TRENCHANT: TRENd PrediCtion on Heterogeneous InformAtion NeTworks

P. do Carmo*[,1], I. J. Reis Filho[2,1], R. Marcacini[1]

[1] University of São Paulo - São Carlos, Brazil
paulo.carmo@usp.br,ricardo.marcacini@icmc.usp.br
[2] State University of Minas Gerais - Frutal, Brazil
ivan.filho@uemg.br

Abstract. Events can be defined as an action or a series of actions with a determined theme, time, and place. Recently, event analysis tasks for knowledge extraction from news and social media have been explored. In particular, agribusiness events have multiple components for a successful prediction model. For example, price trend predictions for commodities can be performed through a time series analysis of prices. However, we can also consider events that represent external factors during the training step of predictive models. This paper presents a method for integrating agribusiness news events into trend prediction tasks. First, we propose to model events and time-series information through heterogeneous information networks (HIN) that allow multiple components to be directly modeled through multi-type nodes and edges. Second, we learn features from HIN through network embedding methods, i.e., network nodes are mapped to a dense vector of features. In particular, we propose a network embedding method that propagates the semantics of the pre-trained language models to a heterogeneous information network and evaluates its performance in trend prediction for agribusiness commodities prices. Finally, we propose a second method that leverages the HIN architecture to fine-tune a pre-trained language model before propagation. We show that using our proposed models of language-based embedding propagation is competitive with state-of-art network embeddings algorithms. Moreover, our proposal performs network embedding incrementally, allowing new events to be inserted in the same semantic space without rebuilding the entire network embedding.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3 [**Information Storage and Retrieval**]: Miscellaneous; I.7 [**Document and Text Processing**]: Miscellaneous

Keywords: agribusiness, event analysis, heterogeneous networks, network embedding, text mining

1. INTRODUCTION

Events can be defined as an action or a series of actions that occur at a specific time and place [Allan 2012; Cordeiro and Gama 2016; Chen and Li 2020]. Different types of event analysis can allow real-time monitoring for application domains like the economy, agribusiness, epidemics, medicine, sentiment analysis, and many other social behavior studies [Marcacini et al. 2017]. Agribusiness events are challenging to predict since they depend on (i) climate changes, (ii) historical data from the market, (iii) supply and demand, (iv) aggregate demand, and (v) politics. Climate changes are predictable through meteorologic data. Market history, as well as supply and demand data, are obtained from temporal series. However, aggregate demand and politics are also available in text data from the news or social networks. Since textual data is made for humans, text mining is a challenging task that requires multiple steps of processing [Venter et al. 2013].

Event analysis is the computational task for automatically identifying related events through text and other data such as timestamps, places, people, and entities involved [Hamborg et al. 2018; Xue

*This work was partially funded by The Coordination for the Improvement of Higher Education Personnel (CAPES) [process number 88887.513429/2020-00].

et al. 2019]. The data extracted comes in 5W1H model which is described as [Chen and Li 2020]: **what** happened; **when** it happened; **where** it happened; **who** is involved; **why** it happened; and **how** it happened. Thus, these techniques can be used to explore useful agribusiness data for trend prediction [Radinsky and Horvitz 2013; Ning et al. 2019], since they can be modeled as events. Event analysis is challenging since different topics can be related at some level. For example, the corn value can drop due to new exporting politics that lower international demand, and if identified earlier, a producer might be able to close a better domestic deal [dos Reis Filho et al. 2020].

Recently, heterogeneous networks have been used successfully for modeling large event datasets [Shi et al. 2016], since they model different components from events as nodes (e.g., when, where, who, why, and how), and network links express different relationships between these nodes. Thus, we can create a heterogeneous network, where each event is represented by itself and a set of components that complete the 5W1H. The use of heterogeneous networks also allows network completion tasks. Network completion tasks derive from link prediction. It is the task of estimating the connection between two nodes based on observed links and node features. The main difference from link prediction is that the method only tries to connect nodes of specific types in network completion. It can also be defined as a simple binary classification problem [Shi et al. 2016]. Network completion can also be modeled as a multi-class classification problem whenever a network node can be used as a label. Network completion as a classification uses node features as data. There are many techniques for obtaining node features from a network.

Network embeddings methods map each node into a low dimensional vector that represents the network topology and node type information [Chang et al. 2015; Huang and Mamoulis 2017; Setty and Hose 2018; Cui et al. 2018; Wu et al. 2020]. The recent literature has explored several methods to solve this task, from random walks within the entire network to deep learning methods [Wu et al. 2020]. However, since events can be partially represented as texts, such as news headlines or social networks posts, we can use neural language models, such as the Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al. 2018], to compute meaningful representations. Neural language models are trained over large textual datasets, including news headlines, so they have valuable general-purpose knowledge for event prediction. However, using just text embedding is not recommended since it ignores essential component information [Chen and Li 2020], which may lead to deeper connections between events, like location and actors involved.

This paper presents the TrEnd pRediction on heteRogeneous InFormatIon nEtwoRks (TRENCHANT), a language model-based embedding propagation method for heterogeneous event networks. While most existing network embedding methods mainly explore the network's topology, our method maps both (i) textual information about events; (ii) the complex relationships between events and their components to a low-dimensional vector space; and (iii) a fine-tuning strategy that leverages topological properties from the heterogeneous information network to enrich the textual embeddings. Existing methods of network embeddings are offline, which requires repeating the entire process when new nodes are added to the network. Some pipelines may include different techniques for dynamic insertions [Deng et al. 2019; 2020], but they must be modeled apart from the network embedding. On the other hand, our proposed method is naturally incremental. They use an initial neural language model embedding, pre-trained and fine-tuned respectively, to maintain a fixed vector space propagated to the entire network.

A preliminary version of our method was proposed in Carmo et al. [2021]. In this extended version, we incorporate the BERT model fine-tuning strategy using topological properties of the event network, as well as a more robust experimental analysis. Our main contributions are threefold:

—First, we propose a regularization framework for embedding propagation from a pre-trained BERT language model to all nodes of a heterogeneous event network. Thus, all network nodes are mapped to the same embedding space, allowing to compute similarities between textual and non-textual nodes of an event dataset.

—Second, we introduce a fine-tuning method of the BERT model from the topological properties of the event network. Although the proposed method can handle general-purpose pre-trained BERT models (TRENCHANT method), we discuss and evaluate scenarios in which fine-tuning the initial embeddings before propagation in the heterogeneous network yields promising results (FT-TRENCHANT method).

—Finally, we model the trend prediction problem as a classification task from the final embeddings extracted from an event network. This is a practical solution that allows different well-known classifiers, such as Long Short-Term Memory (LSTM) neural networks.

We evaluate the proposed methods with other information network embeddings methods on an agribusiness news dataset. We consider corn and soybean prices for trend prediction. We also demonstrate how enrolling the general knowledge of a pre-trained and fine-tuned neural language model into a heterogeneous information network embedding method performs in trend prediction scenarios. We show that using TRENCHANT and FT-TRENCHANT proposed methods is competitive with state-of-art network embeddings algorithms. Moreover, our proposal performs network embedding incrementally, allowing the insertion of new nodes in the same semantic space without rebuilding the entire network embedding.

2. RELATED WORK

Commodities trend prediction is usually based on time series analysis techniques like Integrated Autoregressive Moving Average (ARIMA, [Darekar and Reddy 2017]) and Integrated Seasonal Autoregressive Moving Average (SARIMA [Adanacioglu et al. 2012]). With the advance in text mining techniques, some works began to combine these text features with time series data for stock and commodity prediction [Wang et al. 2019; Chen et al. 2016]. In dos Reis Filho et al. [2020] the authors combine text and time-series data. The text was obtained from a bag-of-words model and concatenated with a decision tree model based on time series data. They evaluate the models by comparing the results from time series and the proposed model in a Support Vector Regression model for commodity price prediction.

Network embedding methods allow for node feature extraction on local and global levels. From Perozzi et al. [2014], the DeepWalk is a Word2Vec-based information network embedding method that executes multiple random walks to sample data for skip-gram model training. Node2Vec from Grover and Leskovec [2016] is an extension of the DeepWalk method that allows the user to choose between two parameters p and q that control how much of the random walk will follow breadth or depth search, respectively. Metapath2Vec was proposed by Dong et al. [2017] and is another extension of the DeepWalk designed for heterogeneous networks. It transforms the random walks into meta path-biased walks, i.e., the user defines a set of node types the algorithm should follow to explore the heterogeneous network. Struc2Vec [Ribeiro et al. 2017] aims to generate a representation that is aware of the network structure by constructing a multi-layer graph to encode the structural context through the hierarchy of paths and then applies random walks like DeepWalk. LINE was proposed by Tang et al. [2015], a network embedding method for very large information networks using edge sampling walks for local and global neighbors separately and unifying them through negative sampling. The authors Kipf and Welling [2017] proposed the Graph Convolutional Network (GCN) method based on the assumption that the 1-hop connections between nodes might not always indicate they have the same label. They modeled a neural network model to learn the graph correlations within some labeled nodes and then propagate these relations throughout the network to overcome this problem.

In Deng et al. [2019] the authors developed a dynamic vocabulary graph and modeled a GCN to identify key events and understand their evolution. Their proposal was evaluated within protest events datasets. Models like these can also be used for many forecasting problems, such as the multi-event and multi-actor forecasting tackled by Deng et al. [2020] with dynamic knowledge graphs and graph completion techniques, which are derived from link prediction. However, even though these methods successfully introduced structures for multi-typed events, they lack general information that we believe can be obtained through the propagation of pre-trained neural embeddings on top of an event HIN architecture.

Alongside network embedding and graph neural networks, pre-trained language models on billions of text data can be used to obtain dense vectors directly from textual data [Mikolov et al. 2013; Radford et al. 2018; Devlin et al. 2018]. Although they allow obtaining meaningful representations from event texts, they cannot extract embeddings from non-textual event components or consider more complex relationships. Neural language models like BERT also allow embedding fine-tuning of specific domain knowledge. Many authors explore BERT's fine-tuning capabilities and how this process affects its transformers [Hao et al. 2020; Merchant et al. 2020]. The authors in Sun et al. [2019] explored different approaches for fine-tuning BERT to different tasks and provided vital insights on what affects this process the most. In Peters et al. [2019], authors demonstrate that fine-tuned and other more specific models outperform their pre-trained counterparts, and they also provided recommendations for fine-tuning NLP tasks.

Information networks can be used to organize heterogeneous data from multiple contexts. They can also provide different features to extract knowledge from data, like the number of nodes, edges, and common neighbors. In this paper, we innovate them by leveraging information network topological properties to fine-tune a pre-trained BERT neural language model. Thus, we introduced a new approach to take advantage of the BERT model fine-tuning for applications involving event analysis from heterogeneous networks.

3. TREND PREDICTION ON HETEROGENEOUS INFORMATION NETWORKS

3.1 Event Modeling with Heterogeneous Networks

Events extracted from news data act as digital sensors, as they are published around the time they happened and contain information. This information can be extracted directly from the text, and its metadata [Hamborg et al. 2018]. We chose to extract a 4W1H variation without the *when* characteristics since it is not as good data as the publication date. We also consider metadata from commodities prices to generate the trend indicators of the temporal series. The trend labels are calculated according to the $[week|month]/year$ period and commodity they represent. With the news features extracted and trend labels calculated, we can model different heterogeneous networks to each combination (Figure 1).

A heterogeneous event network is formally defined as a triple $N = (O, R, W)$, where O is the set of object nodes, R is the set of connections between objects, and W is the weight of those connections [dos Santos et al. 2020]. Heterogeneous event networks are information networks with the set $O > 1$, meaning they have more than one object node type. This allows the representation to adapt to various real-world data and event analysis, in particular events represented by multiple components, such as places, time, names of people, and organizations. In this case, the set of objects is defined as $O = \{O_e \cup O_d \cup O_w \cup O_l \cup O_a \cup O_y \cup O_h \cup O_t\}$, where the subset O_e represents event nodes, O_d represents date nodes, O_w represents what nodes, O_l represents location nodes, O_a represents actors nodes (e.g. people or organizations), O_y represents why nodes, O_h represents how nodes and O_t represents trend nodes (our desired labels).

3.2 Event Embedding Propagation

Network embeddings methods map each node in a low dimensional vector that represents network topology and node type information [Chang et al. 2015; Huang and Mamoulis 2017; Setty and Hose 2018; Cui et al. 2018; Wu et al. 2020]. Formally, network embedding generates a function $g : N \rightarrow \mathbb{R}^d$,

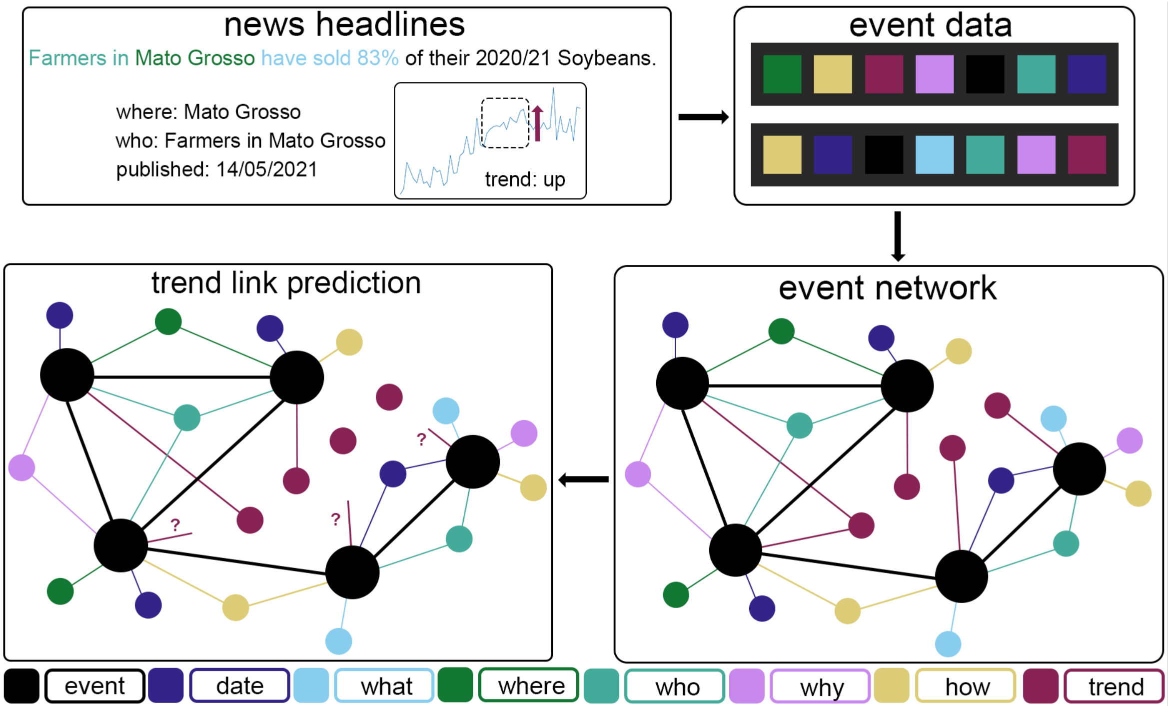


Fig. 1: Visual representation of the proposal. The event components are extracted from news headlines and metadata, while the trend symbolizes the fluctuation of a commodity price.

where the nodes N are represented in a d-dimensional space. Each node $n \in N$ has an embedding $\mathbf{f_n} \in \mathbb{R}^d$.

Our paper investigates network embedding through embedding propagation [Yang et al. 2019], where we can take advantage of initial event embeddings using a BERT-based pre-trained neural language model. The general problem for embedding propagation proposed in this paper is presented in Equation 1, inspired by a general graph regularization framework [Ji et al. 2010]. It can be solved via minimization with quadratic programming or through iterative methods based on label propagation [Belkin et al. 2006; Zhu et al. 2003]. The first term determines that neighboring nodes n_i and n_j in the network have similar embedding vectors $\mathbf{f}_{n_i}$ and $\mathbf{f}_{n_j}$, where Ω is a distance function, and w_{n_i,n_j} indicates the weight of the connection between the nodes n_i and n_j. In the second term, the objective is that the initial embeddings $\mathbf{g}_{n_i}$ of the event nodes O_e are preserved according to a μ factor (with $\mu > 0$). In this case, $\mathbf{g}_{n_i} \in \mathbb{R}^d$ are d-dimensional embeddings obtained by pre-trained language models from the texts of the event node n_i, where $n_i \in O_e$.

$$Q(\mathbf{F}) = \frac{1}{2} \sum_{n_i,n_j \in N} w_{n_i,n_j} \Omega(\mathbf{f}_{n_i}, \mathbf{f}_{n_j}) + \mu \sum_{O(n_i) \in O_e} \Omega(\mathbf{f}_{n_i}, \mathbf{g}_{n_i}) \tag{1}$$

Using the initial BERT embeddings strategy for events is potentially promising for network event analysis. In particular, trend prediction is a challenging repeatable task. Pre-trained embedding allows new nodes to be added and used in the network immediately after an incremental embedding propagation. The embedding propagation maintains the same original semantic space of the BERT model. In contrast, other implementations of network embedding may require recreating the embeddings for the whole network.

3.3 BERT Fine-tuning using Event Network Topological Properties

The pre-trained embeddings of the BERT language neural model are often obtained using two approaches. The first approach involves extracting embeddings from a pre-trained model from general-purpose texts, such as news and Wikipedia. The second approach focuses on fine-tuning the neural language model considering the textual dataset available from the downstream task. Although BERT fine-tuning is a promising alternative to extract better initial embeddings to our proposed method, the success of this approach depends on a large volume of texts and labeled data. Such dependencies often make fine-tuning in real-world applications unfeasible.

Since our proposed method assumes a pre-trained language model, both approaches (general-purpose or fine-tuned) can be employed. However, in addition to using pre-trained general-purpose BERT models to generate initial event embeddings, our proposed method also allows a BERT fine-tuned model. Therefore, we propose a fine-tuning strategy by leveraging the event network topology to learn better embeddings for the downstream trend prediction task without relying on a human-labeled event dataset for embeddings learning. We claim this is relevant to fine-tuning language models for event analysis, as we incorporate complex relationships between event components into BERT embeddings.

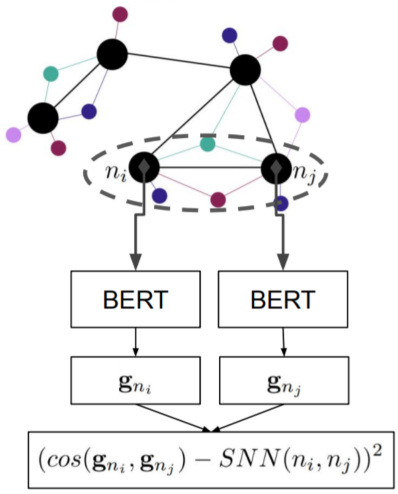


Fig. 2: Fine-tuning

Figure 2 shows an overview of our fine-tuning pipeline, the BERT model using topological properties of the event network. Let n_i and n_j be two event nodes. Remember that event nodes have associated textual data. Moreover, event nodes are connected with nodes representing components of geographic information, person names, places, and others. Equation 2 defines the Shared Nearest Neighbors (SNN) measure, which calculates the similarity between event nodes from the network topology, measuring the number of shared component nodes,

$$SNN(n_i, n_j) = \frac{\mathcal{S}(n_i) \cap \mathcal{S}(n_j)}{\mathcal{S}(n_i) \cup \mathcal{S}(n_j)} \tag{2}$$

where $\mathcal{S}(n_i)$ returns the set of neighboring nodes to n_i. The SNN measure varies in the range $[0, 1]$, meaning the more shared neighbors, the closer to 1 and the greater the topological similarity between the two nodes.

We extend the well-known Siamese Neural Networks framework for BERT fine-tuning in the context of sentence similarity [Reimers and Gurevych 2019]. Siamese networks use two different input vectors (e.g., initial BERT event embeddings) to compute comparable output vectors (fine-tuned event embeddings). While the existing approaches use predefined manual scores between pairs of texts as similarity ground truth (which may be unfeasible in real-world applications), we propose using the SNN measure as a target for fine-tuning. In this case, we fine-tune the embeddings of the event texts so that the cosine similarity $cos(\mathbf{g}_{n_i}, \mathbf{g}_{n_j})$, where $\mathbf{g}_{n_i}$ and $\mathbf{g}_{n_j}$ are the BERT embeddings of event nodes n_i and n_j respectively, is approximated by $SNN(n_i, n_j)$ — which represents topological properties of the event network. The loss function for the proposed fine-tuning is the MSE function (mean squared error), according to Equation 3,

$$MSE = \frac{1}{k}\sum_{i=1}^{k}(cos(\mathbf{g}_{n_i}, \mathbf{g}_{n_j}) - SNN(n_i, n_j))^2 \tag{3}$$

where k is the number of event pairs extracted from the event network for the BERT fine-tuning process. After the fine-tuning process, such embeddings can be used in the embedding propagation function described in Equation 1.

3.4 Trend Prediction

With the network embeddings calculated, we must apply the data collected to a machine learning algorithm. In this paper, we have chosen a long-short term memory (LSTM) for supervised multi-class classification with the network embedding data. The LSTM we used accommodates memory units. The memory units are composed of three smaller units (u, c_{in}, c_{out}). The units u (Equation 4 (i)) apply a weighted sum, where w represents the weight, for each value y, and its result is outputted through an activation function. These units are limited by control units c_{in} (Equation 4 (ii)) and c_{out} (Equation 4 (iii)) that control when the iteration t will go forward or keep recurring. The initial value y inserted to the LSTM comes from the propagated embeddings f and changes throughout training.

$$\begin{array}{rl} (i) & {u_u}^{(t)} = \sum_u w_u y^{u(t-1)} \\ (ii) & {c_{in}}^{(t)} = \sum_u w_{in_j u} y^{u(t-1)} \\ (iii) & {c_{out}}^{(t)} = \sum_u w_{out_j u} y^{u(t-1)} \end{array} \tag{4}$$

These units must learn the correct weights to predict trends from an embedding vector. Thus the LSTM is trained by a process called back-propagation. A back-propagation optimizer uses a gradient (e.g. Equation 5) that calculates the error E between the actual label from a training to the time step $T^{'}, T$ and generates a Δ to update each weight w_{ij} from the LSTM network.

$$\Delta w^{E^{total}}(T^{'}, T) = \sum_{t=T^{i}+1}^{T} \Delta w^{E(t)} \rightarrow \Delta w_{ij} = -\alpha \frac{\partial E^{total}(T^{'}, T)}{\partial w_{ij}} \tag{5}$$

The LSTM used in this proposal has: (i) a dense layer with as many units as the input (512) that uses a $relu$ activation function (Equation 6), allowing the optimization process to turn units off when necessary; (ii) we mini-batch experiments and determined 64 memory units were the optimal number for these experiments; (iii) the last layer calculates the probabilities of each class since it uses a SoftMax [Goodfellow et al. 2016] activation function; and (iv) it uses the Adam optimizer during training. With this LSTM, we can leverage the data represented by the embeddings in a neural classification, allowing non-supervised methods to compete with graph neural networks.

$$f(u) = \max(0, u) \quad (6)$$

4. EXPERIMENTAL EVALUATION

4.1 Datasets

We use a dataset related to agribusiness news to evaluate the trend prediction. It contains news related to the corn and soybean commodities extracted from Soybean & Corn Advisor[1] and the historical prices from CEPEA[2] (Centro de Estudos Avançados em Economia Aplicada). This dataset allows us to extract the events and components from the news text. We calculate Trends' classes in four ways, generating four networks as shown in Table I. Each network has the same events and components but generates trends' labels nodes by different commodities and time windows. In addition, we use the DistilBERT-Multilingual[3] model to generate the initial embedding for each event headline text.

Table I: Overview of heterogeneous event networks used in the experimental evaluation.

Network	#Nodes	#Events	#Dates	#Whats	#Wheres	#Whos	#Whys	#Hows	Time window	Commodity
#1	4348	2322	380	48	115	9	244	1226	weekly	corn
#2	4348	2322	380	48	115	9	244	1226	weekly	soybean
#3	4056	2322	89	48	115	9	244	1226	monthly	corn
#4	4056	2322	89	48	115	9	244	1226	monthly	soybean

4.2 Evaluation criteria and experiment setup

We use traditional metrics to evaluate the multi-class classification scenario: macro precision, macro recall, and macro $F1$. Another important metric to evaluate these methods is the $F1$ for the *big_down* and *big_up* classes. We consider it essential because these classes indicate a critical market tendency transition. It is important to know the performance of the methods on them. The evaluation scenarios consist of two windows for obtaining trends and two splits for predicting them on all four networks. We also evaluate average execution times. In Table II we present class numbers for training examples for each scenario.

Table II: Overview of scenarios configurations with train and test sizes.

Scenario	Time window	Commodity	#Weeks	#Months	Train size	Test size
#1	weekly	corn	24	-	2196	126
#2	weekly	corn	48	-	2037	285
#3	weekly	soybean	24	-	2196	126
#4	weekly	soybean	48	-	2037	285
#5	monthly	corn	-	6	2203	119
#6	monthly	corn	-	12	2029	293
#7	monthly	soybean	-	6	2203	119
#8	monthly	soybean	-	12	2029	293

With the experiment scenarios displayed, we also need to account for class imbalance between all four labels (*big_down*, *down*, *up*, and *big_up*). Table III presents all class distributions for each scenario train/test split

[1] Available at: `http://soybeansandcorn.com`
[2] Available at: `https://www.cepea.org.br`
[3] Available at: `https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2`

Table III: Overview of class balance for each scenario on *train*/*test* splits.

Scenario	*big_down*(*train*/*test*)	*down*(*train*/*test*)	*up*(*train*/*test*)	*big_up*(*train*/*test*)
#1	97/5	999/57	977/57	123/7
#2	83/19	964/92	887/147	103/27
#3	189/12	918/63	898/41	191/10
#4	155/46	888/93	838/101	156/45
#5	28/0	1006/38	1093/81	76/0
#6	28/0	973/71	952/222	76/0
#7	26/23	1096/15	1000/61	81/20
#8	26/23	1096/15	846/215	61/40

We compare TRENCHANT and FT-TRENCHANT with state-of-art network embeddings methods: DeepWalk, Node2Vec, Metapath2Vec, Struc2Vec, LINE, and GCN. We use the parameters recommended by the authors in the respective original papers for each baseline method. Regarding the number of dimensions of the embeddings, we used 512 for all methods since it is the dimension used by the DistilBERT language model. All experimental data, source code, and networks are available at the GitHub repository `https://github.com/PauloRVdC/TRENCHANT`.

4.3 Results and Discussion

Considering the experiment setup explained previously, we will present results for each scenario, pointing out relevant overall and specific class performance results. We will also evaluate the models considering stability through box plot variations on scenario results.

In Table IV we present the results for the odd number scenarios (#1, #3, #5 and, #7), which represent the smallest splits. We can see TRENCHANT and FT-TRENCHANT are the best performers on most scenarios and metrics. More specifically, FT-TRENCHANT is best in both scenarios constructed with soybean trends. At the same time, TRENCHANT has the best $F1$ and better recall for scenario #1, which is the smallest split for corn trends. We can also observe that TRENCHANT had worst performance than baseline methods on scenario #3, and FT-TRENCHANT managed to beat them with some margin. These results show that our fine-tuning pipeline adds some knowledge to the representation. However, it also shows that our fine-tuning pipeline does not help all scenarios since performance has decayed in all corn scenarios.

Table IV: Trend prediction performance for scenarios #1, #3, #5 and, #7 on three metrics (macro $F1$, macro precision (pre) and, macro recall (rcl)). Highest scores are in bold.

	Scenario #1			**Scenario #3**			**Scenario #5**			**Scenario #7**		
	$F1$	**pre**	**rcl**	$F1$	**pre**	**rcl**	$F1$	**pre**	**rcl**	$F1$	**pre**	**rcl**
DeepWalk	0.24	0.24	0.26	0.23	0.23	0.26	**0.48**	**0.48**	**0.48**	0.13	0.12	0.21
Node2Vec	0.27	0.30	0.27	0.23	0.23	0.25	0.24	0.25	0.24	0.18	0.22	0.25
Metapath2Vec	0.22	**0.32**	0.27	0.20	0.28	0.26	0.11	0.29	0.08	0.15	0.21	0.23
Struc2Vec	0.17	0.22	0.22	0.20	0.23	0.23	0.18	0.29	0.14	0.19	0.21	0.27
LINE	0.24	0.25	0.24	0.25	0.26	0.26	0.31	0.32	0.31	0.18	0.21	0.25
GCN	0.27	**0.32**	**0.30**	0.21	0.22	0.23	0.39	0.42	0.45	0.14	0.12	0.24
TRENCHANT	**0.29**	0.30	**0.30**	0.21	0.22	0.24	0.23	0.24	0.23	0.21	0.30	0.24
FT-TRENCHANT	0.21	0.22	0.21	**0.33**	**0.34**	**0.34**	0.26	0.27	0.26	**0.25**	**0.31**	**0.35**

In Table V we present the results for the even number scenarios (#2, #4, #6 and, #8), which represent the largest splits. Compared to the smallest scenarios, TRENCHANT and FT-TRENCHANT have lacked some performance, going from being the best in most scenarios to having the best $F1$ and precision on scenario #8. Nevertheless, this shows us that the fine-tuning process adds some knowledge that helps discriminate trends for the soybean commodity. We can also see that TRENCHANT is the second-best on scenario #2, which aligns with its good performance on scenario #1.

Table V: Trend prediction performance for scenarios #2, #4, #6 and, #8 on three metrics (macro $F1$, macro precision (pre) and, macro recall (rcl)). Highest scores are in bold.

	Scenario #2			**Scenario #4**			**Scenario #6**			**Scenario #8**		
	$F1$	**pre**	**rcl**	$F1$	**pre**	**rcl**	$F1$	**pre**	**rcl**	$F1$	**pre**	**rcl**
DeepWalk	0.24	0.26	0.27	0.20	0.27	0.24	**0.36**	**0.40**	**0.39**	0.16	0.20	0.24
Node2Vec	0.24	0.25	0.26	0.22	0.23	0.26	0.24	0.27	0.26	0.17	0.24	0.25
Metapath2Vec	0.19	0.25	0.20	**0.30**	**0.35**	**0.32**	0.10	0.28	0.07	0.13	0.19	0.24
Struc2Vec	0.20	0.23	0.22	0.22	0.24	0.24	0.16	0.28	0.12	0.17	0.22	0.22
LINE	0.24	0.26	0.25	0.22	0.23	0.24	0.25	0.28	0.27	0.17	0.26	0.25
GCN	**0.26**	**0.31**	**0.31**	0.19	0.17	0.24	0.37	0.38	0.40	0.16	0.24	**0.32**
TRENCHANT	0.25	0.26	0.27	0.19	0.18	0.20	0.22	0.25	0.23	0.18	0.38	0.28
FT-TRENCHANT	0.23	0.23	0.24	0.20	0.20	0.22	0.20	0.23	0.20	**0.20**	**0.39**	0.25

In Table VI we present the results specific for the critical classes (*big_down* and *big_up*). We can observe that even though TRENCHANT and FT-TRENCHANT only achieve the best results on scenarios #1 and #3, respectively, the overall performance guarantees a result throughout all scenarios. Another takeaway from these results is that Metapath2Vec achieves the best performance in most scenarios, showing us that the meta-paths allow for good coverage of unbalanced class features.

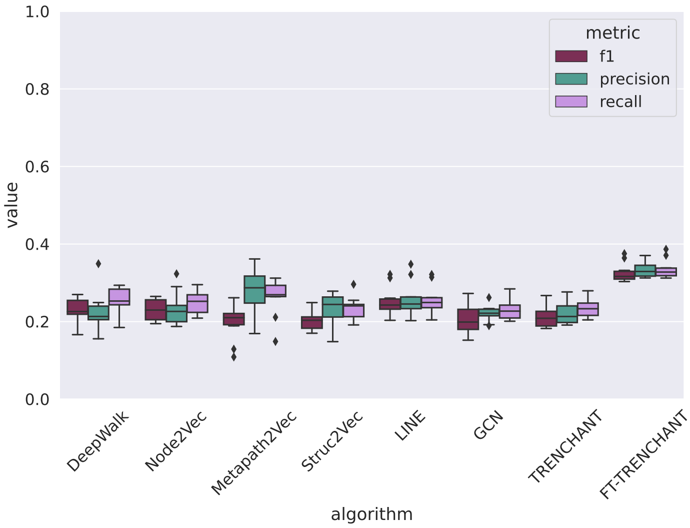
Table VI: Trend prediction performance for all scenarios on two metrics (*big_down* $F1$ (bd), *big_up* $F1$, (bu)). Highest scores are in bold.

	Scenario #1		**Scenario #2**		**Scenario #3**		**Scenario #4**		**Scenario #5**		**Scenario #6**		**Scenario #7**		**Scenario #8**	
	(bd)	**(bu)**	**(bd)**	**(bu)**	**(bd)**	**(bu)**	**(bd)**	**(bu)**	**(bd)**	**(bu)**	**(bd)**	**(bu)**	**(bd)**	**(bu)**	**(bd)**	**(bu)**
DeepWalk	0.00	0.00	0.01	0.01	0.01	0.00	0.03	0.03	-	-	-	-	0.00	0.00	0.00	0.00
Node2Vec	**0.07**	0.02	0.02	0.01	0.03	0.03	0.04	0.02	-	-	-	-	0.00	0.05	0.01	0.03
Metapath2Vec	**0.07**	0.00	0.01	**0.12**	0.07	0.10	**0.29**	**0.25**	-	-	-	-	**0.28**	0.16	**0.27**	0.14
Struc2Vec	0.04	0.10	**0.08**	**0.12**	0.09	**0.13**	0.16	0.20	-	-	-	-	0.05	**0.21**	0.04	**0.20**
LINE	0.00	0.09	0.02	0.07	0.07	0.05	0.09	0.07	-	-	-	-	0.01	0.04	0.01	0.05
GCN	0.00	0.00	0.00	0.02	0.00	0.00	0.01	0.00	-	-	-	-	0.00	0.00	0.00	0.00
TRENCHANT	**0.07**	**0.12**	0.00	0.05	0.06	0.04	0.06	0.02	-	-	-	-	0.02	0.09	0.04	0.07
FT-TRENCHANT	0.00	0.02	0.00	0.07	**0.23**	0.01	0.03	0.02	-	-	-	-	0.04	0.09	0.07	0.10

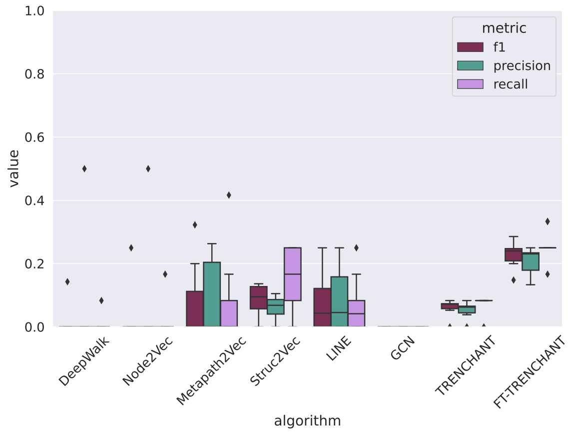
In Figure 3 we present five box plots from scenario #3 that showcase the stability of methods through five variations of the $F1$, precision, and recall metrics. The first box plot (Figure 3a) shows that FT-TRENCHANT not only has the highest average of the macro metrics but is also the most stable, followed by TRENCHANT and LINE. Another standout is that box plots for specific classes (Figures 3b, 3c, 3d, 3e) show that different methods get better results on different classes. Most methods have a good performance on the *up* class, but FT-TRENCHANT is among the best, while TRENCHANT is among the methods that managed *big_down* class performance on some runs. Overall, FT-TRENCHANT has good performance throughout the macro and class-specific metrics and manages to edge out TRENCHANT and the baselines in most metrics.

We also evaluated average execution times for all algorithms on all executions for all scenarios. In Figure 4 we present the average seconds for each algorithm. It is important to denote that: (i) GCN is a semi-supervised graph neural network that is parallel on GPU; (ii) LINE is a neural network embedding method that is parallel on GPU; (iii) Struc2Vec is an extension of DeepWalk parallel on CPU; (iv) DeepWalk, Node2Vec, Metapath2Vec, TRENCHANT and, FT-TRENCHANT are sequential; and (v) all methods, except GCN, execution times include training and prediction with the LSTM. With that in mind, GCN has the fastest execution times, followed by TRENCHANT and FT-TRENCHANT. These results can be explained for two reasons: (i) they are linear methods considering nodes and edges of the network; and (ii) their stable embeddings allowed the LSTM loss to be stable sooner, resulting in a shorter training, specially FT-TRENCHANT, which has fine-tuned initial embeddings.

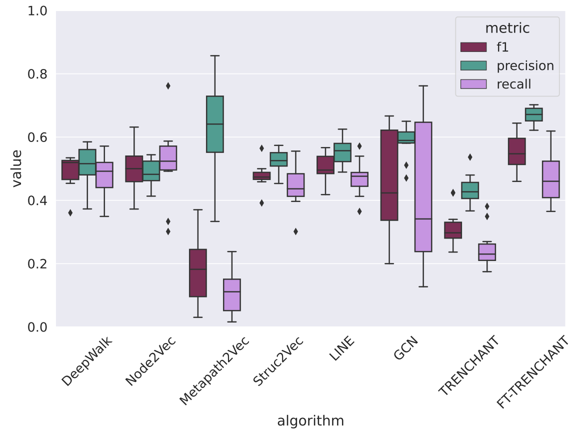
Overall, TRENCHANT and FT-TRENCHANT are methods competitive with the state-of-art network embedding methods. Also, an initial embedding ensures that all nodes will be in the same vector space. This single vector space allows new nodes added to the network to receive a final embedding



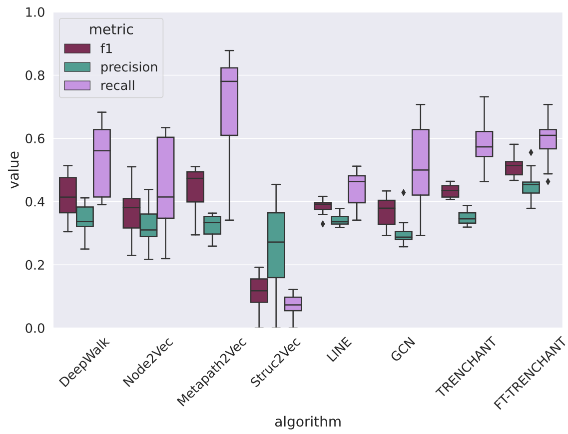
(a) macro $F1$, precision and recall.



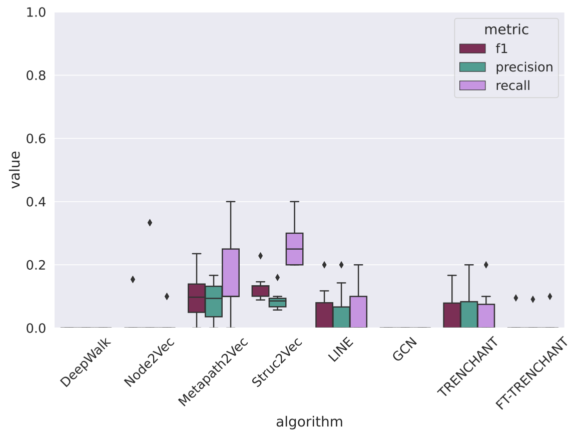
(b) *big_down* $F1$, precision and recall.



(c) *down* $F1$, precision and recall.



(d) *up* $F1$, precision and recall.



(e) *big_up* $F1$, precision and recall.

Fig. 3: Box plots for scenario #3 with the metrics: macro and class-specific $F1$, precision, and recall.

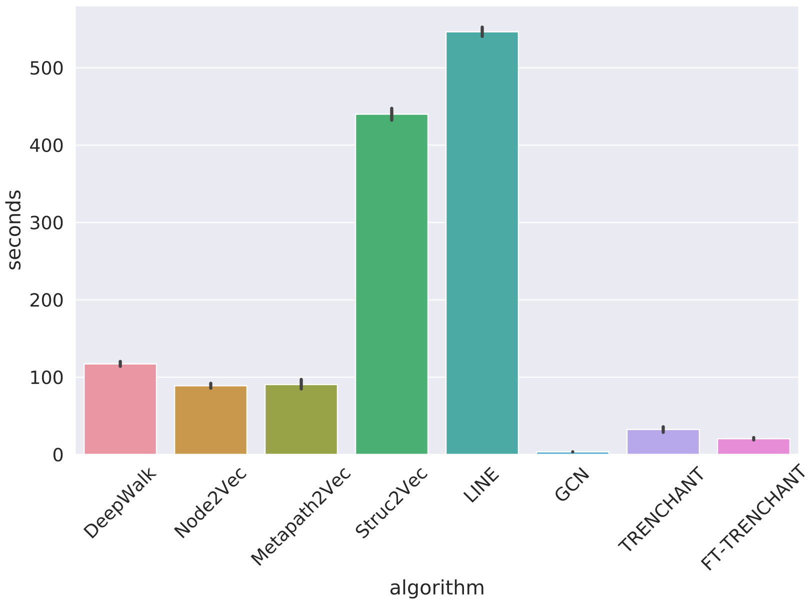


Fig. 4: Average execution times for all algorithms on all executions for each scenario in seconds.

with a few iterations on top of the existing embedding. It also provides a more stable performance since all executions will have the same starting point.

5. CONCLUSION

This paper introduces an embedding propagation method of pre-trained neural language models. It also proposes an extension that leverages the heterogeneous network architecture for fine-tuning a neural language model. Furthermore, we propose a pipeline for trend price prediction of agribusiness commodities prices and evaluate our proposed models against network embedding models. Finally, our experiment results show that using an embedding propagation technique from a BERT-based model allows network embedding without recalculating the entire network. We also show that the use of text information, combined with simple network topology, is competitive against state-of-art topology network embeddings algorithms when the text database is well-curated. The results with the fine-tuned extension also show that graph data can be incorporated into BERT embeddings.

We plan to incorporate weights on the different types of relations for future work, thereby allowing the embedding propagation to consider more topology information. We also want to investigate semi-supervised embedding propagation methods to incorporate the existing labels into the resulting embeddings. Finally, Attention mechanisms in place of an LSTM is another scenario we can evolve this work.

Acknowledgments: The authors of this work would like to thank the Center for Artificial Intelligence (C4AI-USP) and the support from the São Paulo Research Foundation (FAPESP grant 2019/07665-4) and from the IBM Corporation. This work was also supported by National Council for Scientific and Technological Development (CNPq) [process number 426663/2018-7] and FAPESP [process number 2019/25010-5].

REFERENCES

Adanacioglu, H., Yercan, M., et al. An analysis of tomato prices at wholesale level in turkey: an application of sarima model. *Custos e Agronegócio Online* 8 (4): 52–75, 2012.

Allan, J. *Topic detection and tracking: event-based information organization.* Vol. 12. Springer Science & Business Media, 2012.

Belkin, M., Niyogi, P., and Sindhwani, V. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research* vol. 7, pp. 2399–2434, 2006.

Carmo, P., Reis Filho, I., and Marcacini, R. Commodities trend link prediction on heterogeneous information networks. In *Anais do IX Symposium on Knowledge Discovery, Mining and Learning.* SBC, pp. 81–88, 2021.

Chang, S., Han, W., Tang, J., Qi, G.-J., Aggarwal, C. C., and Huang, T. S. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* pp. 119–128, 2015.

Chen, H.-H., Chen, M., and Chiu, C.-C. The integration of artificial neural networks and text mining to forecast gold futures prices. *Communications in Statistics - Simulation and Computation* 45 (4): 1213–1225, 2016.

Chen, X. and Li, Q. Event modeling and mining: a long journey toward explainable events. *The VLDB Journal* 29 (1): 459–482, 2020.

Cordeiro, M. and Gama, J. Online social networks event detection: a survey. In *Solving Large Scale Learning Tasks. Challenges and Algorithms.* Springer, pp. 1–41, 2016.

Cui, P., Wang, X., Pei, J., and Zhu, W. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering* 31 (5): 833–852, 2018.

Darekar, A. and Reddy, A. Predicting market price of soybean in major india studies through arima model. *Journal of Food Legumes* 30 (2): 73–76, 2017.

Deng, S., Rangwala, H., and Ning, Y. Learning dynamic context graphs for predicting social events. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* pp. 1007–1016, 2019.

Deng, S., Rangwala, H., and Ning, Y. Dynamic knowledge graph based multi-event forecasting. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* pp. 1585–1595, 2020.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dong, Y., Chawla, N. V., and Swami, A. metapath2vec: Scalable representation learning for heterogeneous networks. In *ACM SIGKDD international conference on knowledge discovery and data mining.* pp. 135–144, 2017.

dos Reis Filho, I. J., Correa, G. B., Freire, G. M., and Rezende, S. O. Forecasting future corn and soybean prices: an analysis of the use of textual information to enrich time-series. In *Anais do VIII Symposium on Knowledge Discovery, Mining and Learning.* SBC, pp. 113–120, 2020.

dos Santos, B. N., Rossi, R. G., Rezende, S. O., and Marcacini, R. M. A two-stage regularization framework for heterogeneous event networks. *Pattern Recognition Letters* vol. 138, pp. 490–496, 2020.

Goodfellow, I., Bengio, Y., and Courville, A. 6.2. 2.3 softmax units for multinoulli output distributions. *Deep learning* (1): 180, 2016.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining.* pp. 855–864, 2016.

Hamborg, F., Lachnit, S., Schubotz, M., Hepp, T., and Gipp, B. Giveme5w: main event retrieval from news articles by extraction of the five journalistic w questions. In *International Conference on Information.* Springer, pp. 356–366, 2018.

Hao, Y., Dong, L., Wei, F., and Xu, K. Investigating learning dynamics of BERT fine-tuning. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing.* Association for Computational Linguistics, Suzhou, China, pp. 87–92, 2020.

Huang, Z. and Mamoulis, N. Heterogeneous information network embedding for meta path based proximity. *arXiv preprint arXiv:1701.05291*, 2017.

Ji, M., Sun, Y., Danilevsky, M., Han, J., and Gao, J. Graph regularized transductive classification on heterogeneous information networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases.* Springer, pp. 570–586, 2010.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Marcacini, R. M., Rossi, R. G., Nogueira, B. M., Martins, L. V., Cherman, E. A., and Rezende, S. O. Websensors analytics: Learning to sense the real world using web news events. In *Simp. Brasileiro de Sistemas Multimídia e Web.* pp. 169–173, 2017.

Merchant, A., Rahimtoroghi, E., Pavlick, E., and Tenney, I. What happens to bert embeddings during fine-tuning? *arXiv preprint arXiv:2004.14448*, 2020.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems.* pp. 3111–3119, 2013.

Ning, Y., Zhao, L., Chen, F., Lu, C.-T., and Rangwala, H. Spatio-temporal event forecasting and precursor identification. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. pp. 3237–3238, 2019.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. pp. 701–710, 2014.

Peters, M. E., Ruder, S., and Smith, N. A. To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987*, 2019.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding with unsupervised learning. *Technical report, OpenAI*, 2018.

Radinsky, K. and Horvitz, E. Mining the web to predict future events. In *Proceedings of the sixth ACM international conference on Web search and data mining*. pp. 255–264, 2013.

Reimers, N. and Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. pp. 3982–3992, 2019.

Ribeiro, L. F., Saverese, P. H., and Figueiredo, D. R. struc2vec: Learning node representations from structural identity. In *ACM SIGKDD international conference on knowledge discovery and data mining*. pp. 385–394, 2017.

Setty, V. and Hose, K. Event2vec: Neural embeddings for news events. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. pp. 1013–1016, 2018.

Shi, C., Li, Y., Zhang, J., Sun, Y., and Philip, S. Y. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering* 29 (1): 17–37, 2016.

Sun, C., Qiu, X., Xu, Y., and Huang, X. How to fine-tune bert for text classification? In *China national conference on Chinese computational linguistics*. Springer, pp. 194–206, 2019.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*. pp. 1067–1077, 2015.

Venter, M., Strydom, D., and Grové, B. Stochastic efficiency analysis of alternative basic grain marketing strategies. *Agrekon* 52 (sup1): 46–63, 2013.

Wang, J., Wang, Z., Li, X., and Zhou, H. Artificial bee colony-based combination approach to forecasting agricultural commodity prices. *International Journal of Forecasting*, 2019.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Xue, F., Hong, R., He, X., Wang, J., Qian, S., and Xu, C. Knowledge based topic model for multi-modal social event analysis. *IEEE Transactions on Multimedia*, 2019.

Yang, C., Zhang, J., and Han, J. Neural embedding propagation on heterogeneous networks. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, pp. 698–707, 2019.

Zhu, X., Ghahramani, Z., and Lafferty, J. D. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*. pp. 912–919, 2003.