Text Representation Enrichment Utilizing Graph based Approaches: Stock Market Technical Analysis Case Study

Sara Salamat∗
Nima Tavassoli∗
Behnam Sabeti
Reza Fahmi
{sara,nima,behnam,reza}@eveince.com
Eveince
Berlin, Germany

ABSTRACT

Graph neural networks (GNNs) have been utilized for various natural language processing (NLP) tasks lately. The ability to encode corpus-wide features in graph representation made GNN models popular in various tasks such as document classification. One major shortcoming of such models is that they mainly work on homogeneous graphs, while representing text datasets as graphs requires several node types which leads to a heterogeneous schema. In this paper, we propose a transductive hybrid approach composed of an unsupervised node representation learning model followed by a node classification/Edge prediction model. The proposed model is capable of processing heterogeneous graphs to produce unified node embeddings which are then utilized for node classification or link prediction as the downstream task. The proposed model is developed to classify stock market technical analysis reports, which to our knowledge is the first work in this domain. Experiments, which are carried away using a constructed dataset, demonstrate the ability of the model in embedding extraction and the downstream tasks.

1 INTRODUCTION

The way we understand and process natural language mainly depends on the way it is represented. While representing language as a bag of tokens and sequence of tokens are largely used in the NLP community, graph representation is another approach for structuring texts, as well as utilizing relational information between different text elements. Representing text as a graph has a long history as several specialized text graphs, including dependency graphs [39], constituency graphs [23], lexical networks [25], knowledge graphs [37], etc. has been created in this field. Moreover, it is also possible to represent entities with different hierarchies such as document, passage, sentence, and word in one graph. Because of this property, many NLP tasks require heterogeneous graphs to model different entities and relations of the problem. However, GNN models, as one of the most powerful frameworks of analyzing graphs, such as GCN[16], GAT[31], and GraphSAGE[13] are mostly designed for homogeneous conditions, and therefore cannot fit in these problems directly.

We identify stock market technical analysis reports as a text dataset that provides great analytical opportunities in financial markets. Hundreds of these analyses in the form of text documents are published in TradingView1 on a daily basis. Market analysts and investors share their analysis, insights, and predictions within the platform. They can also assign a tag to their posts from a predefined set (Long, Short, Education) to indicate the overall status of that analysis. Table 1 shows a sample of a post in our dataset. As a novel work in this domain, we strive to automatically extract insights and information from these reports with the help of graph representation learning. In particular, this paper focuses on a specific version of that vision that is the problem of classifying technical analysis reports in the cryptocurrencies market. Classifying all documents into the Long, Short, and Education classes helps us to perceive what position analysts and investors generally take on a particular symbol in a given time window. In order to classify properly, we use authors’ tags as the supervision of the classification task and train a classifier based on those labels.

To exploit richer relations among documents, we follow the graph representation approach for the proposed document classification problem and construct a graph based on the given input text. There are multiple methods for static graph construction during preprocessing documents [35]. We take a hybrid approach, in which we capture both co-occurrence and similarity relations between words and documents. We also cover the loss of the text sequential information to some extent, by creating n-grams during the preprocessing steps and using them in graph modelling.

Our base graph structure is solely constructed on the information

1Both authors contributed equally to this research.

1Tradingview.com
contained in the technical reports: documents and words, but formu-
lating this classification problem in a graph-based structure allows
us to represent our desired entities that are related to our specific
problem as part of the graph structure. We can take price patterns
as an example, which are formations that appear on stock price
charts and are one of the key components of technical analysis in
financial markets. If we consider each price pattern as a separate
node type, it is possible to connect each one to its implication de-
defined by the tag node type, as well as the document containing that
pattern. Here, we have used an external knowledge base to enrich
our document dataset. As a result, we can incorporate different
domain knowledge into the current structure, and capture richer
relationships among different elements.

In this work, we introduce a novel classification task of stock mar-
tetrical analysis documents and develop a hybrid transductive
graph-based solution. In the proposed approach, first, we store
TradingView’s posts published in the Ideas section and preprocess
all documents. Then, we define word and sentence representations
by language models to construct our graph. We also add price
patterns, corpus topics, and post labels to the graph as separate
node types. After constructing the graph, node embeddings are
trained in an unsupervised manner, and we update the graph nodes
with new representations. This step can be considered optional.
Finally, we train both node classification on documents and link
prediction on the edges between documents and tags in order to
estimate document labels. We conduct various experiments to ex-
amine our proposed method and compare it with state-of-the-art
baselines in different settings. Experimental results show that our
best graph-based approach achieves 89% F1-score, while the top
baseline method (Fine-tuned BERT) has the F1-score 76%.
In short, we summarize the main contributions as follows:

• Proposing a novel heterogeneous graph construction method
  for representing technical analysis documents.
• Proposing a graph neural network model for technical anal-
ysis reports classification.
• Providing experimental results showing that our method
  outperforms several state-of-the-art document classification
  baselines.
• Introducing the “Stock Market Technical Analysis Reports
  Classification” task, which to our knowledge is a novel work
  in related domains.
• Presenting a dataset of technical analysis reports in the cryp-
tocurrencies market that enables further research in this field.

The remaining parts of this paper are structured as follows. Sec-
tion 2 further discusses the stock market technical analysis reports
and price patterns in financial markets. Following that, in Section
3 our graph construction approach is described and our hybrid pro-
posed model is explained in detail. Section 4 introduces the baseline
methods, our constructed dataset, and experimental results. Finally,
a categorization of graph-based approaches for text classification
is presented as related works in Section 5 and the paper is concludeed
in Section 6.

2 STOCK MARKET TECHNICAL ANALYSIS
REPORT

Predicting market behaviour from financial text data has been under
extensive research in recent years. Analyzing financial texts every-
day and extracting insights from them is a heavy task; therefore,
researchers have been trying to leverage machine learning methods
to derive insights. Research in this area has been mostly done on
sentiment analysis of social media data to find out whether people
are talking about the market in a positive, negative or neutral way
[2, 4, 6, 7, 9, 14, 22, 42]. These approaches miss an important point.
Most of the times a rise or fall in market prices leads to social media
posts and people’s reactions. It is rare to get insights about market’s
future trend from people’s emotions in their posts. To address this
matter, technical analysis reports are used in this study. Technical
analysts use price movements in price charts to decide the future
trends of the price [17]. They use different indicators and patterns
for their analysis. Technical analysis reports are the outcome of
technical analysts’ investigation of the price chart. They talk about
the behaviour of the price chart and how they think the behaviour
affects the future trend of the chart. These reports carry much more
information about the movements of the market. Therefore, it is
more likely that technical reports are correlated with price move-
ments because writers are usually traders with expertise in market
analysis.

2.1 Research Target

Market analysts usually talk about their analysis of the market
behaviour and how they think market will behave afterwards in
their writings. In this study, we aim to process their analysis and
find out what they expect to see in the chart. However, technical
analysis can be prone to error as not every analysis is correct and
not everyone is expert enough to predict future behaviour correctly.
By looking at what the majority of the people are seeing in the
price chart and implications in the market, it is possible to derive
insights from their implications and finally predict the possible next
move of the market from the technical analysts’ point of view.

2.2 Price Pattern

Price patterns are configurations in price charts that are often used
by technical analysts for anticipating the direction of the market
prices. Price patterns have different shapes and are often identified
by trendlines and curves [28]. Patterns can be categorized into two
groups of reversal and continuation patterns. Reversal patterns state
a change in the direction of the market and continuation patterns
state that the trend remains the same. In this work, most popular
price patterns among technical analysts were used as graph nodes.
In stock trading, having a long position means that a rise is expected
in the price of the market as opposed to a short position which
means a fall is expected in the price. Price patterns usually signal a
rise or fall afterwards so their implications in the market is utilized
in this research in graph modeling. Table 2 shows some examples
of patterns, their description and implications in the market.

3 PROPOSED APPROACH

In this section, our approach for financial text analysis is explored.
First, the graph structure is explained and then graph processing
We use a fixed-size sliding window on all documents to calculate the number of documents that contain that word. The document and inverse document frequency is the inverse of the set. Term frequency is the number of times the word appears in frequency-inverse document frequency (TF-IDF) \[36\]. TF-IDF evaluates how relevant a word is to a document in the entire document set. Term frequency is the number of times the word appears in the document and inverse document frequency is the inverse of the number of documents that contain that word.

We use a fixed-size sliding window on all documents to calculate point-wise mutual information (PMI) between every two words within the window. Then, we connect words with positive PMI values together and assign the derived PMI as the weight of that connection. The PMI value of a word pair \(w_1, w_2\) is computed as

\[
PMI(w_1, w_2) = \log \frac{p(w_1, w_2)}{p(w_1)p(w_2)}
\]

(1)

\[
p(w_1, w_2) = \frac{W(w_1, w_2)}{#W}
\]

(2)

\[
p(w_1) = \frac{#W(w_1)}{#W}
\]

(3)

Where \(#W(w_1)\) is the number of sliding windows in a corpus that contain word \(w_1\), \(#W(w_1, w_2)\) is the number of sliding windows that contain both \(w_1\) and \(w_2\), and \(#W\) is the total number of sliding windows. In this approach, we have used both co-occurrence and similarity relations between words and documents. Each of those methods only considers one specific type of relation and has limitations in representing others.

To utilize high-level semantic information of documents, we connect topic node \(t\), resulted from the topic modeling, to document, word, and price pattern nodes with the relations shown in the table 3. As aforementioned, we introduce the price pattern node type that is connected to the documents containing that pattern. We also connect patterns that have known implications in the technical analysis literature to the position nodes. Figure 2 illustrates a sample of nodes and their relations in our graph. Table 3 summarizes edges between different node types.

### 3.2 Heterogeneous Graph Modeling

As we are dealing with a heterogeneous structure, we design a model that is capable of processing these kinds of graphs. In this section, we describe the components of our proposed model.

#### 3.2.1 HinSAGE

GraphSAGE [13] is a framework for inductive representation learning on large graphs. GraphSAGE is used to generate low-dimensional vector representations for nodes, and is capable of learning an embedding function that generalizes to unseen nodes, without requiring a re-training procedure. The key idea of this approach is training a set of aggregator functions that learn to aggregate feature information from a node’s neighborhood. We extend the GraphSAGE model to be able to aggregate feature vectors from different node types in heterogeneous graphs. Our work is inspired by the HinSAGE model introduced by the StellarGraph library [8]. Algorithm 1 describes HinSAGE in the case where entire graph \(G(V, E)\) and features for all nodes are provided as input. First, all node embeddings are initialized to the node features. Then, at each iteration (search depth), each node aggregates the representations of the nodes in its one-hop neighborhood via edges of type \(r\), and concatenates its current representation \(h^{k-1}_v\) with the aggregated neighborhood vector, \(\mathbf{h}_N^{k}(v)\). As we can observe in the algorithm, there are separate neighborhood weight matrices \(W_{self}\) for each relation between two node types. There are also separate self-feature matrices \(W_{self}\) for every node type. HinSAGE finally passes the concatenated vector through a neural network layer with non-linear activation function \(\sigma\) to update the node embedding.

| Name                  | Description                                | Implication |
|-----------------------|--------------------------------------------|-------------|
| Head and shoulders    | A baseline with three peaks where the middle is the highest. | Short       |
| Double top flag       | Two highs, looking like M                  | Short       |
| Cup handle Bearish    | Two convex arcs which three top points touch a resistance level | Long        |
| flag                  | Characterized by parallel trendlines over the consolidation area-lines have upward sloping | Short       |

**Table 3: Description of edges of the graph**

| Source          | Destination     | Relation                                |
|-----------------|-----------------|-----------------------------------------|
| Word            | Word            | PMI                                     |
| Document        | Word            | TF-IDF                                  |
| Word            | Topic           | \(P(w|t)\)                                |
| Document        | Topic           | \(P(t|d)\)                                |
| Document        | Price Pattern   | binary                                  |
| Price Pattern   | Topic           | \(P(t|pp)\)                               |
| Document        | Position        | User-generated label (Link Prediction problem) |
| Price Pattern   | Position        | Price pattern implication (Link Prediction problem) |

**Table 2: Price patterns**

| Name                  | Description                                | Implication |
|-----------------------|--------------------------------------------|-------------|

**Source** | **Destination** | **Relation** |
|------------|-----------------|--------------|
| Word       | Word            | PMI          |
| Document   | Word            | TF-IDF       |
| Word       | Topic           | \(P(w|t)\)   |
| Document   | Topic           | \(P(t|d)\)   |
| Document   | Price Pattern   | binary       |
| Price Pattern | Topic         | \(P(t|pp)\)   |
| Document   | Position        | User-generated label (Link Prediction problem) |
| Price Pattern | Position     | Price pattern implication (Link Prediction problem) |
Algorithm 1 HInSAGE Algorithm

Input: Graph $G(V, E)$; input features $\{x_v, \forall v \in V\}$; edge types $\tau \in \{1, \ldots, R\}$; depth $K$; weight matrices $W^k, \forall k \in \{1, \ldots, K\}$; non-linearity $\sigma$; neighborhood function $N: v \rightarrow 2^V$

Output: Vector representations $z_v$ for all $v \in V$

1: $h^0_v \leftarrow x_v, \forall v \in V$
2: for $k = 1 \ldots K$ do
3: for $v \in V$ do
4: for $r = 1 \ldots R$ do
5: $h^k_v \leftarrow \frac{1}{|N_v(\tau)|} \sum_{u \in N_v(\tau)} \mathcal{D}_P[h_u^{k-1}]$
6: end for
7: $h^k_v \leftarrow \sigma[\text{CONCAT}[W^k_\tau \cdot \text{self} \mathcal{D}_P[h_u^{k-1}], W^k_\text{neigh} h_N^k, \ldots, W^k_\text{neigh} h_{N_{\text{neigh}}(\tau)} + b]]$
8: end for
9: $h^k_v \leftarrow \frac{\|h^k_v\|_2}{\|h^k_v\|_2}, \forall v \in V$
10: end for
11: $z_v \leftarrow h^k_v, \forall v \in V$

In order to learn the weights of the aggregator functions, GraphSAGE applies a differentiable, graph-based loss function:

$$J_G(\theta_G) = -\log(\sigma(z^T_\theta v)) - \mathbb{E}_{\gamma_n} \log(\sigma(-z^T_\theta \gamma_n)), \quad (4)$$

where node $v$ is a neighbor of node $u$, node $\gamma_n$ is a distant node to node $v$ and is sampled from a negative sampling distribution $\gamma_n$, and $Q$ is the number of negative samples. This loss function enforces close nodes to have similar representations and distant nodes to have dissimilar representations.

3.2.2 DGI. Deep Graph Infomax (DGI) [32] is an unsupervised node representation learning approach for graph-structured data. DGI’s core idea is learning to distinguish between the original graph and the corrupted one that derives from a corruption procedure. Given the true input graph $G$, they change it into the mutated version $H = C(G)$ by randomly shuffling the node features among nodes. $H$ has the same edges as $G$, but the features associated with each node differ.

The model consists of an encoder that takes an input graph and computes an embedding vector for each node. It is typically based on well-known graph machine learning models such as GCN or GraphSAGE. The weights of the encoder are trained according to the distinguishing step. Thus, DGI learns to discriminate between nodes that have sensible connections and nodes that have unexpected connections. After training, the encoder can be used independently to compute node embedding vectors directly. We use DGI to obtain node embeddings for the downstream tasks as shown in the figure 1. Because of heterogeneity, we separately run the same algorithm with the same parameters for each node type.

4 EXPERIMENTS AND RESULTS

4.1 Dataset

Tradingview is a platform and social network for traders and investors where they can use different charts and technical features to make trading decisions. Traders can also share their opinions and analysis of the market in a section called Ideas. Users can choose between three options (i.e., Long, Short, and Education) to describe general purpose of their post. These user-generated labels were utilized for text classification tasks and a dataset was put together from people’s posts about the cryptocurrency market and their decision inferred from their analysis.
Table 4: Statistics of the dataset

| Field                                | Amount          |
|--------------------------------------|-----------------|
| Total posts                          | 24420           |
| Total labelled posts                  | 16590           |
| Mean length of posts                 | 70.05 words     |
| Median length of posts               | 34 words        |
| Total posts containing at least one  | 793             |
| price pattern                        |                 |
| Posts labelled as Long               | 13379 (80.6% of labels) |
| Posts labelled as Short              | 2984 (18% of labels) |
| Posts labelled as Education          | 227 (1.4% of labels) |

In this work, posts from 163 most popular cryptocurrency symbols were collected and stored in a database. In addition to assigning a label, market analysts select a timeframe for their posts to define the time range where their analysis or decisions are valid. Each post has a signature part in which writers usually share their websites or Telegram channels. Table 1 shows all the crawled information from Tradingview Ideas and a sample data point. Posts containing no text data were discarded. Overall 24420 posts are available in the dataset. Table 4 shows some statistics of our dataset.

4.2 Pre-process

Text data needs to be normalized and preprocessed for further use. In order to clean the documents in the corpus, these steps were taken:

- Normalizing documents
- Removing stop-words and punctuation marks
- Removing email addresses, newline characters, URLs and emojis
- Unifying price pattern variations, e.g., changing head n shoulders, head & shoulders, h & s to head-and-shoulders.

After cleaning the documents, they were broken down into their words to form a dictionary of words from the corpus.

4.3 Topic Modeling

As stated in previous sections, topic nodes are part of our graph structure. To extract topics and their distribution, Latent Dirichlet Allocation (LDA) method [3] is utilized. To find an optimal number of topics for the corpus, coherence score was calculated for different topic numbers. Figure 3 shows the result. This measure evaluates a single topic by measuring the degree of semantic similarity between high scoring words in the topic. This kind of measurements help distinguish between topics that are semantically interpretable topics and topics that are artifacts of statistical inference. There are various ways to calculate coherence score. Here we have used the Context Vector coherence score introduced by [1] that uses words co-occurrence counts. Based on the results shown in Table 3 the optimal number of topics was set to 6.

The final results of the topic modeling is shown in Table 5 in the form of each topic’s most frequent words. Each topic’s distribution was calculated for each document. Then the resulting vectors were mapped into two-dimensional space using t-SNE method [30]. The visualization of the data-points is shown in Figure 4. The distributions of words in topics and topics in documents is later used in our modeling process. The effectiveness of topic nodes in classification task is studied and the results show that these nodes improve the accuracy of the prediction.

For initial embedding of graph nodes, two models were used. FastText library [11] was used to encode words in our dictionary in a 100 dimensional embedding space. To encode the documents, sentence-transformers [27] pre-trained MiniLM model[33] with 6 layers was used. The MiniLM model maps documents to a 384 dimensional embedding space. Summary of all node embeddings and their dimensions are shown in Table 6.

4.4 Baselines and Results

The following models are used as baselines for comparison purposes:

Table 5: Topics and their frequent words

| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| time | buy | binance | drop | beh | price |
| bitcoin | term | analysis | trendline | crypto | support |
| market | tp | coin | beginning | structure | btc |
| wave | trade | position | monthly | charts | bullish |
| zec | short | dont | find | strategy | long |

Table 6: Graph nodes’ initial embedding

| Nodes                  | Embedding                           | Dim |
|------------------------|-------------------------------------|-----|
| Word node              | fasttext word embedding             | 100 |
| Document node          | Pretrained LM document embedding    | 384 |
| Topic node             | One-hot                             | 6   |
| Price Pattern node     | One-hot                             | 29  |
| Position node          | One-hot (used in the Link Prediction problem.) | 3   |
We also connect some of the price patterns to their implications (long or short), which brings new kinds of information to this setting. Both modifications result in a more informative and richer graph structure that is essential for GNN models to learn better representations. We illustrate the output embedding of documents in 4 different settings in Figure 4.

To examine the effectiveness of each group of nodes in the graph, experiments were done using different combinations of nodes. Table 8 summarizes the results for these experiments. The highest scoring result belongs to the link prediction task using HinSAGE model with word, document, topic, and position nodes. Using DGI and HinSAGE in the link prediction achieves almost the same result, however, it can be seen that DGI improves the performance of node classification in many cases.

5 RELATED WORKS

Graph-based approaches have recently gained attraction in natural language processing tasks. Graphs can express much more information in their structure than the bag-of-words approach. Therefore, representing text documents in form of graphs could result in more accurate text classification. Text graphs can have different kinds of structures. Osman and Barukab [24] categorized graph-based representation approaches in natural language processing into 5 groups. For example, word co-occurrence graphs, document-word graphs, phrase as a graph, etc. Graph representation approaches can be divided into two groups: single graphs for a corpus that contain all documents [19, 36] and individual graphs for each text in the document [15, 29, 40, 41]. Graphs can also be heterogeneous or non-heterogeneous. Non-heterogeneous graphs contain only one type of nodes and node representations which is more common due to simplicity of the processing steps [18, 19]. Heterogeneous graphs contain different types of nodes and representations [15, 20, 26]. A group of works in this area utilize graph-based approach for extracting embeddings that have captured similarities in the graph structures [5, 29, 34]. However, graph structures can be used in the downstream tasks by turning the classification problem into node classification or link prediction problems [36]. Shanavas et al. [29] used graph representation for each text document and defined a graph kernel for measuring similarity between graphs and then used SVM to classify documents. Jiang et al. [15] used a graph representation approach for documents with information of words like part of speech and their semantic features and then used graph mining approaches for feature vector extraction and classified the resulting vectors for each document.

In recent years, several deep learning methods have been proposed for graph-based text classification. For example, Yao et al. [36] leveraged Graph convolutional networks [16] for text classification. They constructed a graph with relations between words and documents and turned the text classification problem into a node classification task. Zhang et al. [40] proposed TextING for inductive text classification task and trained a graph neural network that learns from detailed word-word relations from individual graphs for each document. Another method was proposed by Zhao and Huang et al. [41] for text classification using graphs for each document in the corpus. They used Bi-LSTMs to extract sequential features from graphs and then used graph convolutional networks for classification. Liu and You et al. [21] constructed a text graph
| Model                  | Task                  | Precision | Recall | F1-score |
|-----------------------|-----------------------|-----------|--------|----------|
| DGI+HinSAGE           | Link prediction       | 0.9       | 0.89   | 0.89     |
| HinSAGE               | Link prediction       | 0.89      | 0.88   | 0.89     |
| Node2Vec[12]+Hinsage  | Link prediction       | 0.86      | 0.85   | 0.85     |
| fine-tuned BERT       | Document classification| 0.80      | 0.73   | 0.76     |
| SBERT embeddings+SVM  | Document classification| 0.82      | 0.67   | 0.72     |
| DGI+HinSAGE           | Node classification   | 0.7       | 0.68   | 0.68     |
| HinSAGE               | Node classification   | 0.71      | 0.66   | 0.68     |
| finBERT+SVM           | Document classification| 0.78      | 0.61   | 0.66     |

| Model                  | Task                  | Word | Document | Topic | Pattern | Position | F1-score |
|-----------------------|-----------------------|------|----------|-------|---------|----------|----------|
| DGI + HinSAGE         | Link Prediction       | ✓    |          | ✓     |         | ✓        | 0.885    |
| DGI + HinSAGE         | Link Prediction       | ✓    | ✓        | ✓     |         | ✓        | 0.89     |
| DGI + HinSAGE         | Link Prediction       | ✓    | ✓        | ✓     |         | ✓        | 0.885    |
| DGI + HinSAGE         | Link Prediction       | ✓    | ✓        | ✓     |         | ✓        | 0.888    |
| DGI + HinSAGE         | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.68     |
| DGI + HinSAGE         | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.688    |
| DGI + HinSAGE         | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.68     |
| DGI + HinSAGE         | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.512    |
| HinSAGE               | Link Prediction       | ✓    | ✓        | ✓     |         | ✓        | 0.894    |
| HinSAGE               | Link Prediction       | ✓    | ✓        | ✓     |         | ✓        | 0.896    |
| HinSAGE               | Link Prediction       | ✓    | ✓        | ✓     |         | ✓        | 0.888    |
| HinSAGE               | Link Prediction       | ✓    | ✓        | ✓     |         | ✓        | 0.888    |
| HinSAGE               | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.674    |
| HinSAGE               | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.652    |
| HinSAGE               | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.662    |
| HinSAGE               | Node Classification   | ✓    | ✓        | ✓     |         | ✓        | 0.681    |

tensor and developed a learning method to harmonize information from multiple graphs of their graph tensor for text classification. In summary, this work builds on previous works in machine learning on graphs and explores the text classification task on a corpus represented as a heterogeneous graph. This study also applies text classification in finance for predicting investors’ opinions which is new and different from previous sentiment analysis studies in this field.

6 CONCLUSION

We introduced a novel graph-based representation of stock market technical analysis reports for extracting financial insights and information. A heterogeneous graph with document, word, topic, price pattern, and position nodes was constructed from technical analysis documents such that each group of nodes had their specific initial node embeddings. Then, a graph neural network model was proposed to learn expressive representations for the node classification and edge prediction downstream tasks. Experimental results show that our model is capable of predicting labels accurately on our highly imbalanced dataset and outperforms the baseline methods. For the future direction, it is worthwhile to explore more types of nodes and relations for constructing more effective graph structures.

ACKNOWLEDGMENTS

We would like to thank everyone at Eveince for their valuable insights and support.

REFERENCES

[1] Nikolaos Aletras and Mark Stevenson. 2013. Evaluating Topic Coherence Using Distributional Semantics. In IWCS.
[2] Dogu Araci. 2019. FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. CoRR abs/1908.10063 (2019). arXiv:1908.10063 http://arxiv.org/abs/1908.10063
[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. J. Mach. Learn. Res. 3, null (mar 2003), 993–1022.
[4] Samuel W.K. Chan and Mickey W.C. Chong. 2017. Sentiment analysis in financial texts. Decision Support Systems 94 (2017), 53–64. https://doi.org/10.1016/j.dss.2016.10.006
[5] Boyan Chen, Guangquan Lu, Bo Peng, and Wenzhao Zhang. 2020. DGRL: Text Classification with Deep Graph Residual Learning. In Advanced Data Mining and Applications, Xiaochun Yang, Chang-Dong Wang, Md. Saiful Islam, and Zheng Zhang (Eds.). Springer International Publishing, Cham, 83–97.
[6] Raymond Chioung, Zongwen Fan, Zhongyi Hu, Marc T. F. Adam, Bernhard Lutz, and Dirk Neumann. 2018. A Sentiment Analysis-Based Machine Learning Approach for Financial Market Prediction via News Disclosures. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (Kyoto, Japan)
