documents in the corpus. Hence, a word's importance could be captured and then deduced whether it is an entity.

   Additional representations worth mentioning are FastText (Joulin et al., 2016), Global Vectors for Word Representation (GloVe) (Pennington et al., 2014), or Embeddings from Language Model (ELMo) (Peters et al., 2018). While these data representations may also find application in feature engineering-based methods, they must be met in the context of deep learning for effective textual data utilisation in subsequent processes.

2. **Context encoder** conveys the most crucial stage. Once we have the data in a suitable representation capable of capturing the context in which the word is found, the type of neural network selection to learn the model

contextual semantics of the input data comes next. As the deep learning methods approach is introduced, it is good to mention that "deep" refers to the depth of the layers in the neural network. Thus, the neural networks mentioned afterwards contain more than one hidden layer of neutrons, a layer that is neither an input nor an output layer. The articles (IBM, 2024b; IBM, 2024a) describe what neurons are and how deep neural networks differ from regular neural networks.

TODO!!! Následující část s mi nelíbí a pravděpodobně udělám ještě krátkou a ukázkovou sekci o Neural networks. Protože jsou využívány v Sentiment Ana ysis, Named Entity Recogn tion, Time Series Forecastin Text Similarity

3. **Entity decoder** is the final stage, where the model identifies and classifies entities in the text based on word representation and extracted context information. Entity decoding has two flavours based on the way the output of the previous stage is processed.

   (a) **Classification models** such as softmax regression or multilayer perceptrons (MLP) can be used to assign labels to individual words in the text based on their representation and context.

   (b) **Sequential models** such as recurrent neural networks (RNNs) or transformers are able to sequentially predict entity labels for individual words in a text given their context and relationships with other words.

RNN

CNN

Transformer 1. **BERT**

Large language models

1.3 Workflow of Entity-level Sentiment Analysis

The sentiment analysis process can be divided into three main steps: data retrieval, preprocessing, and analysis. The following sections will discuss these steps in more detail.

1.4 Time Series Forecasting Integration

We will focus on integrating time series forecasting in the context of news.

1.5 Text similarity

(Q-1.3) Zeptat se - Bude třeba úvod do stock marketu? - odpověď: ANO, bude třeba krátké intro.

2. Related work

While examining online applications with similar characteristics, a common challenge arises from the need for more transparency regarding the techniques applied in sentiment analysis. Specifically, employed methods and the details of the models remain undisclosed. We acknowledge this limitation due to the proprietary nature of the software, mainly as it is an essential part of the business model. Hence, this chapter is organized into two primary sections.

The initial part provides an overview of existing applications. At the same time, the second section will concern the most recent and relevant research on predicting stock market behaviour using data mining techniques and news sentiment analysis. Additionally, it explores research studies on entity-level sentiment analysis over news articles and its application in the stock market.

2.1 Existing Application Overview

Sentiment analysis applications accessible to the public are typically based on investigating social network posts, with StockTwist[1] as a notable illustration. According to Reuters (2022), StockTwits in 2022 boasts more than six million registered users and one million monthly active users, underscoring its prominent user base. With the growing volume of social media contributions, it is difficult to determine which post will prompt action. In our case, these posts do not constitute highly relevant data since our interest is in news articles containing a more significant amount of information. Even if the selection of accounts is limited to informative sources, News organizations' social media accounts only link to their articles, usually accompanied by a headline or lead paragraph. These posts are not enough for our purposes, as we need the entire article to perform better context for entity-level sentiment analysis (discussed further in Chapter 3).

Q-2.1 Lze takto citovat "pouze" webový článek deníku? Url je moc velký a chtěl bych se vyhnout tomu, abych ho tlačil do footnote

2.1.1 Bloomberg Terminal

The only software similar to the one created is a module in the so-called Bloomberg Terminal[2] from Bloomberg L.P.. This software system provides investors with analytical tools over financial data, including sentiment analysis of news articles and posts on social network X, formerly known as Twitter. The cost of

[1] *https://www.stocktwits.com*

[2] *https://www.bloomberg.com/professional/solution/bloomberg-terminal/*

TODO: a nebo the ?

a Bloomberg Terminal depends on the required specific features and services. A standard subscription typically amounts to approximately $24,000 per year. It is a very complex and powerful software, but it is not accessible to the general public.

Bloomberg, in its work (Cui et al., 2024), describes two types of sentiment analysis: story-level and company-level sentiment, utilizing a suite of Supervised Machine Learning (SML) techniques. Classification engines are trained on labeled datasets containing news articles and social media posts. Reportedly, the labeling process is based on the question:[3]

> *"If an investor having a long position in the security mentioned were to read this news or tweet, is he/she bullish, bearish or neutral on his/her holdings?"*

Once model training is completed, the models are employed to analyze recently published posts and articles associated with organizations, seeking distinctive sentiment signals related to the business and finance domain. As mentioned above, sentiment is divided into two levels:

Story-level Sentiment score value of articles and posts is calculated after arrival in real time. The calculation includes score and confidence, where the score has one of three options: positive, negative, or neutral, each described by a numerical value from the set $\{-1, 0, 1\}$. The confidence is defined by a value ranging from 0 to 1, demonstrating the intensity[4] of the sentiment. Hence, the story-level sentiment ranges from -1 to 1. For both, we get the following equation:

$$\textit{Story-level}_c^{\textit{Articles}} = S_c^a C_c^a, \quad a \in P(c) \tag{2.1}$$

$$\textit{Story-level}_c^{\textit{Posts}} = S_c^p C_c^p, \quad p \in P(c) \tag{2.2}$$

where a represents an article and p represents a post from $P(c)$, the set of published articles and posts referencing company c. S_c^a and S_c^p are the sentiment polarity scores of article a and post p that reference company c. C_c^a and C_c^p are the confidences of article a and post p that reference company c.

Company-level Sentiment score value is then calculated as the confidence-weighted average of the story-level sentiment scores, incorporating all relevant news articles and social media posts mentioning the company.

$$\textit{Company-level}_{c,t}^{Articles} = \frac{\sum_{a \in P(c,T)} S_c^a C_c^a}{N_{c,T}^a}, \quad T \in [t_b, t] \tag{2.3}$$

[3] This information is difficult to verify due to the unavailability of the dataset.

[4] Probability of being positive, negative, or neutral

$$\textit{Company-level}_{c,t}^{Posts} = \frac{\sum_{p \in P(c,T)} S_c^p C_c^p}{N_{c,T}^p}, \quad T \in [t_b, t] \tag{2.4}$$

where a represents an article and p represents a post from $P(c,T)$, the set of published articles and posts referencing company c during period T. Period T is a time interval of length t_b to t, where t_b is the time constant of the beginning. $N_{c,T}^a$ and $N_{c,T}^p$ are the number of articles and posts referencing company c during period T. In this way, the company-level sentiment is calculated as follows:

$$\textit{Company-level}_{c,t} = \textit{Company-level}_{c,t}^{Articles} + \textit{Company-level}_{c,t}^{Posts} \tag{2.5}$$

Intraday Company-level sentiment score for news articles is recalculated every two minutes, utilizing an eight-hour rolling window. The sentiment score for social network posts is recalculated every minute, employing a 30-minute rolling window. Due to the previous definitions, we can express these by a simple substitution of t_b depending on the rolling window as follows:

$$\textit{Intraday Company-level}_{c,t}^{Articles} = \frac{\sum_{a \in P(c,T)} S_c^a C_c^a}{N_{c,T}^a}, \quad T \in [t-8, t] \tag{2.6}$$

$$\textit{Intraday Company-level}_{c,t}^{Posts} = \frac{\sum_{p \in P(c,T)} S_c^p C_c^p}{N_{c,T}^p}, \quad T \in [t-0.5, t] \tag{2.7}$$

$$\textit{Intraday Company-level}_{c,t} = \textit{Company-level}_{c,t}^{Articles} + \textit{Company-level}_{c,t}^{Posts} \tag{2.8}$$

Daily company-level sentiment scores are published every morning about 10 minutes before the market opens. The calculation is determined as a confidence-weighted average of sentiment scores derived from the story-level sentiments of news and social media posts over the past 24 hours as follows:

$$\textit{Daily Company-level}_{c,t} = \frac{\sum_{d \in P(c,T)} S_c^d C_c^d}{N_{c,T}}, \quad T \in [t-24, t] \tag{2.9}$$

where document d represents a news article or social media post, the sentiment polarity score of document d referencing company c is denoted as S_c^d, and the confidence associated with this reference is represented by C_c^d. The set $P(c,T)$ encompasses non-neutral documents referencing company c published within the last 24 hours. $N_{c,T}$ expresses the count of non-neutral documents referencing company c during period T. This approach is further explored in terms of the informational role of social media by Gu et al. (2020).

2.2 Predicting Stock Market Behaviour

Several approaches for predicting stock market behaviour and price trends have been proposed, utilizing sentiment analysis of financial news and historical stock prices. Several studies prove a strong correlation between financial news sentiment and stock prices (Li et al., 2014) (Wan et al., 2021). Due to the nature of unstructured textual data, predicting stock market behaviour is a challenging task.

Khedr et al. (2017) focuses on creating an effective model for forecasting future trends in the stock market, using sentiment analysis of financial news and historical stock prices. The model achieves more accurate results than previous works by considering different market and company news types combined with historical stock prices. The experiments utilize datasets from three companies: Yahoo Inc., Facebook Inc.[5], and Microsoft Corporation. The authors use well-known and informative news sources such as Reuters, The Wall Street Journal, and Nasdaq. The first step of sentiment analysis to get the text polarity using a Naive Bayes classifier was shown to achieve accuracy from 72.73% to 86.21%, while the second step, which combines news sentiment with historical prices, improved prediction accuracy up to 89.80%. Moreover, that is why we find this study motivating and inspiring.

2.3 Entity-level Sentiment Analysis

This section will discuss the most recent and relevant research conducted in entity-level sentiment analysis and its application in news articles, including named entity recognition over news articles.

Zhao et al., 2021 employed RoBERTa, a Robustly Optimized Bidirectional Encoder Representations from Transformers (BERT) Pretraining Approach (Liu et al., 2019b), to propose a sentiment analysis and entity detection strategy in financial text mining and public opinion analysis in social media. In the first step, sentiment analysis, mainly focusing on negative polarity, is performed. Then, entity detection is considered in different granularities with MRC, Machine Reading Comprehension (Liu et al., 2019a), or sentence-matching tasks. As a result, this study serves entity detection differently than traditional Named Entity Recognition. The authors claim that the proposed method outperforms traditional sentiment analysis and entity detection methods. The authors also emphasize the importance of the financial domain, where the sentiment of a single entity can significantly affect the stock price.

[5]Known as Meta since 2021

Named entity recognition plays a significant role in entity-level sentiment analysis, as discussed in Section 2.3. Vybírám články, které stojí za zmínku.

2.4 Mining dynamic Social Networks

Ještě si nejsem jistý, zda-li je tato sekce potřebná.

This section will discuss interesting research Jin et al., 2012 dealing with mining dynamic social networks and its application in the stock market.

3. Textual data

ROZPRACOVANÉ, NEMÁ SMYSL ČÍST.

The integration of data, particularly newspaper article content, constitutes a fundamental component within the framework of our web application. We must consider several essential aspects to integrate these data into our web application to ensure a smooth and effective implementation. The following chapter will discuss these aspects from different perspectives, including the programmer's viewpoint and legislative considerations.

In this chapter, we will first present the data source options and then the aspects we will explore for each. In section X.X. we will give an overview...

3.1 Aspects for considerations

refer to chapter Related Works, where we discuss maybe why others work only with titles, describe why is better whole text and not only headlines

For the purpose of entity-level sentiment analysis, it is necessary to retrieve the entire content of each article, including full body text, as we detailed in Chapter 1 . This requirement complicates the actual development process from the beginning, especially since building an application on a dataset from the past would be inefficient as it would not include current news coverage and would be useless to the user. Therefore, we address this unexplored data problem.

When selecting a data source for news article content, it is essential to consider several main aspects.

Reliability Expresses the degree to which a source can be trusted based on its history and reputation.

Availability Expresses the degree to which a source is available to the public.

Accessibility Refers to the ease with which the data source can be accessed. Consider factors such as API availability, data retrieval methods, and any restrictions on accessing the news articles.

Consistency Look for a data source that maintains a consistent format and structure, facilitating easier integration into your web application.

Licensing and Copyright Ensure compliance with legal considerations. Verify the licensing terms and copyright issues associated with using the news articles in your application.

This thesis will mainly focus on the API caused by accessibility and functionality. In our case, RSS (Rich Site Summary) feeds are not very appropriate as a format for providing regularly changing web content.

3.2 Data sources

3.2.1 Web Scraping

3.2.2 RSS Feeds

3.2.3 News publisher's APIs

3.2.4 Third party data providers

3.3 First party data providers

3.3.1 The New York Times

3.3.2 The Guardian

The Guardian is a British daily newspaper that covers American and international news for an online, global audience.

We will focus to three basic domains.

3.4 Third party data providers

bla bla

3.4.1 Alpha Ventage

Application programming interface (API) třetích stran jsou dostupná v různých cenových plánech. Každý plán poskytuje odlišný rozsah přístupu k datům, který typicky spočívá v rozsahu dat, jenž jsou v rámci daného plánu dostupná. Dalším nejběžnějším omezení je maximálním počtem dotazů v rámci specifikované časové periody. Drtivá většina poskytovatelů nabízí bezlpatné plány, díky kterým může vývojář otestovat různé endpointy a ověřit, zda odpovídají požadavkům jeho aplikace.

There is always some compromise at the expense of something else (, a proto bychom naší apilikaci dokázali omezit na počet dotazů tak). V naší aplikaci

bychom se dokázali omezit na počet dotazů tak, abychom mohli uvažovat i bezplatného plánu anižbychom přišli o endpointy, jenž jsou pro naši aplikaci důležité. Avšak data obsahují častokrát velké mezery. Například Alpha Ventage poskytuje vyhledávání článků na základě tickeru a možností uvedení time range, ve kterém byly články vydány.

Vypadá to, že se můžeme dotazovat pouze na články v intervvalu 5 dní, avšak tento fakt není nikde v API zaznamenán.

Vždy jě něco na úkor něčeho jiného.

Nutno podotknout, že někteří provideři poskytují i RSS feedy, ale spise se jedná o shromaždujici agregator, který agreguje články z různých zdrojů.

Nám jde o webovou aplikaci, tj. jako zdroj nepovažujeme dataset.

With the growing popularity of the internet, the web has become one of the largest mediums of information.

Stejně jako při zobrazování číselného dopadu zpráv o NFLX, chci porovnat dopad vyhledávacího štítku NFLX na sociální síti X. To by nám mělo ukázat, že jsme se vyhnuli možnosti setkat se s náhodnými příspěvky od náhodných uživatelů, což v našem případě můžeme nazvat datovým šumem, protože je nepravděpodobné, že by měly vliv na hodnotu firmy. Pokud bychom filtrovali NFLX na X zpravodajských zdrojů, stále bychom nedostali celé články, pouze odkazy na ně, což je zbytečné. To může také poukázat na chybu ve volbě našeho zdroje dat. Vložit sem a nebo do sekce Data?

4. Company to Symbol Linking

This chapter delves into the challenge of linking companies mentioned in the text to their corresponding ticker symbols and exchanges. It highlights the limitations of the naive approach that relies on static data and score thresholds. However, it also introduces a more robust approach to entity linking using Wikidata.

The cornerstone of the entity linking approach is the comprehensive Wikidata knowledge base. By leveraging structured data and relationships between entities, the entity linking method can effectively link companies to their ticker symbols, even if their names appear in different forms. Moreover, it eliminates the need for arbitrary score thresholds and provides access to wealthy company information.

The entity linking approach offers several advantages over the naive method, including increased accuracy, better handling of name variations, and access to comprehensive company data. However, it is crucial to recognize potential limitations. Nevertheless, the result shows that the entity linking approach using Wikidata is a powerful tool for linking companies to their ticker symbols, providing a flexible and accurate solution over the naive approach.

TODO: Přidat nebo už je to moc navíc? The implementation involves using a spaCy entity linker pipeline built on a knowledge base to link text entities with Wikidata entities of the organization type. Subsequently, SPARQL queries are used to extract ticker symbols and exchanges from Wikidata, covering direct retrieval, owner-based retrieval, and differentiated retrieval scenarios.

The chapter also distinguishes between a company and an organisation. A company certainly refers to an entity that trades on an exchange (has a ticker). In contrast, an organisation refers only to a potential candidate by named entity recognition that might be tradable on an exchange or associated with that company. In addition, the Python notebook[1] is provided for both sections of this chapter to guide the reader through the process by which the results were obtained.

TODO: První část posledního odstavce pravděpodobně přesunu do Stock Exchange kapitoly, kde se budou popisovat nějaké základní info.

4.1 Introduction

Before proceeding further into this chapter, let us briefly revisit the concept of named entity recognition. It is a classification task to recognise entities within a given text, including names of organisations, people, places, dates, and more. While named entity recognition assigns entities to classes based on the syntax and semantics of the text, it does not provide specific details about individual entities. The necessity to identify companies in articles arises from their potential appearance in various forms and references. To give an example, the company Microsoft might be mentioned as "*Microsoft*", "*Microsoft Corp.*", "*Microsoft Corporation*", or "*MSFT*". In cases where multiple syntactic variations referring to the

[1] In the directory /ipynbs/company-to-symbol-linking/.

same company occur in an article, named entity recognition categorizes them all as organizations. However, it is crucial to have additional information to confirm that they represent the same entity. Bringing together these diverse forms[2] into a single entity identifiable by a unique symbol is essential.

TODO: Je nutná tato footnote?

TODO: Dodat odkaz a zmínit identifikátor v sekci.

As discussed in the Stock Market section of Chapter 1, each company has a unique identifier called a ticker, used for stock market identification. This ticker must be assigned to every mention of the company within the article. Additionally, we need to ensure that every mention of a company has a corresponding ticker. The assignment facilitates the extraction of companies operating across various exchanges and helps eliminate unnecessary entities identified by named entity recognition. Not every organisation identified by named entity recognition, such as Greenpeace, the World Health Organization, or the United Nations, necessarily represents a company listed on the exchange. Moreover, this process will play a pivotal role in the subsequent stages of application development.

While some newspaper articles, such as those from Bloomberg, Reuters, and CNBC, may include a company's ticker symbol or another specific identifier immediately following its name in the text, as demonstrated by "*Microsoft (MSFT)*", this practice is generally the exception rather than the norm across all news sources. Our primary textual data source, the Guardian, does not contain information about company tickers in this way (for more details, see Chapter 3). Hence, the task involves associating a company's name with its symbol. This chapter will focus on strategies for addressing this challenge, aiming to identify the optimal approach for handling this issue.

4.2 Problem definition

Our objective is to address the challenge of assigning a unique identifier to each company mentioned in the text. Companies identified in the named entity recognition classification typically belong to the organisation class, narrowing our focus to entities categorised as organisations. In general, this challenge can be divided into three essential parts:

Input: An article's text.

Output: A set of companies with their unique identifiers.

1. Recognising companies mentioned in the article.

[2]Different forms often encountered in newspaper articles due to authors' inconsistencies and text length.

2. Assigning a unique identifier to each identified company, provided one exists.

3. Extracting a set of companies contained in the article.

The initial phase of the concern involves utilising named entity recognition to classify entities, as mentioned above, specifically focusing on those categorised as organisations. The subsequent step entails implementing a method to assign a unique identifier to each company. Hence, the second phase relies on matching company names against a database of companies and their detailed information. The third component is straightforward and involves the unification of all occurrences into a set of companies already assigned identifiers.

To ensure we take advantage of every mention of a company. In the subsequent sections, we will explore various methods that could address the given problem and select the most suitable one for our case. Let us consider the following excerpt from the article (Guardian, 2024) about deepfake technology published by the Guardian on February 25, 2024:

> *"Executives from Adobe, Amazon, Google, IBM, Meta, Microsoft, OpenAI and TikTok gathered at the Munich Security Conference to announce a new framework for how they will respond to AI-generated deepfakes that deliberately trick voters."*

The named entity recognition identifies Adobe, Amazon, Google, IBM, Microsoft, OpenAI, TikTok and Meta as organisations. It is good to note that TikTok is owned by ByteDance, not a publicly traded company. OpenAI finds itself in a similar position to that of a private company. Thus, buying directly from companies that own TikTok or OpenAI is impossible. However, a different scenario arises when discussing Facebook, Instagram, or Messenger, organisations owned by Meta, a publicly listed company on the exchange. The task is to assign the correct ticker symbol to each company mentioned in the article and to determine the stock exchange on which the company is listed. The following Table 4.1 lists the companies mentioned in the article with their official name, corresponding ticker symbol, and stock exchange.

According to Table 4.1, organisation names are not always identical to their official names. This prevalent discrepancy underscores the necessity to address this issue, which will be thoroughly examined in subsequent sections focusing on methods for linking company symbols. Consequently, our approach will leverage a database containing companies listed on exchanges and their corresponding tickers. This database will enable us to match the official names of companies with the recognised organisation entities extracted from the article. By obtaining the ticker symbol, we can confidently access additional information about the

Organisation	Official Name	Ticker	Stock Exchange
Adobe	Adobe Inc.	ADBE	NASDAQ
Amazon	Amazon.com Inc.	AMZN	NASDAQ
Google	Alphabet Inc.	GOOGL	NASDAQ
IBM	International Business Machines Corp.	IBM	NYSE
Meta	Meta Platforms Inc.	META	NASDAQ
Microsoft	Microsoft Corp.	MSFT	NASDAQ
OpenAI	N/A	N/A	N/A
TikTok	N/A	N/A	N/A

Table 4.1 Organisations identified in the Guardian's article about deepfake technology with their official names, ticker symbols, and stock exchanges.

company from other databases, including its stock exchange listing and other pertinent details.

4.3 Naive approach

Methods presented in this section are based on matching company names against an exchange static dataset of companies. The dataset contains a list of companies with their official name and ticker symbol. The naive approach involves comparing organisation names extracted from the article with the official names in the dataset. The corresponding ticker symbol and exchange are assigned to the organisation if a match is found. This straightforward approach is a good starting point for linking company names to their ticker symbols. However, it has its limitations, as it may not be able to handle variations in company names.

4.3.1 Database

The first step in implementing the naive approach is to obtain the data. We need a dataset containing company official names, ticker symbols, and information about relevant stock exchanges. The dataset should be in a format conducive to efficient processing, encompassing file formats such as CSV, JSON, and XML, or alternatively, be structured within a relational database table. We discovered suitable datasets of companies listed on various exchanges on the EODData website[3], with a particular focus on the NASDAQ, NYSE, and AMEX datasets. The datasets contain information about companies listed on exchanges, including

[3]https://www.eoddata.com

their official names and ticker symbols. Each exchange has its dataset, so we have information on which exchange the company is listed. The datasets are in CSV format and can be easily preprocessed, allowing us to match organisation names extracted from the article.

4.3.2 Data preprocessing

It would be beneficial to do some preprocessing before starting the matching process. This involves making a few simple adjustments to make it easier to compare individual strings. We execute these steps on the dataset of each exchange and the organisation names, which we are trying to match accurately. The preprocessing steps encompass:

Lowercase conversion Convert all characters to lowercase to ensure case-insensitive matching.

ASCII conversion Convert all characters to ASCII to remove any non-ASCII characters in the string.

Punctuation removal Remove punctuation to eliminate any special characters that could interfere with the matching process.

Common suffix removal Remove common suffixes such as "*Inc.*", "*Corp.*", "*Ltd.*", and similar terms, as their usage may be inconsistent across the article's company name and the dataset's official name.

Common word removal Elimination of common words such as "*holding*", "*company*", "*group*", and others, which may not be essential for matching the company name.

During the implementation, we discovered an effective Python library called Name Matching, available on GitHub[4]. This library enables data preprocessing and the use of the distance metrics discussed in the following subsections. Additionally, after data preprocessing, the Cosine similarity method is used to reduce the number of potential matches by converting strings to n-grams and applying a TF-IDF transformation. With the preprocessed and reduced data in hand, we are ready to advance to the matching process based on distance metrics.

Dopsat citace na cosine similarity a n-grams. + Ověřit správnost citace knihovny, ale ve footnote by měla být v pořádku - verzi zapíšu do ipynbs.

[4]https://www.github.com/DeNederlandscheBank/name_matching/

4.3.3 Fuzzy matching

Fuzzy matching, or approximate string matching, is a technique used to determine the similarity between two strings using distance metrics. The lower the distance, the more similar the strings are. In contrast to exact matching, which requires a perfect match, fuzzy matching allows working with data that may contain incomplete matches. Therefore, this method is beneficial when dealing with variations in company names. In our case, the fundamental aim of this matching approach is to identify the most similar company name from the dataset for each organisation name extracted from the article.

Discounted Levenshtein distance

The leading and most advantageous distance metric, particularly suited to our use case, is the Levenshtein distance (Levenshtein et al., 1966) used to calculate the number of single-character operations such as insertion, substitution, and deletion, needed to transform one string into another. When calculating the distance, we can utilise the ability to weight individual operations differently. Specifically, the discounted variant reduces the cost of adjustments made closer to the end of the string. This characteristic holds significant importance in scenarios involving company names.

To illustrate an example, let us examine the process of matching "*Amazon*" from the Guardian's article and "*Amazon.com Inc*" from the dataset. Following all preprocessing steps, the extracted entities become "*Amazon*" and "*Amazon.com*". In such cases, employing the discounted Levenshtein distance ensures that "*amazoncom*" is not considered significantly outlying from "*amazon*" within the metric system. This approach is crucial because it helps discern that any other company name with differing first three letters can not accurately refer to "*Amazon.com Inc*" aligning with our intended goal. In summary, variations in suffixes are more common for each company name than variations in prefixes.

Using all preprocessing steps and the sample of the Guardian's article in which we have extracted organisation entities, we get the most similar company name from the dataset with a score based on discounted Levenshtein distance for each. An exact match scores 100, while 0 signifies no similarity. The results are presented in Table 4.2.

Wighted Jaccard similarity

Another possible approach is Weighted Jaccard similarity, in which the distance metric is expressed by a measure used to compare the similarity between two token sets. One set obtains an organisation name in an article, while the other represents an official company name in the dataset. The Name Matcher library

Organisation	Official Name	Score
Adobe	Adobe Systems Inc	100
Amazon	Amazon.com Inc	74.450
Google	Neos Yield Premium Strategy Google [Googl] ETF	35.917
IBM	Ibio Inc	55.647
Meta	Kennametal Inc	37.796
Microsoft	Microsoft Corp	100
OpenAI	Open Bank	73.286
TikTok	Cytokinetics	27.436

Table 4.2 Extracted organisation names from the Guardian's article and their matches with official names in the dataset using the match quality scores based on the discounted Levensthein distance.

defines the Weighted Jaccard similarity for the article organisation name set X, the official company name set Y, and a weight w as follows:

$$sim_{Jaccard_w}(X, Y) = \frac{w \cdot |X \cap Y|}{w \cdot |X \cap Y| + |X \setminus Y| + |Y \setminus X|} \tag{4.1}$$

In terms of a two-by-two confusion table, this similarity is expressed as:

$$sim_{Jaccard_w} = \frac{w \cdot a}{w \cdot a + b + c} \tag{4.2}$$

Where the following definitions apply:

- $a = |X \cap Y|$: The number of words common to both sets (true positives).
- $b = |X \setminus Y|$: The number of words in set X but not in set Y (false positives).
- $c = |Y \setminus X|$: The number of words in set Y but not in set X (false negatives).

Using the weight w, which is set by default to 3, we apply the same preprocessing steps to our sample article as we did with the discounted Levenshtein distance. The results we obtained are shown in Table 4.3.

The results differ when we compare the search results using discounted Levenshtein distance (in Table 4.2) and Weighted Jaccard similarity (in Table 4.3). With discounted Levenshtein distance, the result for "*Amazon*" is 74.450, while the Weighted Jaccard similarity result is 78.261. In this case, there is an improvement in identifying the correct company name from the dataset. However, the "*Google*" result is 35.917 for discounted Levenshtein distance and 50 for Weighted Jaccard similarity. Even though the "*Google*" score is higher with Weighted Jaccard similarity, the correct company name is still not displayed. Instead, it shows the "*Neos*

Organisation	Official Name	Score
Adobe	Adobe Systems Inc	100
Amazon	Amazon.com Inc	78.261
Google	Neos Yield Premium Strategy Google [Googl] ETF	50
IBM	Ibio Inc	54.545
Meta	Kennametal Inc	47.368
Microsoft	Microsoft Corp	100
OpenAI	Open Bank	75
TikTok	Cytokinetics	39.130

Table 4.3 Extracted organisation names from the Guardian's article and their matches with official names in the dataset using the match quality scores based on the Weighted Jaccard Similarity.

Yield Premium Strategy Google [Google] ETF", which needs to be corrected. Our objective is to directly label the parent Google's company, Alphabet Inc., with the ticker symbol GOOGL on the NASDAQ exchange. Therefore, higher scores do not necessarily indicate finer accuracy in this context. Similar results can be observed for the other matches. The only minimal improvement is in the case of "*IBM*", where there is a 1.102 reduction in the score for the incorrectly labelled company name.

Token Set Ratio

The Token Set Ratio represents another possible approach to solving our problem. It prioritizes the strings' meaning over the original word order and duplicate word removal, making the method flexible for comparing text accuracy.

Let us consider two token sets, X and Y, denoting the name of an organization extracted from the text and an arbitrary company from the dataset, respectively. We also operate with the intersection of $X \cap Y$, representing common words between the two strings. The resulting similarity score is then determined by the highest value among the following three similarity combinations:

Q: Tady mi zase přijde na druhou stranu, že čísla nejsou potřeba. Mám číslování equations psát vždy a všude?

- The similarity between the article organisation name and common words.

$$sim(X, X \cap Y) \tag{4.3}$$

- The similarity between common words and the dataset company name.

$$sim(X \cap Y, Y) \tag{4.4}$$

- The similarity between the article organisation name and the dataset company name.

$$sim(X, Y) \tag{4.5}$$

Where sim denotes the similarity score calculated by SequenceMatcher ratio[5] from the difflib library in Python. Also, in this case, keeping all preprocessing steps as in the previous two discussed metrics makes sense. Again, we will use our sample of Guardain's article to demonstrate this approach. The results are shown in Table 4.4.

Organisation	Official Name	Score
Adobe	Adobe Systems Inc	100
Amazon	Amazon.com Inc	82.353
Google	Neos Yield Premium Strategy Google [Googl] ETF	100
IBM	Ibio Inc	66.667
Meta	Kennametal Inc	62.500
Microsoft	Microsoft Corp	100
OpenAI	Open Bank	83.333
TikTok	Cytokinetics	40

Table 4.4 Extracted organisation names from the Guardian's article and their matches with official company names in the dataset using the match quality scores based on the Token Set Ratio.

The Token Set Ratio achieves a higher score in most matches than the previous two similarity approaches. However, as we mentioned in our previous results discussion, this does not always point in the right direction. Ignoring the identical results for "*Adobe*" and "*Microsoft*", which we achieved for all naive approaches, there is an improvement in the case of "*Amazon*", which leads to the best result for the correct company name match. However, in the case of "*Google*", it is more of a deterioration as the score increases to 100 for the mislabeled company from the dataset. The same problem occurs for all other cases as we increase the score for incorrect labels.

Summary

Upon examining all the data (see Table 4.5) obtained using the presented metrics, we are still dealing with the problem of the highest scores that do not guarantee the correct identification of a company from the dataset. We can be more willing to try different combinations of parameters or different preprocessing steps. However,

[5]https://docs.python.org/3/library/difflib.html#difflib.SequenceMatcher.ratio

it needs more than the approximate string matching to retrieve "*Alphabet Inc*" for "*Google*"[6]. Another problem is the score bound, which we must determine to mark a company match as correct. In our case, we could choose a score ranging from 70 to 80, but this would mean the possibility of skipping some companies that could be correctly labelled while facing the problem of labelling the wrong company. The results are still insufficient. Therefore, we need to take a different approach to get the correct company labelling from the dataset with higher accuracy.

Organisation	Official Name	Score		
		DL	WJ	TSR
Adobe	Adobe Systems Inc	100	100	100
Amazon	Amazon.com Inc	74.450	78.261	82.353
Google	Neos Yield Premium Strategy Google [Googl] ETF	35.917	50	100
IBM	Ibio Inc	55.647	54.545	66.667
Meta	Kennametal Inc	37.796	47.368	62.500
Microsoft	Microsoft Corp	100	100	100
OpenAI	Open Bank	73.286	75	83.333
TikTok	Cytokinetics	27.436	39.130	40

Table 4.5 The match quality scores for extracted organisation names from the Guardian's article using different similarity measures.

4.4 Entity linking approach

This section focuses on a more sophisticated technique, addressing the shortcomings encountered with the naive approach discussed in the previous section 4.3. It presents a method that leverages the possibility of adding a trained knowledge base (spaCy, 2024) to a named entity recognition module to enhance information extraction. Utilising a knowledge base connected to a source offers significant advantages, as it allows us to fully exploit the context and meaning of the words identified by named entity recognition. The naive methods primarily relied on fundamental word similarity, which we have shown to be insufficient in some cases to solve our problem.

QA: Je citace správně? Případně co mi tam chybí? Definuju v latexu jako @misc, kde urldate=den vytvoření citace a year=2024, aby se v textu nezobrazovalo n.d.

It is essential to point out that we are concerned with more than just basic information, such as the fact that the extracted entity is an organisation, person, or event. We already have such data through named entity recognition processing. Therefore, the optimal solution is to link the already extracted organization

[6]Je to běžné, příklad facebook a meta

entities with specific companies traded on the stock exchange or have relevant relationships with them (Meta Platforms owns Facebook). Furthermore, it would be beneficial to eliminate the need to set arbitrary score thresholds to determine the correct company.

We aim to be independent of a static dataset and have access to dynamic online information with a more comprehensive structure that can evolve. Thus, we present methods that outperform our naive approach and can process text with greater accuracy and success despite possible relationships. In addition, we will not use the official name in the results since the actual approach does not matter to it compared to the previous matching approach, and in the future, we will be able to obtain it based on the ticker. The main task from this chapter's problem definition is to obtain the ticker and the exchange it is associated with.

4.4.1 Wikidata

The cornerstone of the linking approach is Wikidata[7], which meets all the criteria needed to achieve the desired results. Wikidata is a storage repository of structured data freely available online, easily readable, and editable by humans and machines. It is a part of the Wikimedia family, which includes Wikipedia, Wikibooks, and others.

Wikidata consists of items with unique identifiers, denoted as Q<number> as QID. In Figure 4.1, the identifier for the item labelled as Microsoft is Q2283. This item has the description "*American multinational technology corporation*" and several aliases (also known as). Items in Wikidata provide statements containing individual properties tagged P<number> and their corresponding values. These values can have various types, including multiple values, item values, quantitative values, and unknown or no values. Microsoft has a property *instance of*, which refers to multiple item values describing Microsoft as a software company, enterprise, technology company, and public company. In this case, the individual values refer to other items. Additional information can be found in Wikidata glossary (Wikidata, 2024).

4.4.2 Spacy Entity Linker

The most suitable solution for our problem, which meets the requirements and provides adequate integration with our spaCy implementation of the named entity recognition module, is the Spacy Entity Linker library in Python, available on GitHub[8]. This library creates a pipeline with an external knowledge base built on

[7] https://www.wikidata.org/

[8] https://www.github.com/egerber/spaCy-entity-linker

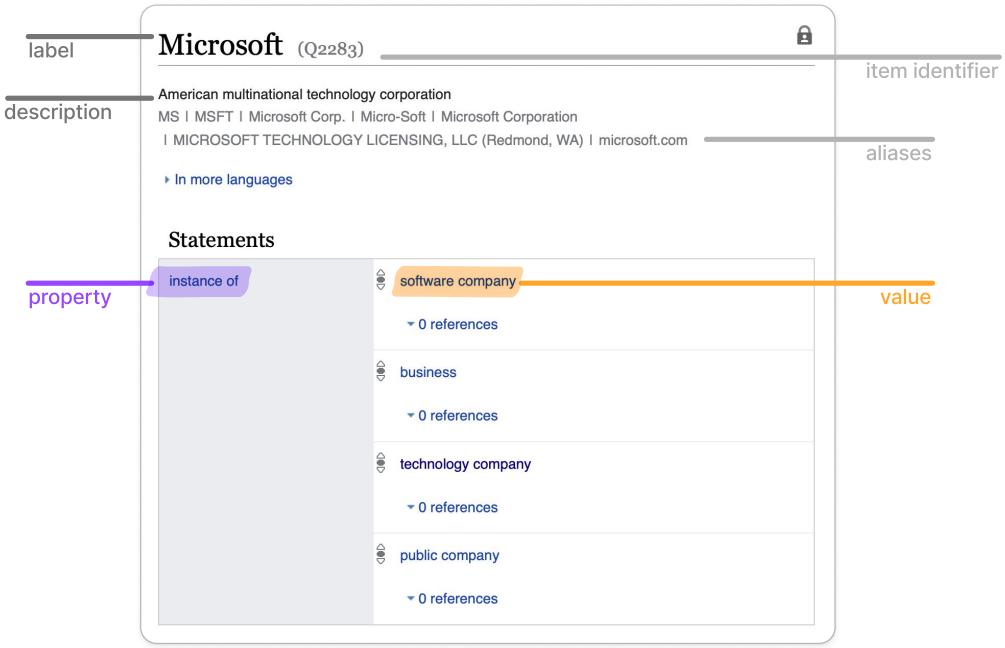

Figure 4.1 Wikidata item for Microsoft.

Wikidata that matches possible entities from the text with potential entities (the Entity Linker refers to each item in Wikidata as an entity) on Wikidata. The main advantage of this library is that each entity found in the text provides a QID on Wikidata, allowing us to obtain additional information about the entity, including its label, aliases, description, and more.

The main priority now is to filter out entities in the text that are not identified as organizations. The Entity Linker does not provide entity type information to determine if an entity is an organization. Instead, it only uses references to other entities via the properties *instance of* (P31), *part of* (P361) and eventually *subclass of* (P279) to determine the type. For example, when we extract the entity Microsoft, we obtain information about the categories (understood as types) it belongs to, such as software company, business, technology company, and public company, which can further branch into various subcategories. This allows an entity to be part of a wide range of possible classes. However, even if an entity belongs to the organization class according to our criteria, it may be labelled differently and miss one of the classes, causing us to lose the extracted entity. Therefore, we stick to the original partitioning associated with the organization type that handles the initial named entity recognition and compare it with the entities extracted from the entity linker using span. Additionally, staying with

the original implementation of spaCy entity objects in the code will make our work more manageable in the next phase of sentiment analysis and encourage code consistency.

The Entity Linker only uses the properties we mentioned when categorizing into classes. Thus, we can not directly get information about the ticker using the Entity Linker since the *stock exchange* property provides the ticker information. Regardless of aliases, it would be necessary to compare them with a database of tickers to see if one of the many aliases has an exact match. However, the Entity Linker provides us with the QID of the entity, which we can use to get any information about the entity. This opens up several possibilities for us to get this data. First, we tried the pywikibot[9] library, which works similarly to scraping. This approach is not ideal, as each entity page must first be created as an object and then loaded from the page repository, where the data is extracted. However, it is much more efficient and more accessible to query Wikidata using the Wikidata SPARQL endpoint, giving us more modularity and flexibility we want to maintain in the code.

4.4.3 SPARQL Wrapper

Access to Wikidata's SPARQL endpoint is enabled by the SPARQL Wrapper library in Python, which is available on GitHub[10]. Now, we have a tool that we can use to query data about entities with a given QID using SPARQL. We will show three queries that cover the most common situations where we want to get a ticker along with the name of the exchange on which the company is traded. The queries are run sequentially, with each subsequent query processing entity that did not produce a result in the previous query. We also mention the list of essential properties that we will use in the queries:

QA: Dlouho jsem hledal nějakou operaci jak sekvenční přístup nahradit a všechny queries sloučit do jedné. Nakonec se mi ale líbí, navíc můžeme přidávat další filtry za sebe.

- **P414**: The ***stock exchange*** on which the entity is traded.
- **P249**: The ***ticker symbol*** of the entity.
- **P582**: The ***end time*** of the property.
- **P127**: The entity ***owned by*** by another entity.
- **P1889**: The entity is ***different from*** another entity.

To better illustrate, we add an Instagram entity to the actual entities extracted from our sample Guardian article to demonstrate the second query associated

[9] https://www.github.com/wikimedia/pywikibot
[10] https://www.github.com/egerber/spaCy-entity-linker/tree/master

with the *owned by* property. The reader can try the following queries on the Wikidata SPARQL endpoint[11].

Query 1: Direct ticker retrieval

The first query aims to retrieve information about a *ticker symbol* and a *stock exchange* associated with an entity's *stock exchange* property. The SPARQL query can be found in Appendix A.1. This query selects entities that directly possess the *stock exchange* property, providing details about the ticker symbols and the exchanges on which the entities are traded. It also filters the results to include only those records where the exchanges do not have an *end time* specified. The results of this query are presented in Table 4.6.

Organisation		Ticker	Stock Exchange
QID	Label		
Q11463	Adobe	ADBE	NASDAQ
Q3884	Amazon	AMZN	NASDAQ
Q37156	IBM	IBM	NYSE
Q2283	Microsoft	MSFT	NASDAQ

Table 4.6 The results of the SPARQL Query 1: Direct ticker retrieval.

The data retrieved for individual organizations is accurate. Nevertheless, the results do not include the expected information about the Google entity. Although Google has a ticker through the *stock exchange* property on Wikidata, it also has an *end time* value set despite still being tradable on NASDAQ under the tickers GOOG and GOOGL. At the time of writing, this *end time* value can not be edited for unknown reasons, which presents a limitation we must accept. Consequently, additional queries are needed to determine details for the entities Meta, TikTok, OpenAI, and Instagram.

QA/TODO: Dodat, že není tolik běžné, že mají companies na námi specializoavných burzách více než 2 tickery a pro jednoduchost aplikace a struktury od každé entity vezmeme pouze jeden. Na druhou stranu jsem ještě nikdy neviděl API, které by zahrnovalo více než jeden ticker na company, protože mi přijde, že se od jednoho dají další odvodit. Pro jednoduchost a implementační záležitosti zatím volím raději pro jeden ticker. Co si o tom myslíte Vy?

Query 2: Owner-based ticker retrieval

The second query focuses on retrieving information about the exchange and ticker of the entities associated with the querying entity through an *owned by* relationship. The SPARQL query is displayed in Appendix A.2. This query selects entities with the *owned by* property and retrieves information about the tickers and exchanges associated with those entities. It explicitly targets entities related to the queried entity through the *owned by* property. The results of this query are shown in Table 4.7.

[11] https://www.query.wikidata.org/

Organisation		Ticker	Stock Exchange
QID	Label		
Q209330	Instagram	META	NASDAQ

Table 4.7 The results of the SPARQL Query 2: Owner-based ticker retrieval.

The retrieved data are again accurate. Meta Platforms indeed own Instagram, which is accessed through the Instagram *owned by* Meta Platforms association. In this case, we successfully filtered out the FB ticker, now META[12], due to the *end time* filtering.

Query 3: Differentiated ticker retrieval

The third query retrieves information about the exchange and ticker of entities associated with the querying entity through a *different from* relationship. The SPARQL query is presented in Appendix A.3. Similar to the owner-based ticker retrieval, this query selects entities with the *different from* property and retrieves information about the tickers and exchanges associated with those entities. It focuses on entities related to the queried entity through the *different from* property. The results of this query are shown in Table 4.8.

Organisation		Ticker	Stock Exchange
QID	Label		
Q18811574	Meta	META	NASDAQ

Table 4.8 The results of the SPARQL Query 3: Differentiated ticker retrieval.

Again, the obtained data aligns with what is generally accepted as accurate. However, it is essential to note that Meta represents multiple distinct entities (as many others) on Wikidata. Therefore, querying the *different from* property is necessary in this context to retrieve Meta Platforms.

Summary

Using the three queries outlined above, we have successfully covered a noteworthy amount of organizations that are directly traded or have relationships with companies traded on the exchange. As shown in Table 4.9, we achieved an almost one hundred per cent success rate. However, Google is not included in the results due to the previously mentioned issue with the *end time* setting. We must accept this limitation for now and will not alter our overall approach because of one

[12]https://www.bloomberg.com/quote/FB:US

entity. Regarding changing the data in Wikidata, a future opportunity to modify can arise.

Additionally, TikTok and OpenAI are absent in the results, which is expected since neither has a ticker or a relationship with a company traded on an exchange. Our querying approach allows flexibility, enabling us to create additional filtering queries as needed. This flexibility is a valuable advantage in refining our results.

Organisation		Ticker	Stock Exchange
QID	Label		
Q11463	Adobe	ADBE	NASDAQ
Q3884	Amazon	AMZN	NASDAQ
Q37156	IBM	IBM	NYSE
Q2283	Microsoft	MSFT	NASDAQ
Q209330	Instagram	META	NASDAQ
Q18811574	Meta	META	NASDAQ

Table 4.9 The results of the SPARQL queries for all extracted organisation names from the Guardian's article.

5. Architecture

Conclusion

In the conclusion, you should summarize what was achieved by the thesis. In a few paragraphs, try to answer the following:

- Was the problem stated in the introduction solved? (Ideally include a list of successfully achieved goals.)
- What is the quality of the result? Is the problem solved for good and the mankind does not need to ever think about it again, or just partially improved upon? (Is the incompleteness caused by overwhelming problem complexity that would be out of thesis scope, or any theoretical reasons, such as computational hardness?)

This is quite common.

- Does the result have any practical applications that improve upon something realistic?
- Is there any good future development or research direction that could further improve the results of this thesis? (This is often summarized in a separate subsection called 'Future work'.)

Bibliography

Piryani, R. et al. (2017). *Analytical mapping of opinion mining and sentiment analysis research during 2000–2015*. In: *Information Processing & Management* 53.1, pp. 122–150. ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2016.07.001. URL: https://www.sciencedirect.com/science/article/pii/S030645731630245X.

Saunders, Danielle (2020). *Domain adaptation for neural machine translation*. PhD thesis. Apollo - University of Cambridge Repository. DOI: 10.17863/CAM.66458. URL: https://www.repository.cam.ac.uk/handle/1810/319335.

Liu, Bing (2022). *Sentiment analysis and opinion mining*. In: Springer Nature. Chap. 3, pp. 31–36.

Wankhade, Mayur et al. (2022). *A survey on sentiment analysis methods, applications, and challenges*. In: *Artificial Intelligence Review* 55.7, pp. 5731–5780. ISSN: 1573-7462. DOI: 10.1007/s10462-022-10144-1. URL: https://doi.org/10.1007/s10462-022-10144-1.

Mary, A. Jenifer Jothi et al. (2017). *Jen-Ton: A framework to enhance the accuracy of aspect level sentiment analysis in big data*. In: *2017 International Conference on Inventive Computing and Informatics (ICICI)*, pp. 452–457. URL: https://api.semanticscholar.org/CorpusID:44112128.

Wang, Yequan et al. (2019). *Aspect-level Sentiment Analysis using AS-Capsules*. In: *The World Wide Web Conference*. WWW '19. San Francisco, CA, USA: Association for Computing Machinery, 2033–2044. ISBN: 9781450366748. DOI: 10.1145/3308558.3313750. URL: https://doi.org/10.1145/3308558.3313750.

Rønningstad, Egil et al. (Oct. 2022). *Entity-Level Sentiment Analysis (ELSA): An Exploratory Task Survey*. In: *Proceedings of the 29th International Conference on Computational Linguistics*. Ed. by Nicoletta Calzolari et al. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, pp. 6773–6783. URL: https://aclanthology.org/2022.coling-1.589.

Liu, Bing (2015). *Aspect and Entity Extraction*. In: *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, 137–188.

Zhang, Lei et al. (2014). *Aspect and Entity Extraction for Opinion Mining*. In: *Data Mining and Knowledge Discovery for Big Data: Methodologies, Challenge and Opportunities*. Ed. by Wesley W. Chu. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1–40. ISBN: 978-3-642-40837-3. DOI: 10.1007/978-3-642-40837-3_1. URL: https://doi.org/10.1007/978-3-642-40837-3_1.

Ramshaw, Lance et al. (1995). *Text Chunking using Transformation-Based Learning*. In: *Third Workshop on Very Large Corpora*. URL: https://aclanthology.org/W95-0107.

Keraghel, Imed et al. (2024). *A survey on recent advances in named entity recognition*. arXiv: 2401.10825 [cs.CL].

Sinaga, Kristina P. et al. (2020). *Unsupervised K-Means Clustering Algorithm*. In: *IEEE Access* 8, pp. 80716–80727. DOI: 10.1109/ACCESS.2020.2988796.

Wang, Lipo (2005). *Support vector machines: theory and applications*. Vol. 177. Springer Science & Business Media.

Sutton, Charles et al. (2012). *An introduction to conditional random fields*. In: *Foundations and Trends® in Machine Learning* 4.4, pp. 267–373.

Berger, Adam et al. (1996). *A maximum entropy approach to natural language processing*. In: *Computational linguistics* 22.1, pp. 39–71.

Eddy, Sean R (1996). *Hidden Markov models*. In: *Current Opinion in Structural Biology* 6.3, pp. 361–365. ISSN: 0959-440X. DOI: https://doi.org/10.1016/S0959-440X(96)80056-X. URL: https://www.sciencedirect.com/science/article/pii/S0959440X9680056X.

Mikolov, Tomas et al. (2013). *Efficient estimation of word representations in vector space*. In: *arXiv preprint arXiv:1301.3781*.

Aizawa, Akiko (2003). *An information-theoretic perspective of tf–idf measures*. In: *Information Processing & Management* 39.1, pp. 45–65. ISSN: 0306-4573. DOI: https://doi.org/10.1016/S0306-4573(02)00021-3. URL: https://www.sciencedirect.com/science/article/pii/S0306457302000213.

Joulin, Armand et al. (2016). *Bag of Tricks for Efficient Text Classification*. arXiv: 1607.01759 [cs.CL].

Pennington, Jeffrey et al. (Oct. 2014). *GloVe: Global Vectors for Word Representation*. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Alessandro Moschitti et al. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://aclanthology.org/D14-1162.

Peters, Matthew E. et al. (2018). *Deep contextualized word representations*. arXiv: 1802.05365 [cs.CL].

IBM (2024b). *Neural Networks*. URL: https://www.ibm.com/topics/neural-networks (visited on 03/08/2024).

— (2024a). *AI vs. Machine Learning vs. Deep Learning vs. Neural Networks*. URL: https://www.ibm.com/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks/ (visited on 03/08/2024).

Reuters (2022). *After crypto, Stocktwits takes aim at stocks trading*. URL: https://www.reuters.com/technology/after-crypto-stocktwits-takes-aim-stocks-trading-2022-07-19/.

Cui, Xin et al. (2024). *Embedded Value in Bloomberg News & Social Sentiment Data*. Received via email from Bloomberg L.P. on 2.2.2024. URL: https://data.bloomberglp.com/promo/sites/12/725454457_EDFSentimentWP.pdf (visited on 02/02/2024).

Gu, Chen et al. (2020). *Informational role of social media: Evidence from Twitter sentiment*. In: *Journal of Banking & Finance* 121, p. 105969. ISSN: 0378-4266. DOI: https://doi.org/10.1016/j.jbankfin.2020.105969. URL: https://www.sciencedirect.com/science/article/pii/S0378426620302314.

Li, Xiaodong et al. (2014). *News impact on stock price return via sentiment analysis*. In: *Knowledge-Based Systems* 69, pp. 14–23. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2014.04.022. URL: https://www.sciencedirect.com/science/article/pii/S0950705114001440.

Wan, Xingchen et al. (2021). *Sentiment correlation in financial news networks and associated market movements*. In: *Scientific Reports* 11.1, p. 3062. ISSN: 2045-2322. DOI: 10.1038/s41598-021-82338-6. URL: https://doi.org/10.1038/s41598-021-82338-6.

Khedr, Ayman E et al. (2017). *Predicting stock market behavior using data mining technique and news sentiment analysis*. In: *International Journal of Intelligent Systems and Applications* 9.7, p. 22.

Zhao, Lingyun et al. (2021). *A BERT based Sentiment Analysis and Key Entity Detection Approach for Online Financial Texts*. In: *2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 1233–1238. DOI: 10.1109/CSCWD49262.2021.9437616.

Liu, Yinhan et al. (2019b). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. arXiv: 1907.11692 [cs.CL].

Liu, Shanshan et al. (2019a). *Neural Machine Reading Comprehension: Methods and Trends*. In: *Applied Sciences* 9.18. ISSN: 2076-3417. DOI: 10.3390/app9183698. URL: https://www.mdpi.com/2076-3417/9/18/3698.

Jin, Yingzi et al. (2012). *Mining dynamic social networks from public news articles for company value prediction*. In: *Social Network Analysis and Mining* 2.3, pp. 217–228. ISSN: 1869-5469. DOI: 10.1007/s13278-011-0045-5. URL: https://doi.org/10.1007/s13278-011-0045-5.

Guardian, The (2024). *UK's enemies could use AI deepfakes to try to rig election, says James Cleverly*. Article discussing the potential use of AI-generated deepfakes in elections, featuring statements from James Cleverly, the UK's Home Secretary. URL: https://www.theguardian.com/uk-news/2024/feb/25/uks-enemies-could-use-ai-deepfakes-to-try-to-rig-election-says-james-cleverly (visited on 04/22/2024).

Levenshtein, Vladimir I et al. (1966). *Binary codes capable of correcting deletions, insertions, and reversals*. In: *Soviet physics doklady*. Vol. 10. 8. Soviet Union, pp. 707–710.

spaCy (2024). *KnowledgeBase · spaCy API Documentation*. URL: https://spacy.io/api/kb (visited on 06/09/2024).
Wikidata (2024). *Wikidata:Glossary - Wikidata*. URL: https://www.wikidata.org/wiki/Wikidata:Glossary (visited on 06/09/2024).

Acronyms

BERT Bidirectional Encoder Representations from Transformers. 17

BIO Beginning-Inside-Outside. 6, 7

BMEWO Beginning-Middle-End-Whole-Outside. 6, 7

CNN Convolutional Neural Network. 12

CRF Conditional Random Field. 11

ELMo Embeddings from Language Model. 12

GloVe Global Vectors for Word Representation. 12

HMM Hidden Markov Model. 11

IO Inside-Outside. 6, 7

IOB Inside-Outside-Beginning. 6

ME Maximum Entropy. 11

NLP Natural Language Processing. 3, 6

RNN Recurrent Neural Network. 12

SML Supervised Machine Learning. 15

SVM Support Vector Machine. 11

TF-IDF Term Frequency-Inverse Document Frequency. 12

A. SPARQL Wrapper

To ensure consistency in the text when comparing results with the naive approach methods, we will refer to the QID (*?id*) and Label (*?idLabel*) related to the Organization entity, as well as Ticker (*?ticker*) and Stock Exchange (*?exchangeLabels*) in the query result tables.

A.1 Query 1: Direct ticker retrieval

Listing A.1 SPARQL Query 1: Retrieve entity information for entities directly with the *stock exchange* property.

```
SELECT DISTINCT
  ?id              # Selects the entity ID
  ?idLabel         # Selects the label of the entity
  ?exchangesLabel # Selects the label of the exchange
  ?ticker          # Selects the ticker symbol

WHERE {
  # Retrieves labels in English
  SERVICE wikibase:label {
    bd:serviceParam wikibase:language
        "[AUTO_LANGUAGE],en".
  }

  # Specifies the QIDs of the entities
  VALUES ?id {
    wd:Q11463     # Adobe
    wd:Q3884      # Amazon
    wd:Q95        # Google
    wd:Q37156     # IBM
    wd:Q18811574 # Meta
    wd:Q2283      # Microsoft
    wd:Q48938223 # TikTok
    wd:Q21708200 # OpenAI
    wd:Q209330    # Instagram
  }

  # Specifies the QIDs of the stock exchanges
  VALUES ?exchanges {
    wd:Q82059     # NASDAQ
    wd:Q13677     # NYSE
  }
```

```
  # Matches entities with stock exchange property
  ?id p:P414 ?exchange.

  # Filters the exchanges to those specified
  # and retrieves the ticker symbol
  ?exchange ps:P414 ?exchanges;
            pq:P249 ?ticker.

  # Filters tickers without an end time
  FILTER NOT EXISTS {
      ?exchange pq:P582 ?endTime.
  }
}
```

A.2 Query 2: Owner-based ticker retrieval

Listing A.2 SPARQL Query 2: Retrieve entity information for remaining entities with the *owned by* property.

```
SELECT DISTINCT
  ?id              # Selects the entity ID
  ?idLabel         # Selects the label of the entity
  ?exchangesLabel # Selects the label of the exchange
  ?ticker          # Selects the ticker symbol

WHERE {
  # Retrieves labels in English
  SERVICE wikibase:label {
    bd:serviceParam wikibase:language
        "[AUTO_LANGUAGE],en".
  }

  # Specifies the QIDs of the remaining entities
  VALUES ?id {
    wd:Q95       # Google
    wd:Q18811574 # Meta
    wd:Q48938223 # TikTok
    wd:Q21708200 # OpenAI
    wd:Q209330   # Instagram
   }

  # Specifies the QIDs of the stock exchanges
  VALUES ?exchanges {
    wd:Q82059    # NASDAQ
    wd:Q13677    # NYSE
  }
```

```
  # Matches entities with owner property
  ?id wdt:P127 ?owner.
  ?owner p:P414 ?exchange.

  # Filters the exchanges to those specified
  # and retrieves the ticker symbol
  ?exchange ps:P414 ?exchanges;
            pq:P249 ?ticker.

  # Filters tickers without an end time
  FILTER NOT EXISTS {
      ?exchange pq:P582 ?endTime.
  }
}
```

A.3 Query 3: Differentiated ticker retrieval

Listing A.3 SPARQL Query 3: Retrieve entity information for remaining entities with the *different from* property.

```
SELECT DISTINCT
  ?id             # Selects the entity ID
  ?idLabel        # Selects the label of the entity
  ?exchangesLabel # Selects the label of the exchange
  ?ticker         # Selects the ticker symbol

WHERE {
  # Retrieves labels in English
  SERVICE wikibase:label {
  bd:serviceParam wikibase:language
    "[AUTO_LANGUAGE],en".
  }

  # Specifies the QIDs of the remaining entities
  VALUES ?id {
  wd:Q18811574 # Meta
  wd:Q48938223 # TikTok
  wd:Q21708200 # OpenAI
  }

  # Specifies the QIDs of the stock exchanges
  VALUES ?exchanges {
  wd:Q82059    # NASDAQ
  wd:Q13677    # NYSE
  }
```

```
  # Matches entities with the 'different from' property
  ?id wdt:P1889 ?differs.
  ?differs p:P414 ?exchange.

  # Filters the exchanges to those specified
  # and retrieves the ticker symbol
  ?exchange ps:P414 ?exchanges;
       pq:P249 ?ticker.

  # Filters tickers without an end time
  FILTER NOT EXISTS {
    ?exchange pq:P582 ?endTime.
  }
}
```