Extracting Entities and Relations in Analyst Stock Ratings News

Ivan Krstev[1], Igor Mishkovski[1], Miroslav Mirchev[1], Blagica Golubova[1], and Sasho Gramatikov[1]

Faculty of Computer Science and Engineering, Rugjer Boshkovikj 16, 1000 Skopje, North Macedonia https://www.finki.ukim.mk/en

Abstract. Massive volumes of finance-related data are created on the Internet daily, whether on question-answering forums, news articles, or stocks analysis sites. This data can be critical in the decision-making process for targeting investments in the stock market. Our research paper aims to extract information from such sources in order to utilize the volumes of data, which is impossible to process manually. In particular, analysts' ratings on the stocks of well-known companies are considered data of interest. Two subdomains of Information Extraction will be performed on the analysts' ratings, Named Entity Recognition and Relation Extraction. The former is a technique for extracting entities from a raw text, giving us insights into phrases that have a special meaning in the domain of interest. However, apart from the actual positions and labels of those phrases, it lacks the ability to explain the mutual relations between them, bringing up the necessity of the latter model, which explains the semantic relationships between entities and enriches the amount of information we can extract when stacked on top of the Named Entity Recognition model. This study is based on the employment of different models for word embedding and different Deep Learning classification architectures for extracting the entities and predicting relations between them. Furthermore, the multilingual abilities of a joint pipeline are being explored by combining English and German corpora. For both subtasks, we record state-of-the-art performances of 97.69% F_1 score for named entity recognition and 89.70% F_1 score for relation extraction.

Keywords: Analysts' Ratings · Financial Data · Information Extraction · Named Entity Recognition · Relation Extraction · Multilingual.

1 Introduction

Stock markets are one of the leading concepts in today's open economy. They can be defined as a collection of exchanges and trades where shares of companies can be bought, sold or issued[1]. These operations are governed by a set of tight rules and regulations, which opens the opportunity for their analysis by experts

[1] Stock Market, www.investopedia.com/terms/s/stockmarket.asp, Accessed: 2023-07-15

in the field, like analysts who aim to predict the price target of the shares of a particular company in the near future based on its financial activities. On the other side, we have investors who invest money in a certain business entity in hopes of making a profit under acceptable risk levels. Investors rely heavily on what experts (analysts) have to say and predict about their company or asset of interest. Investments are typically organized in stock portfolios which balance between expected returns and possible risks. There is an abundant scientific literature regarding stock price prediction, portfolio management, risk assessment, algorithmic trading, etc. Numerous works have explored applications of machine learning for these financial applications, and recently particularly deep learning [4,20]. However, many experts are cautious when applying machine learning algorithms as it has shown mixed performance [4,5].

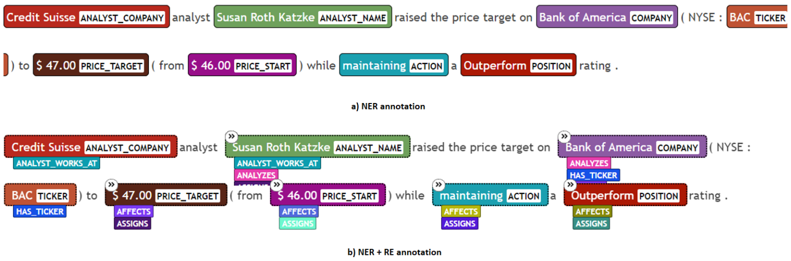
According to [5], when investing in equities investors can base their decisions either on analyst ratings given by human experts or quantitative ratings generated by machine learning. A question arises of whether investors should trust human wisdom more than the advice of machines. Their quantitative ratings are generated using the random forests algorithm, and they employ the human-generated ratings information by analyzing their sentiment. The results reveal that the analysts ratings outperform the quantitative rating, implying that analysts ratings are much more useful for making good decisions. Another study [24], explores a variety of ways for identifying a feature set for stock classification and emphasizes the significance of analyst ratings for bringing valuable human knowledge of the current stock market situation. Analysts provide their knowledge of trading activity statistics derived from historical data and external factors impacting companies' operations. This information is often biased as analysts are pressured to make more optimistic projections due to relations with investment banks [24], but taking into account the number of analysts per stock and the assumption that not all are related to the same banks, we can assume that the ratings variance cancels a significant amount of the bias. Finally, the authors combine features from a technical and fundamental analysis and pose a classification problem where each stock is labeled as buy, hold, or sell.

Keeping in mind the importance of the analyst ratings, we proceed with the process of extracting information out of them, in a form which can be then utilized more easily. Analysts typically share their publicly distributed expertise in a form of a raw unannotated text, containing key information about a company's shares, price targets, and conclusions, usually in a buy-sell-hold form, implying their suggestions on those particular stocks. However, there are dozens of analysts that analyze one company and there are dozens of companies analyzed by one analyst. Hence, it is useful to have a tool that automatically extracts all the information needed from the raw analysts' ratings into an annotated form without manual effort. Therefore, in our study, we utilize information extraction techniques and map the knowledge in the analysts' ratings in order to build a system that can facilitate analyses of companies' performances, improve predictions of stock prices trends and enhance portfolio management.

The problem of information extraction (IE) from analysts' ratings can be divided into two subtasks, Named Entity Recognition (NER) and Relation Extraction (RE). A recent survey of the state-of-the-art methods for NER and RE can be found in [19], while another survey focusing on deep learning methods for named entity recognition can be found in [15]. NER has a long and illustrious history as a tool for financial texts analysis. Its function is to process text in order to identify specific expressions as belonging to a label of interest [18]. For example, the study in [8] identifies entities such as ”invoice sender name”, ”invoice number” and ”invoice date” in business documents, such as invoices, business forms, or emails. In terms of analysts' ratings, there are a few key concepts i.e., entities that need to be retrieved in order to gain information from the raw text. One might be interested in extracting entities including the name of the analyst, the company of interest, and the predicted price target and position as shown in Figure 1 a). In the figure, NER retrieves the information that there is an analyst ”Susan Roth Katzke”, a company ”Bank of America” and a price target ”$ 47.00”, and points to their exact location in the text. Nonetheless, we have no way of knowing if ”Susan Roth Katzke” is analyzing ”Bank of America” or whether the price objective is for the same company. Although in that particular example, there is only one analyst that evaluates one company with a sole price target and their mutual relationship can be taken for granted, in the wilderness of analyst ratings websites, things can get way more complicated and one text can contain information about multiple analysts evaluating multiple companies. Therefore, in order to find the semantic relationship between the entities [30] we need to employ Relation Extraction (RE). RE typically operates on top of NER or any other sequence labeling architecture, although it can be also solved jointly with NER [25,27,28]. However, we choose the first option, as demonstrated in Figure 1 b), and after NER we proceed with annotating relationships between the entities to obtain the semantics of the raw text. Now, it is clear that not only this rating is written by some analyst ”Susan Roth Katzke” and there is a company ”Bank of America”, but it can be also stated that the analyst is analyzing that particular company and she assigns the price target of ”$ 47.00”.

There are numerous other applications of automatic information extraction from financial texts using NER [7,26,3,16], or NER & RE [11,31]. In [22], the reader can find an overview of NER and RE applications in financial texts as well as knowledge graphs construction and analysis. Recent papers have also addressed some other related useful information extraction problems. In [12], the authors have assembled and annotated a corpus of economic and financial news in English language and used it in the context of event extraction, while another study in [29], focused on event extraction from Chinese financial news using automated labeling. Another work in [16], solves a joint problem of opinion extraction and NER using a dataset of financial reviews.

To our knowledge, there are not many recent works in NER and RE using analysts ratings data, and the closest research was presented in [11], where the authors collect and annotate a French corpus with financial data that is not re-


Credit Suisse ANALYST_COMPANY analyst Susan Roth Katzke ANALYST_NAME raised the price target on Bank of America COMPANY (NYSE : BAC TICKER
) to $ 47.00 PRICE_TARGET (from $ 46.00 PRICE_START) while maintaining ACTION a Outperform POSITION rating .
Credit Suisse ANALYST_COMPANY analyst Susan Roth Katzke ANALYST_NAME raised the price target on Bank of America COMPANY (NYSE :
ANALYST_WORKS_AT
ANALYST_WORKS_AT
ANALYZES
ANALYZES
HAS_TICKER
BAC TICKER) to $ 47.00 PRICE_TARGET (from $ 46.00 PRICE_START) while maintaining ACTION a Outperform POSITION rating .
HAS_TICKER
AFFECTS
ASSIGNS
AFFECTS
ASSIGNS
AFFECTS
ASSIGNS
AFFECTS
ASSIGNS

a) NER annotation

b) NER + RE annotation

Fig. 1. a) Named Entity Recognition annotation for one Analyst's Rating; b) Relation Extraction annotation on top of NER entities, explaining mutual semantic relationships between them.

quired to be analyst ratings and use it to train models for extracting entities and their mutual relationships. This study puts an accent on the data collection and preparation for NER and RE. Namely, they collect and manually annotate 130 financial news articles and only perform proof of concept experiments for entities and relations extraction. They base their models on SpaCy v2 [9] and obtained 73.55% F_1-score for NER and 55% F_1-score for relation extraction. We extend this study by collecting and annotating a multilingual corpus with Analyst Ratings in both English and German and exhaustively utilize them for training the proposed models. Furthermore, we switch to SpaCy v3 [10] and employ a newly developed RE component which eliminates the usage of dependency parser for extracting relations, and thus overcomes the gap between precision and recall noted in [11] and obtain a better overall F_1 score. Another research work in [31], addressed the problems of NER and RE jointly using BiGRU with attention in a corpus of manually annotated 3000 financial news articles. However, the authors do not provide many information about the dataset and do not employ a transformer architecture.

The rest of the paper is organized as follows. Section 2 describes the gathering of the dataset used for training. We move on to technical aspects and methodologies for NER and RE and the process of building multilingual language models in German and English in Section 3, and in Section 4, we give the outcomes and the results. Finally, we summarize this research in Section 5.

2 Dataset

The data used in this work was obtained using the API provided by CityFALCON [2] from which we pulled approximately 180000 general financial-related texts. Due to the manually intensive work, we labeled only a few more than 1000 of them. The scraped data also contained a considerable amount of texts not related to analyst ratings, and therefore, we employed a keyword filtering

[2] CITYFALCON, www.cityfalcon.com

strategy to purify the texts. The strategy consisted in manually inspecting common words and phrases occurring in the ratings and discarding all the texts that lacked those words. The annotation process was done using the online annotation tool UBIAI[3] which offers intuitive UI for both NER and RE. The whole process was catalyzed by using pre-annotation strategies including pre-annotation dictionaries and trained models.

The longest rating contained 1088 tokens, whilst the shortest had only 9. On average, the ratings were 79 tokens long and 50% of them were longer than 51. The mean distance between two entities that are part of the same relation was 13 tokens, with the longest distance in the dataset being 260.

Table 1 sums up the statistics for the obtained corpus used for training our proposed NER and RE models. All entities, apart from $POSITION$ and $ACTION$, have descriptive names. $POSITION$ refers to the rating that an analyst gives to certain stocks which might be used as an indicator either to buy, sell or hold the given stock. There are 4 ratings indicating that the analyst believes that the shares should be bought i.e., ”Analyst Buy Rating”, ”Analyst Strong Buy Rating”, ”Analyst Outperform Rating” and ”Analyst Market Perform Rating”. The ”Analyst Hold Rating” indicates that the analyst believes the shares should be held. On the other hand, there is ”Analyst Neutral Rating” where ”Neutral” does not refer to a hold position, but rather a position where the analyst hesitates to share any kind of an opinion. Furthermore, there are 3 ratings indicating that the analyst believes the shares should be sold: ”Analyst Sell Rating”, ”Analyst Strong Sell Rating” and ”Analyst Underperform Rating”. It is important to note that an adjective before the rating describes its intensity, e.g., ”Strong Buy” indicates that the analyst is extremely sure that buying the stocks is a good idea. Other descriptive adjectives of this kind can also be found in the analyst ratings and they all equally apply for sell and hold positions.

In the financial world, it is common that analysts change their mind regarding a given position on a rating after conducting more thorough research or obtaining new information related to the company activities. To denote these changes in ratings from the analysts, the $ACTION$ entity is used. In our study, we use 4 actions as shown in Table 2, although they can appear with different synonyms in the obtained ratings.

3 Methodologies

Sequence labeling problems today are generally approached by using pre-trained Language Models (LMs) as their backbone, whether transformer architectures like $BERT$ [6] and $RoBERTa$ [17], or other contextual embedding models based on RNNs such as $BiLSTMs$ [2]. All of the pre-trained LMs are trained on huge corpora, which makes them as suitable for the financial domain as they are for any other, and allow us to transfer the general knowledge obtained by processing massive amounts of data to the problem at stake, in a procedure known as

[3] UBIAI, https://ubiai.tools

Table 1. Description and distribution of the entities and relations of the dataset. All of the entities have a descriptive name except POSITION and ACTION. Position refers to the buy-sell-hold concept, but it can usually be found in many other forms like ”outperform”, ”market perform”, ”strong buy” etc. Action refers to the change the analyst has made in the position i.e. if we go from positive to negative POSITION (”buy” to ”hold”) we should expect words like ”downgrade” or ”cut” to be the ACTION. The ”*” sign in the relations refers that the entity has priority over the other one, and if both of them are present, only the prioritized one is taken into consideration.

NAMED ENTITY RECOGNITION		
Entity	**Description**	**Num. Instances**
ANALYST_NAME	Name of a person who analyzes stocks	1773
ANALYST_COMPANY	Company employing the analyst	3101
COMPANY	A company that is analyzed	6739
TICKER	Unique identifier on the stock market for each company	1155
PRICE_TARGET	Projected future price of the stocks	2324
PRICE_START	Starting price of the stock before the rating is announced	941
POSITION	Analyst suggestion whether stocks should be bought, sold or held	1669
ACTION	Explains the change in position the analyst has made from past analysis	1039

RELATION EXTRACTION			
Relation	**From**	**To**	**Num. Instances**
ANALYST_WORKS_AT	ANALYST_NAME	ANALYST_COMPANY	715
ANALYZES	ANALYST_NAME*, ANALYST_COMPANY	COMPANY	1612
HAS_TICKER	COMPANY	TICKER	1073
AFFECTS	PRICE_TARGET, PRICE_START, ACTION, POSITION	COMPANY	3979
ASSIGNS	ANALYST_NAME*, ANALYST_COMPANY	PRICE_TARGET, PRICE_START, ACTION, POSITION	3888

Table 2. The entity ACTION refers to the changes made in the POSITION between two analyses. The aggregation of the ACTION entity to upgrade-downgrade-reiterate-initiate and the POSITION entity to buy-sell-hold is made for the sole purposes of explanation. Note that, in the ratings texts, these entities can occur with different synonyms, forms, and additional adjectives for intensity.

Action	Change in Position
Upgrade	Hold→ Buy, Sell→ Hold, Sell → Buy
Downgrade	Buy → Hold, Hold → Sell, Buy → Sell
Reiterate	No Change
Initiation	Initialize Position

transfer learning [21]. The output of the pre-trained models is used as an input of an often simpler classification model for determining the final label for each token. The same concept is also employed in the RE task, such that instead of classifying entities, we are classifying potential relationships between them.

We can think of the embedding model (EM) as the ”central dogma” for NLP, where each token of the text is converted into a pertinent product for the machine i.e. a vector of real numbers. When building a joint model for NER and RE, there are two approaches that might be taken into consideration depending on the exact implementation and position of that EM. Namely, the NER and RE components of the joint model can be trained either as a single LM or they can be divided into two stacked LMs. The former approach brings two options. In the first option, the embedding layer is shared by the two components and is updated in a mutual fashion, which leads to multi-task learning and faster training. However, in this way, we have to use the shared embedding for both NER and RE in the future, despite the fact that one model may be superior for NER and another for RE. The second option is to train the two components together, but with separate embedding layers, which will make the training process slower, but on the other hand, it cancels the problem with the performance compatibility of the LMs for NER and RE.

The second approach is to train the two components separately and only connect them during the inference stage. As a result, we can conduct more granular and targeted experiments for NER and RE, while using less GPU resources. However, the time complexity increases, but we gain on the simplicity of the overall architecture. All of the following proposed models in this research follow this particular approach. Furthermore, we follow the work presented in [23], where they discuss two alternatives for NER. The first one is to fine-tune the transformer layer on the NER task and only use a simple linear layer for the token classification, and the second is to directly use the embedding from the pre-trained LMs and employ a more complex classification layer. They arrive to the conclusion that the fine-tuning strategy beats the latter, so we follow their lead, but instead of utilizing only a single linear layer as the model's head, we also utilize very simple classifiers including RNN cells.

3.1 Named Entity Recognition

The classification layer used to categorize the tokens into one of the entities is simple, which is an advantage of employing and fine-tuning sophisticated embedding models. The dataset is randomly split into two parts, 90% for training and 10% for testing, and those remain static throughout the fine-tuning process of all models in order to provide a fair comparison. The data was provided in an IOB format (Inside-Outside-Beginning), and it was converted to a $DocBin$ file when training with SpaCy. IOB is considered the standard data representation for NER due to the fact that it introduces 3 different types of tags to denote whether a token is at the beginning, inside, or outside the entity. Let us consider the ANALYST_NAME 3 token entity ”Susan Roth Katzke”. The token ”Susan” will be labeled as $B - ANALYST\_NAME$, denoting the beginning of the entity, and the next two tokens, ”Roth” and ”Katzke” will get the $I - ANALYST\_NAME$ which stands for inside the entity. If a token is not part of any named entity, then it is marked as O which stands for outside any entity.

For the NER task, we compare two powerful NLP frameworks, SpaCy [10] and FLAIR [1]. We make use of SpaCy's CLI (Command Line Interface) which offers commands for initializing and training pipelines. The pipeline used to perform a NER task consists of an embedding model (transformer) and a NER classifier that consists of a single linear layer as an LM Head to the transformer. Most of SpaCy's proposed hyperparameters were adopted without changes because they have been demonstrated to be quite successful. Additionally, Adam [13] with *warmup* steps was used as an optimizer with an initial learning rate of $5e - 5$, which allows the tuning of more sensitive parts in the model like the attention mechanism. In our work, we used the case-sensitive variants of five different transformer architectures as an embedding component in the SpaCy pipeline: $bert-base-cased$, $distilbert-base-cased$, $roberta-base$, $xlm-roberta-base$ and $albert-base-v2$.

We used the same pipeline architecture in the FLAIR framework, with the only difference being the additional RNN layer between the embedding model and the linear decoder. According to prior FLAIR research and experiments, adding a single LSTM layer to an IE task like NER has proven to be quite effective. Furthermore, we utilize stacked embeddings [2] which allows us to use a few embedding models at the same time concatenated as a single vector. The initial learning rate for the models was set to 0.1 with an annealing factor of 0.5 on every two epochs without improvement. These values were taken based on similar prior experiments presented in [14] in order to avoid the expensive cost of hyperparameter optimization. We used FLAIR to explore static embedding types such as GloVe and stacked embeddings combining FLAIR forward and backward LMs with either GloVe or BERT embeddings.

3.2 Relation Extraction

The embedding component of the RE model is identical to that of the NER model. In the case of NER, after embedding the tokens, each obtained vector is fed as an input to a classification layer. However, because one relation is represented by two entities, and entities can have numerous tokens, a few more steps are required for extracting relations. The first step of relation extraction is to create a matrix E that contains the vectors of each entity in a document:

$$E = \begin{bmatrix} e_{111} & e_{112} & e_{113} & \dots & e_{11n} \\ e_{121} & e_{122} & e_{123} & \dots & e_{12n} \\ e_{211} & e_{212} & e_{213} & \dots & e_{21n} \\ \dots & \dots & \dots & \dots & \dots \\ e_{m11} & e_{m12} & e_{m13} & \dots & e_{m1n} \end{bmatrix},$$

s.t. n is the dimension of the embedding space, and m is the number of entities in the document. Each entity can contain multiple tokens, so in e_{121}, the first 1 denotes entity 1 in the document, 2 denotes the second token in the entity, and the second 1 is the index of a single value from that embedding. After defining matrix E, we deal with the multi-token named entities and use the average pooling operator in order to obtain a single vector per entity:

$$E^{'} = \begin{bmatrix} E_{11} & E_{12} & E_{13} & \dots & E_{1n} \\ E_{21} & E_{22} & E_{23} & \dots & E_{2n} \\ E_{31} & E_{32} & E_{33} & \dots & E_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ E_{m1} & E_{m2} & E_{m3} & \dots & E_{mn} \end{bmatrix}.$$

After obtaining the matrix E', pairs of entities are mutually combined, representing a potential relation:

$$R = \begin{bmatrix} E_{11} & ... & E_{1n} & E_{21} & ... & E_{2n} \\ E_{21} & ... & E_{2n} & E_{11} & ... & E_{1n} \\ E_{11} & ... & E_{1n} & E_{31} & ... & E_{3n} \\ ... & ... & ... & ... & ... & ... \\ E_{(m-1)1} & ... & E_{(m-1)n} & E_{m1} & ... & E_{mn} \end{bmatrix}.$$

Each of the rows in matrix R is a vector representation for a possible relation in a document, and as such, it is fed as an input of a LM. The output of that model is a numerical value denoting the probability of the given entity pair (relation) belonging to one of the relation classes: $ANALYST\_WORKS\_AT$, $ANALYZES$, HAS_TICKER, $AFFECTS$ and $ASSIGNS$. Both, SpaCy and FLAIR follow this idea with minor differences and implement it in different Deep Learning libraries, i.e., Thinc and PyTorch respectively. Again, we utilize the transformer architectures with SpaCy, and FLAIR embeddings and GloVe with the FLAIR framework.

Most analyst ratings are written according to some unofficial criteria, and they all have a similar structure, regardless of the analyst or the analyst company they come from. Usually, they are written in a very concise way and are not prone to ambiguity. As a consequence, we have a well-defined text, such that all the entities that are in a mutual relation are close to one another. Our proximity analysis has shown that 95% of the entities that share a relation are within a window of 40-token radius. Therefore, we discarded all the relations that are not within the predefined window and trained only on entity pairs that are considered to be close enough. This also goes hand in hand with the fact, that both architectures, transformer and biLSTMs, have reduced accuracy when predicting long text sequences.

3.3 Multilingual Models

In NLP, multilingualism refers to the idea of training a single model using data from multiple languages. These models can subsequently be fine-tuned on a multilingual corpus or a monolingual corpus and used for other languages that are similar i.e. from the same language family, as shown in [14], where a multilingual NER model in Macedonian is trained and later tested on Serbian corpus with some fairly promising results. Following the work presented there, we also test the $xlm - roberta - base$ model trained with analyst ratings in English, on a German corpus containing 100 ratings. The model performed better than a random baseline, achieving 44.8% F_1 for NER and 32.61% for RE, however, the results were far from what was achieved for the English ratings. Although both languages come from the Germanic family, they have some fairly different grammar and syntax. Let us look at the verb ”zurückstufen” for example. It translates to English ”downgrade” and denotes the entity ”ACTION” in our use case. In German syntax, this verb splits into two parts, such that ”stufen” stays in the second position, whereas ”zurück” goes last. These parts cannot be annotated together which forms a kind of ambiguity compared to the English

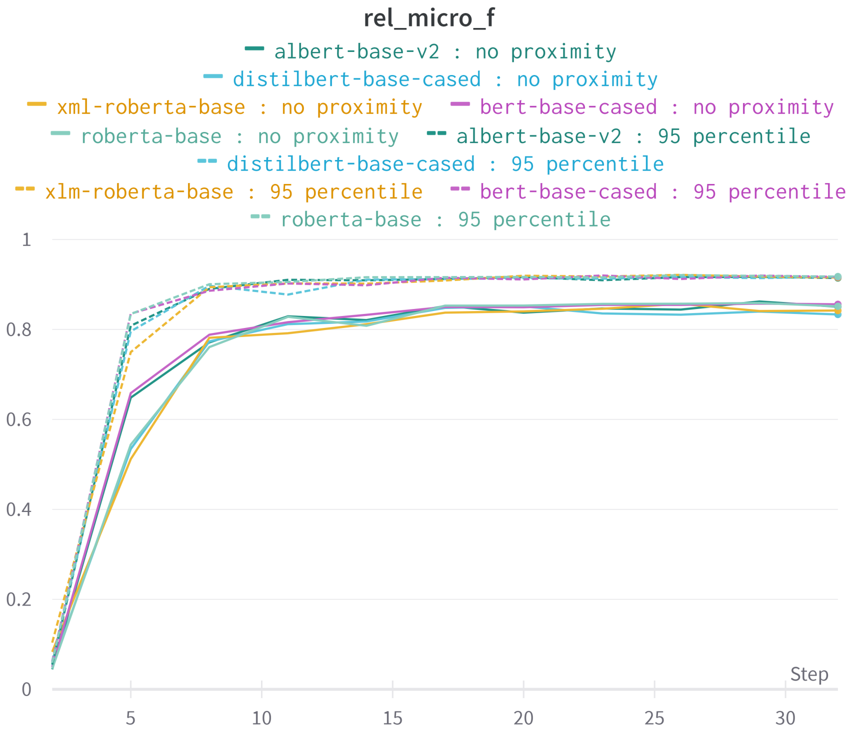


Fig. 2. Difference in the F_1-score (y-axis) for relation extraction between the transformers trained with proximity (95 percentile) and without proximity.

corpus of analyst ratings. We further extended this idea by labeling 100 more analyst ratings in German in order to infiltrate them into the training data and define the special rules for data annotation in German.

The multilingual corpus was fed to 3 transformer models, $xlm - roberta - base$, which is the multilingual version of $roberta - base$, pre-trained on 2.5 TB of data in 100 different languages. We also utilized $bert - base - multilingual - cased$ and $distilbert - base - multilingual - cased$ pre-trained on 104 different languages. On the other hand, we also utilize FLAIR forward and backward multilingual embedding models, pre-trained on more than 300 languages.

4 Results and discussion

All proposed models are evaluated with three different metrics: precision, recall, and F_1-score, calculated as the harmonic mean of the former two metrics. Just like the training phase, the evaluation is also performed separately for the NER and RE subtasks. However, considering the fact that RE stands on top of NER, the evaluation metrics for RE are obtained using the golden labels from our NER annotations. The test sets make up 10% of the total dataset and include analyst ratings not encountered during the training phase.

4.1 Named Entity Recognition

The evaluation ratings were static for each proposed NER model in order to obtain relevant comparisons. In Table 3, evaluation results for the NER task trained with SpaCy are presented.

The results demonstrate that NER with SpaCy achieved state-of-the-art performance with an almost perfect F_1-score, which is not surprising given that the entities are not ambiguous, i.e. when analysts talk about downgrading a company, they indeed mean it. The performance of each model individually approves the aforementioned transformer analysis. In this use-case, RoBERTa slightly outperformed BERT ($F_1 = -0.0043$) and XLM RoBERTa ($F_1 = -0.0058$), demonstrating that using a mixed corpus for pre-training did not degrade the model's overall performance for a significant amount. Although DistilBERT achieves the fourth best performance ($F_1 = -0.0072$), this model is on top of the list when it comes to speed and space complexity, and considering real-world applications where system performances matter, it can be considered even as the best candidate. ALBERT achieved the worst results on the analyst ratings dataset. Although having a descent recall, it struggled with precision, especially for the $PRICE\_START$ entity.

Table 3. Evaluation of the named entity recognition task using transformer architectures as embedding models with SpaCy.

	NER - SpaCy		
Model	Precision	Recall	F_1-score
roberta-base	**0.9797**	**0.9740**	**0.9769**
bert-base-cased	0.9712	**0.9740**	0.9726
xlm-roberta-base	0.9721	0.9702	0.9711
distilbert-base-cased	0.9720	0.9673	0.9697
albert-base-v2	0.8925	0.9568	0.9235

The next set of results for NER come from the FLAIR experiments, presented in Table 4. Even though GloVe is a non-contextual concept for embeddings, based on the co-occurrence matrix of the tokens, it achieves F_1-score almost as high as the other transformers and even outperforms ALBERT. Both triplets of stacked embeddings achieve results comparable with $roberta - base$. Surprisingly, the FLAIR embeddings combined with GloVe slightly outperform the combination with BERT. Due to the fact that W&B is not integrated with FLAIR, we omit evaluating the system performances of these models.

4.2 Relation Extraction

Relation extraction is a newer and less researched task in the information extraction field and it is yet to acquire the same level of accuracy as NER. Although pre-trained LMs like transformers are also employed as the backbone of RE, it seems that the problem of detecting semantic relationships between entities is more complex than detecting the entities. However, apart from the fact that the RE results are worse than the results obtained for the NER task, we still record high scores for RE, as it can be seen in Table 5.

Table 4. Evaluation of the named entity recognition task using static and stacked embedding models with FLAIR (”F” and ”B” stand for FLAIR forward and FLAIR backward models).

	NER - FLAIR		
Model	Precision	Recall	F_1-score
GloVe	0.9754	0.9501	0.9626
FLAIR F + B + GloVe	**0.9796**	0.9674	**0.9734**
FLAIR F + B + BERT	0.9713	**0.9731**	0.9722

The transformers results presented in Table 5 are based only on the proximity analysis, since it yields 4-6% better F_1-scores than the plain models, as seen on Figure 2. RoBERTa still wins the RE task, however, the relative difference in performance between GloVe for NER (Table 4) and GloVe for RE (Table 5) is inevitable to notice. The reason for this difference is the non-contextual nature of GloVe which manages to extract the entities, but it is not powerful enough to extract the semantic relations between them.

Table 5. Evaluation of the RE task with SpaCy (transformers) and FLAIR (GloVe, FLAIR embeddings).

	Relation Extraction		
Model	Precision	Recall	F_1-score
roberta-base	0.9249	**0.8707**	**0.8970**
bert-base-cased	0.9249	0.8595	0.8910
xlm-roberta-base	0.9230	0.8586	0.8896
distilbert-base-cased	**0.9325**	0.8484	0.8885
albert-base-v2	0.9160	0.8521	0.8829
GloVe	0.4673	0.3800	0.4191
FLAIR F + B	0.7740	0.7726	0.7733

4.3 Multilingual Models

Even though English and German are related, using just multilingual models and training on English corpora does not yield the desired results for German.

As a result, infiltrating a portion of the German corpus into the training set with English data is critical. Thus, although the German dataset consisted of only 200 ratings, compared to more than 1000 ratings of the English dataset, when the multilingual models were combined with mixed data, we were able to generate metrics for German that were identical to the English analyst ratings. Table 6 gives an overview of the results for the multilingual models on a German test corpus containing 100 analyst ratings.

Table 6. Evaluation of the multilingual named entity recognition models.

	Multilingual NER		
Model	Precision	Recall	F_1-score
bert-base-multi-cased	0.9564	0.9581	0.9572
xlm-roberta-base	**0.9701**	**0.9633**	**0.9667**
distilbert-base-multi-cased	0.9532	0.9616	0.9574
FLAIR Multi F + B	0.9537	0.9354	0.9445

In Table 7, the results from the multilingual RE models are presented. It is noticeable that the difference between the monolingual and multilingual RE is greater than the monolingual and multilingual NER. We can conclude that more German data are needed in order to obtain identical results for more complex problems like RE.

Table 7. Evaluation of the multilingual relation extraction models.

	Multilingual RE		
Model	Precision	Recall	F_1-score
bert-base-multi-cased	0.7487	**0.9051**	**0.8195**
xlm-roberta-base	0.7316	0.8797	0.7989
distilbert-base-multi-cased	0.7791	0.8038	0.7913
FLAIR Multi F + B	**0.8471**	0.7776	0.8109

We can notice that instead of RoBERTa, the highest F_1-score comes from BERT, followed by the multilingual FLAIR embeddings. It is also important to mention that FLAIR records much higher precision than the transformer models and even better scores than the monolingual task.

5 Conclusion

In this research paper we go through various points of the employment of analysts ratings in stocks analysis. After perceiving their importance, we proceed toward building an information extraction pipeline consisting of extracting entities (NER) and extracting relations (RE). For that point, more than 1000 analysts ratings in English and 200 ratings in German were manually annotated, forming the first such annotated dataset to the best of our knowledge.

We compare two different NLP frameworks, SpaCy and FLAIR, and explore a few different word embedding possibilities, including transformers, Bi-LSTM-based embeddings, and GloVe in order to maximize the results for both subtasks. Our proposed models obtained state-of-the-art results both for NER (97.69% F_1) and RE (89.70%). Furthermore, we explore the system performances of the models and offer a pipeline with an inference time, fast enough for production.

We rounded up this study by examining multilingual models and combing English and German corpora with analyst ratings. Although we had access to only 100 German ratings for training and 100 for testing, the scores for the German ratings were brought extremely close to the ones for English with only 1.02% difference in F_1-score for NER and 7.75% for RE.

Acknowledgement

We are thankful to CityFALCON for providing us the data and for their collaboration and support. This work was partially funded by the Faculty of Computer Science and Engineering at the Ss. Cyril and Methodius University in Skopje.

References

1. Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., Vollgraf, R.: Flair: An easy-to-use framework for state-of-the-art nlp. In: NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations). pp. 54–59 (2019)
2. Akbik, A., Blythe, D., Vollgraf, R.: Contextual string embeddings for sequence labeling. In: COLING 2018, 27th International Conference on Computational Linguistics. pp. 1638–1649 (2018)
3. Alvarado, J.C.S., Verspoor, K., Baldwin, T.: Domain adaption of named entity recognition to support credit risk assessment. In: Proceedings of the Australasian Language Technology Association Workshop 2015. pp. 84–90 (2015)
4. Bartram, S.M., Branke, J., De Rossi, G., Motahari, M.: Machine learning for active portfolio management. The Journal of Financial Data Science **3**(3), 9–30 (2021)
5. Cheng, S., Lu, R., Zhang, X.: What should investors care about? mutual fund ratings by analysts vs. machine learning technique. Machine Learning Technique (August 14, 2021). ADB-IGF Special Working Paper Series "Fintech to Enable Development, Investment, Financial Inclusion, and Sustainability (2021)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805 (2018)
7. Farmakiotou, D., Karkaletsis, V., Koutsias, J., Sigletos, G., Spyropoulos, C.D., Stamatopoulos, P.: Rule-based named entity recognition for greek financial texts. In: Proceedings of the Workshop on Computational lexicography and Multimedia Dictionaries (COMLEX 2000). pp. 75–78 (2000)
8. Francis, S., Van Landeghem, J., Moens, M.F.: Transfer learning for named entity recognition in financial and biomedical documents. Information **10**(8), 248 (2019)

9. Honnibal, M., Montani, I.: spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. Unpublished software application **7**(1), 411–420 (2017)
10. Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A.: spaCy: Industrial-strength Natural Language Processing in Python. Unpublished software application (2020). https://doi.org/10.5281/zenodo.1212303
11. Jabbari, A., Sauvage, O., Zeine, H., Chergui, H.: A french corpus and annotation schema for named entity recognition and relation extraction of financial news. In: Proceedings of The 12th Language Resources and Evaluation Conference. pp. 2293–2299 (2020)
12. Jacobs, G., Hoste, V.: Sentivent: enabling supervised information extraction of company-specific events in economic and financial news. Language Resources and Evaluation pp. 1–33 (2021)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
14. Krstev, I., Fisnik, D., Gramatikov, S., Mirchev, M., Mishkovski, I.: Named entity recognition for macedonian language. repository.ukim.mk (2021)
15. Li, J., Sun, A., Han, J., Li, C.: A survey on deep learning for named entity recognition. IEEE Transactions on Knowledge and Data Engineering **34**(1), 50–70 (2020)
16. Liao, J., Shi, H.: Research on joint extraction model of financial product opinion and entities based on roberta. Electronics **11**(9) (2022)
17. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv:1907.11692 (2019)
18. Mikheev, A., Moens, M., Grover, C.: Named entity recognition without gazetteers. In: Ninth Conference of the European Chapter of the Association for Computational Linguistics. pp. 1–8 (1999)
19. Nasar, Z., Jaffry, S.W., Malik, M.K.: Named entity recognition and relation extraction: State-of-the-art. ACM Computing Surveys (CSUR) **54**(1), 1–39 (2021)
20. Ozbayoglu, A.M., Gudelek, M.U., Sezer, O.B.: Deep learning for financial applications: A survey. Applied Soft Computing **93**, 106384 (2020)
21. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering **22**(10), 1345–1359 (2009)
22. Repke, T., Krestel, R.: Extraction and representation of financial entities from text. In: Data Science for Economics and Finance, pp. 241–263. Springer, Cham (2021)
23. Schweter, S., Akbik, A.: Flert: Document-level features for named entity recognition. arXiv:2011.06993 (2020)
24. Singh, J., Khushi, M.: Feature learning for stock price prediction shows a significant role of analyst rating. Applied System Innovation **4**(1), 17 (2021)
25. Sun, C., Gong, Y., Wu, Y., Gong, M., Jiang, D., Lan, M., Sun, S., Duan, N.: Joint type inference on entities and relations via graph convolutional networks. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 1361–1370 (2019)
26. Wang, S., Xu, R., Liu, B., Gui, L., Zhou, Y.: Financial named entity recognition based on conditional random fields and information entropy. In: 2014 International Conference on Machine Learning and Cybernetics. vol. 2, pp. 838–843. IEEE (2014)
27. Wang, Y., Yu, B., Zhang, Y., Liu, T., Zhu, H., Sun, L.: TPLinker: Single-stage joint extraction of entities and relations through token pair linking. In: Proceedings of the 28th International Conference on Computational Linguistics. pp. 1572–1582. International Committee on Computational Linguistics, Barcelona, Spain

(Online) (Dec 2020). https://doi.org/10.18653/v1/2020.coling-main.138, https://aclanthology.org/2020.coling-main.138

28. Wei, Z., Su, J., Wang, Y., Tian, Y., Chang, Y.: A novel cascade binary tagging framework for relational triple extraction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 1476–1488. Association for Computational Linguistics, Online (Jul 2020). https://doi.org/10.18653/v1/2020.acl-main.136, https://aclanthology.org/2020.acl-main.136
29. Yang, H., Chen, Y., Liu, K., Xiao, Y., Zhao, J.: Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data. In: Proceedings of ACL 2018, System Demonstrations. pp. 50–55 (2018)
30. Zhou, G., Su, J., Zhang, J., Zhang, M.: Exploring various knowledge in relation extraction. In: Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05). pp. 427–434 (2005)
31. Zhou, Z., Zhang, H.: Research on entity relationship extraction in financial and economic field based on deep learning. In: 2018 IEEE 4th International Conference on Computer and Communications (ICCC). pp. 2430–2435. IEEE (2018)