Graph Neural Network for Merger and Acquisition Prediction

Yinfei Li
University College London
London, United Kingdom
ucemigx@ucl.ac.uk

Jiafeng Shou
University College London
London, United Kingdom

Philip Treleaven
University College London
London, United Kingdom

Jun Wang
University College London
London, United Kingdom

ABSTRACT

This paper investigates the application of graph neural networks (GNN) in Mergers and Acquisitions (M&A) prediction, which aims to quantify the relationship between companies, their founders, and investors. M&A is a critical management strategy to decide if the company is to grow or downsize, and M&A prediction has been a challenging research topic in the past few decades. However, the traditional methods of predicting M&A probability are only based on the company's fundamentals, such as revenue, profit, or news. Instead, GNN takes full advantage of those relationship data to expand feature dimension and improve the prediction result. Our M&A prediction solution integrates with the topic model for text analysis, advanced feature engineering, and several tricks to boost GNN. The approach achieves a high Area-Under-Curve score (AUC) 0.952, which is better than the previous record 0.888. The true positive rate is 83% with a low false positive rate 7.8%, which performance is better than the previous benchmark record 70.9%/10.6%.

CCS CONCEPTS

• **Theory of computation → Graph algorithms analysis**.

KEYWORDS

Merger and Acquisition, Graph Theory, Graph Neural Network, Topic Model

ACM Reference Format:
Yinfei Li, Jiafeng Shou, Philip Treleaven, and Jun Wang. 2021. Graph Neural Network for Merger and Acquisition Prediction. In *2nd ACM International Conference on AI in Finance (ICAIF'21), November 3–5, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3490354.3494368

1 INTRODUCTION

Graph Neural Networks (GNN) offers a step-change in M&A analysis. M&A is a critical and essential method to boost the growth of an enterprise through finance transactions. It is a challenging task to predict the probability of a target company to be acquired. With high risk, M&A brings a high return to the Venture Capital (VC). A M&A prediction model with high precision could help VC hedging risk and generate the effective investment strategy. Due to the blurred distinction between mergers and acquisitions, this research uses M&A and acquisition as synonyms.

The traditional M&A prediction is based on company fundamentals, such as company value, revenue, and profit. A Logistic Regression (LR) or Support Vector Machine (SVM) models are then applied to complete classification tasks. With the same data input, all these baseline models have a close prediction result due to 'No Free Lunch' theories [14], because the data quality limits the prediction accuracy upper bound. Data quality means the missing values, the external information, and the error data.

The core idea presented in this paper is that the company's acquisition probability depends on its connected entities, such as its investors, its founders or its providers. For example, say there is a famous VC where 50% of its portfolio companies have been acquired. Hence the next company, funded by the same VC, is likely to have a higher acquisition probability according to the algorithm.

The main motivation of our research is to add relationships as external data, quantify the relationships between social network, and validate whether the Graph Neural Network (GNN) works on M&A prediction. To achieve this goal, this paper will first review the current best models for M&A prediction, for example, Bayesian Networks (BN) integrated with topic model [16], and review graph neural networks with their applications. Then, the traditional machine learning methods will be evaluated and implemented as the benchmark performance. Following these steps, GNN based model with special feature engineering will be evaluated and the influence factors of GNN model will be explored.

In general, there are two main challenges that block high accuracy M&A prediction: a) finance data is sparse, and b) too many negative samples in the M&A data.

To be specific, the first challenge is that most company data have missing features. The primary reason is that small companies usually do not publish their full fundamental data. Typically, business information platforms focus on large scale companies and often ignores the small companies or provide out-of-date data. This leads to biased data and biased classifiers. For example, the trained classifier predicts the large company with a higher probability of being acquired. Although there are some solutions to obtain missing values, such as Bayesian Gaussian tensor decomposition [4], it may be incompatible with some classifier.

ICAIF'21, November 3–5, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-9148-1/21/11…$15.00
https://doi.org/10.1145/3490354.3494368

The second challenge is imbalanced sample data. In M&A sample data, the acquisition is often having low probability of occurring. Then, there are nearly 10% positive samples (being acquired successfully) and 90% negative samples (not being acquired). The imbalanced samples lead the model to a local optimal and over-fitting. While backpropagation is applied, the update direction goes towards negative prediction due to 90% negative samples. Besides, due to small positive samples, instead of being generalized, the model tries to record the positive sample features in memory, which causes the over-fitting.

2 RELATED WORK

Predicting merger and acquisition via machine learning methods is not a new approach in financial computing area. The previous best model, a Bayesian Network (BN) with the topic model, was proposed by a team from Carnegie Mellon University [16]. They also found that data quality is the key to improving the prediction result. For example, integrating the model with article information from TechCrunch for each company. Using this article information, the CMU team enhanced the Crunchbase data with external data. However, the article data provides sparse information. Although they claim the topic model improved prediction results. The research reported in this paper questions the CMU approach (see discussion section).

In order to compare the GNN and other methods, our research has taken the same dataset as the CMU team from CrunchBase and TechCrunch. TechCrunch is a popular news publication website, reporting on the business of technology, start-ups, venture capital funding, and Silicon Valley. CrunchBase is a database with information about start-ups, investors, trends, milestones.

The data used by the CMU team contains 81,219 companies, 107,274 persons, 7,328 financial organizations, 3,955 service providers, 25,895 funding rounds and 6,173 acquisitions from CrunchBase profiles. CrunchBase also provides the relation between those entities, which are applied by the GNN model. The CMU team also downloaded the article data about companies from TechCrunch.

Recently, GNN research has developed multiple variants for different tasks, such as Graph2Seq model (G2S) for edge-informative graph [17], and Graph Structured Recurrent Neural Network model (GSRNN) for graph data in time series [8]. GNNs are widely applied in protein interface prediction [6], to simulate physics interaction network [2], and with social network for recommendation system. We believe this is the first paper to apply GNNs to M&A prediction topic.

3 ALGORITHMS

This section introduces the algorithms covering the core concepts in this research, the basis for graph theory, topic model and graph neural network. The traditional machine learning methods which are widely applied in industry will not be discussed.

Graph theory is a branch of mathematics with "graphs" as the research object, and is an important part of combinatorial and discrete mathematics. A graph is a mathematical structure used to model the pairwise relationship between objects. It consists of "vertices" (also known as "nodes" or "points") and "edges" (also known as "arcs" or "lines") that connect these vertices. ") composition. It is worth noting that the vertex set of a graph cannot be empty, but the set of edges can be empty. The graph may be undirected, which means that the edges in the graph need not distinguish directions when connecting vertices. Otherwise, the graph is directed.

Graph Neural Network: traditional analytics methods for M&A prediction are typically applied to row-column based data, and cannot process graph data. For instance, each company has many investors, and the model is required to quantify investors' value. The traditional data preprocessing would list features: (investor 1 value, investor 2 value, ..., investor n value). However, two problems occur: 1) Each company has variable length investor amount that it cannot verify variable n in GNN model. 2) The order of investors' data input would affect the result that each feature column has different coefficient in training. A potential solution is to calculate the mean value of the company's investors. In fact, mean is one of the aggregators in GNN, which aims to aggregate the information around the nodes. Further issues include: 1) All features are required to do such data preprocessing, which costs time; and 2) It is equivalent to 1-layer GNN that cannot diffuse node information and can-not extract the deeper information. GNNs overcome these issues by providing a solid and general framework for graph data.

In summary, GNNs are connectionist models, which capture the dependence of graphs via message passing between the node and graphs [19]. GNNs are a general topic model? containing 4 categories: 1) Re-current Graph Neural Networks (RecGNNs); 2) Convolu-tional Graph Neural Networks (ConvGNNs); 3) Graph Autoencoders (GAEs); and 4) Spatial-temporal Graph Neural Networks (STGNNs).

Since this research is restricted to node classification tasks and data quality, the paper would not introduce RecGNNs and only focuses on ConvGNNs [15]. There are two types of ConvGNN: 1) spectral-based ConvGNN; and 2) spatial-based ConvGNNs. Both are used in this research. RecGNN's main problem is its recurrent network architecture. Recurrent network architecture is unable to extract the long-distance dependency (low fitting ability) and does not support the parallel computation. Natural Language Process (NLP) domain has replaced the recurrent network architecture with attention mechanism or convolution layers to process the sequence text data. Therefore, ConvGNNs propose applying convolutional operation to extract graph data information, which solves the inefficient computation and information smooth issues.

4 DATA AND FEATURE ENGINEERING

To compare algorithm performance with baseline model from CMU team, this research use the same dataset as baseline. There are two parts of data: 1) information on company, person, financial organization, product, and service-provider downloaded from CrunchBase updated before January 10, 2012; and 2) the article data of companies extracted from TechCrunch.

4.1 Exploratory Data

This section reviews the Crunchbase and TechCrunch data used in the experiments. Crunchbase provides company data, person data, financial organisation data, service provider data and product data.
Company Data
There are 79111 companies in the dataset. Each company at most has the following features: *name*, *permalink*, *crunchbase_url*,

homepage_url, *blog_url*, *blog_feed_url*, *twitter_username*, *category_code*, *number_of_employees*, *founded_year*, *founded_month*, *founded_day*, *deadpooled_year*, *deadpooled_month*, *deadpooled_day*, *deadpooled_url*, *tag_list*, *alias_list*, *email_address*, *phone_number*, *description*, *created_at*, *updated_at*, *overview*, *image*, *products*, *relationships*, *competitions*, *providerships*, *total_money_raised*, *funding_rounds*, *investments*, *acquisition*, *acquisitions*, *offices*, *milestones*, *ipo*, *video_embeds*, *screenshots*, *external_links*.
permalink is the primary key of CrunchBase database for each entity. *permalink* is the unique ID for each entity while *name* might be duplicate. It helps to build the relationship between different entities. *relationships*, *providerships*, *products*, *funding_rounds* define the relationship with person, service-provider, product, financial-organization respectively. Those url data, overview text data, video data and picture data are abandoned that they do not offer effective information.
Person Data
There are 130915 person profiles in dataset. Each person at most has the following features: *first_name*, *last_name*, *permalink*, *crunchbase_url*, *homepage_url*, *birthplace*, *twitter_username*, *blog_url*, *blog_feed_url*, *affiliation_name*, *born_year*, *born_month*, *born_day*, *tag_list*, *alias_list*, *created_at*, *updated_at*, *overview*, *image*, *degrees*, *relationships*, *investments*, *milestones*, *video_embeds*, *external_links*, *web_presences*. In fact, except *born_xxx*, *degree*, *relationships*, *investments* features, the rest of features are no use for providing no information.
Financial Organization Data
There are 7758 financial-organizations in the dataset with following features: *name*, *permalink*, *crunchbase_url*, *homepage_url*, *blog_url*, *blog_feed_url*, *twitter_username*, *phone_number*, *description*, *email_address*, *number_of_employees*, *founded_year*, *founded_month*, *founded_day*, *tag_list*, *alias_list*, *created_at*, *updated_at*, *overview*, *image*, *offices*, *relationships*, *investments*, *milestones*, *providerships*, *funds*, *video_embeds*, *external_links*.
Service Provider Data
There are 4348 service-providers in the dataset with following features: *name*, *permalink*, *crunchbase_url*, *homepage_url*, *phone_number*, *email_address*, *tag_list*, *alias_list*, *created_at*, *updated_at*, *overview*, *image*, *offices*, *providerships*, *external_links*.
Product Data
There are 16979 products with following features: *name*, *permalink*, *crunchbase_url*, *homepage_url*, *blog_url*, *blog_feed_url*, *twitter_username*, *stage_code*, *deadpooled_url*, *invite_share_url*, *tag_list*, *alias_list*, *deadpooled_year*, *deadpooled_month*, *deadpooled_day*, *launched_year*, *launched_month*, *launched_day*, *created_at*, *updated_at*, *overview*, *image*, *company*, *milestones*, *video_embeds*, *external_links*.
TechCrunch Data.
There are 58107 articles for 8688 companies. As shown in Figure 1, the article distribution is of limited value since most of companies only have 1 article or 0. There are 88.5% companies with articles number less than 8. Only the big technology companies, such as Facebook and Google, have more than a thousand articles. Figure 1 also shows the distribution of word count. Most of the article are short, typically a paragraph. In this situation, a topic model designed for document-level analysis may not be the best choice. For paragraph-level text analysis, current State-Of-The-Art (SOTA) pretrained language models, such as BERT [5] or XLNet [18], have better performance on capturing the long-distance dependency.

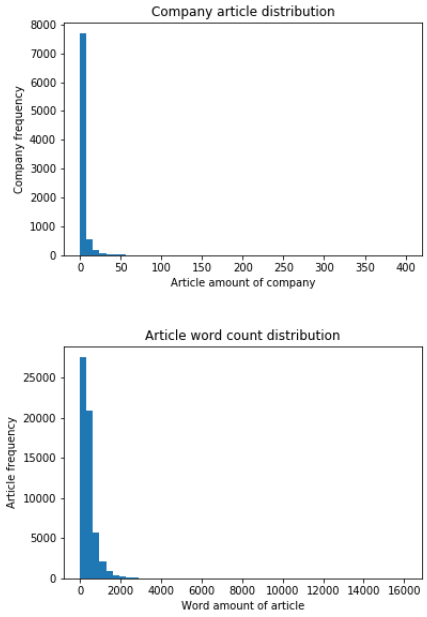


Figure 1: The distribution of article and word count.

4.2 Feature Engineering

In machine learning, feature engineering is the process of using domain knowledge to extract features from raw data via data mining techniques.

Feature engineering is a critical part of the machine learning task. An elaborate feature engineering model using logistic regression has the better performance than SOTA model with raw data. This section covers feature engineering on node data and edge data. This research has multiple algorithms to implement, and each algorithm requires different data format with different feature engineering. Hence, there are three kinds of feature engineering, which are entity data, integrated entity data and edge data on both directed graph and undirected graph. The detailed effective features information and code can be obtained by contacting the lead author.

5 EXPERIMENT SETUP

This section formally presents the GNN algorithm employed in this research. The algorithms cover the baseline model, graph neural network, graph neural network with XGBoost, graph neural network with topic model. In the experiment, we split the dataset into three sets: training set, validation set and test set. We mainly use 5-fold Cross-Validation (CV) [12] to find the optimal parameters.

The loss function for baseline model is cross entropy:

$$Loss(\bar{\mathbf{Y}}, \mathbf{Y}) = -\sum_{i=1}^{N}\sum_{j=1}^{2} \mathbf{Y}_{ij} \log \bar{\mathbf{Y}}_{ij} \tag{1}$$

where $\bar{\mathbf{Y}} \in \mathbb{R}^{N\times 2}$ is the predicted label matrix. GNN model requires different label data structure $\mathbf{Y}^{graph}$. Because GNN predicts M&A

Data	Description
$\mathbf{X}^{company} \in \mathbb{R}^{38492\times24}$	single company entity data.
$\mathbf{X}^{company}_{topic} \in \mathbb{R}^{38492\times29}$	company entity data with topic distribution.
$\mathbf{Y} \in \mathbb{R}^{38492\times2}$	M&A prediction label for baseline model.
$\mathbf{X}^{all} \in \mathbb{R}^{172981\times103}$	integrated node feature matrix for GNN.
$\mathbf{X}^{all}_{topic} \in \mathbb{R}^{172981\times108}$	node feature matrix with topic distribution for GNN.
$\mathbf{A}^{j} \in \mathbb{R}^{172981\times172981}$	adjacency matrix for undirected graph
$\mathbf{E}^{di} \in \mathbb{R}^{206414\times13}$	edge feature matrix for directed graph.
$\mathbf{E}^{multi} \in \mathbb{R}^{412828\times17}$	edge feature matrix for directed graph with inverse edge.
$\mathbf{Y}^{graph} \in \mathbb{R}^{172981\times2}$	M&A prediction label for GNN.
$\mathbf{M} \in \mathbb{R}^{172981}$	M&A prediction mask for GNN.

Table 1: The data used in this research.

probability $\bar{\mathbf{Y}}^{graph}$ on each entity while not each entity is the target of M&A prediction. The mask $\mathbf{M}$ is designed to remove the effect of non-target entity. If i-th node is the company type, then $\mathbf{M}_i = 1$, otherwise $\mathbf{M}_i = 0$. Then the loss function for GNN:

$$Loss(\bar{\mathbf{Y}}^{graph}, \mathbf{Y}^{graph}) = -\sum_{i=1}^{N} \mathbf{M}_i \sum_{j=1}^{2} \mathbf{Y}^{graph}_{ij} \log \bar{\mathbf{Y}}^{graph}_{ij} \tag{2}$$

In the data chapter, it is discussed that the positive/negative sample is not balanced. To solve, this research uses undersampling instead of using weighted loss. According to $\frac{\#positive}{\#negative} = 0.1285$, the idea of undersampling is to sampling the majority class sample that makes $\frac{\#positive}{\#negative} = 1$. In each training step, do the undersampling and obtain the balanced training sample.

5.1 Baseline

The baseline model is XGBoost model. It was supposed to repeat the former best model [16], which is a BN model. However, the author did not post the architecture of the BN. Hence, this research would not implement their model but their baseline model, Logistic Regression (LR) and Support Vector Machine (SVM).

Those baseline models do not require graph data. In this section, only $\mathbf{X}^{company}$ and $\mathbf{Y}$ would be used. They are not the target of the research. Hence, the specific mathematical equations would not listed. Both of two algorithm has two hyper-parameters to tune, which the optimal hyper-parameter could be found by single grid search. Before employing those two algorithms, the data normalization is required:

$$\bar{\mathbf{X}} = \frac{\mathbf{X} - \mu}{\sigma} \tag{3}$$

where μ is the mean of $\mathbf{X}$ and σ is the standard deviation of $\mathbf{X}$. It makes the input data distribution with mean 0 and stand deviation 1. It helps the algorithm converges quickly.
In XGBoost, a boost tree is written in the form:

$$\bar{y} = F(\mathbf{x}) = \sum_{k=1}^{K} f_k(\mathbf{x}), f_k \in \mathcal{F} \tag{4}$$

where f_k is Classification and Regression Tree (CART). The specific training algorithm has been described in Section **??**. There are 6 hyper-parameters to tune. If each hyper-parameter has N possible value to validate, then it costs time complexity $O(N^6)$ to find the optimal hyper-parameters. A better strategy is to find the sub-optimal hyper-parameters through 3 grid searches.

5.2 Spectral based graph neural network

To operate convolution on graph data, spectral-based ConvGNN provides the framework that operates convolution in frequency domain after Fourier transform [10].

Spectral-based ConvGNNs only process the undirected graph, which contains non-information edge. Spectral-based ConvGNNs has a solid mathematical foundation in graph signal process related to Laplacian matrix and Fourier transform [9] [3].

A spectral-based graph neural network is going to use node feature matrix $\mathbf{X}^{all}$, adjacency matrices $\mathbf{A}^{j}$, and label data $\mathbf{Y}^{graph}$, $\mathbf{M}$. Although this algorithm is not invented by this research, it is adopted to this research task with multiple edge types. Spectral-based GNN requires undirected graph with non-informative edge, while spatial-based GNN accepts informative edge. Hence, this algorithm is set as a control group to explore how informative edge affects the M&A prediction result. This algorithm also needs data normalization in Eq. 3. After normalization, Eq. **??** is rewritten in the form:

$$\mathbf{H}^{(0)} = \mathbf{X}^{all} \tag{5}$$

$$\mathbf{H}^{(k)} = \sigma(\sum_{j} \bar{\mathbf{A}}^{j} \mathbf{H}^{(k-1)} \mathbf{W}^{(kj)}) \tag{6}$$

where $\bar{\mathbf{A}}^{j} = I + \mathbf{D}^{-\frac{1}{2}} \mathbf{A}^{j} \mathbf{D}^{-\frac{1}{2}}$, j is the edge type, $\mathbf{W}^{(kj)} \in \mathbb{R}^{B_{k-1}\times B_k}$ is the learnable parameters, B_k is the k-th layer hidden state dimension number $\mathbf{H}^{(k)} \in \mathbb{R}^{172981\times B_k}$. At the last layer t, set $b_t = 2$ then

$$\bar{\mathbf{Y}} = softmax(\mathbf{H}^{(t)}) \tag{7}$$

where $\bar{\mathbf{Y}}$ is the predicted M&A probability.

In general, there are a series of spectral-based ConvGNNs models with solid mathematical foundation and proof, which operates convolution on graph data. However, spectral-based ConvGNN is designed for undirected graph with non-informative edge. If turn a informative edge graph into undirected graph, then the information of the edge wastes.

5.3 Spatial based graph neural network

We now present our spatial-based graph neural network model. Spectral-based ConvGNNs are restricted by undirected graph, and RecGNNs are restricted by recurrent network architecture. Spatial-based ConvGNNs has the flexible settings with- out the restrictions of other models. It could process node-level, edge-level and graph-level task. For instance, edge-level task is to predict the relation between two nodes, and the graph-level task is to find the optimal path between two nodes given a graph.

Spatial-based ConvGNNs is based on a simple core idea: aggregate the spatial neighbor hidden information into the center node, and update the center node. Therefore, the aggregator and updater is the basis of spatial-based ConvGNN model.

Algorithm 1 Graph Convolution Layer Block

Input : $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N]^T$, $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_M]^T$
Output: $\mathbf{H}'$, $\mathbf{E}'$

```
for m=1 to M do
    e'_m ← φ^e(e_m, h_{r_m}, h_{s_m})
end
for n=1 to N do
    let E'_n = {(e'_k, r_k, s_k)}_{r_k=n,k=1:M}
    ē'_n ← ρ^{e→v}(E'_n)
    h'_n ← φ^v(ē'_n, h_n)
end
return (H' = {h'_n}_{n=1:N}, E' = {(e'_m, r_m, s_m)}_{m=1:M})
```

In the implementation, the Graph Nets framework from DeepMind are applied [1]. They denote $E = \{(\mathbf{e}_m, r_m, s_m)\}_{m=1:M}$ as the set of the edge, where r_m is the index the receiver node (start node), and r_k is the index of the sender node (end node).

Algorithm 1 shows the single graph convolution layer in this research, where $\mathbf{H} = \mathbf{X}$ in the first layer. The main difference between this algorithm and DeepMind's Graph Nets [1] is that this algorithm remove the graph-level operation, which contributes nothing to the final result. Hence, there are 3 valid operations for updating and aggregating:

$$\mathbf{e}'_m = \phi^e(\mathbf{e}_m, \mathbf{h}_{r_k}, \mathbf{h}_{s_k}) \tag{8}$$

$$\bar{\mathbf{e}}'_n = \rho^{e\to v}(E'_n) \tag{9}$$

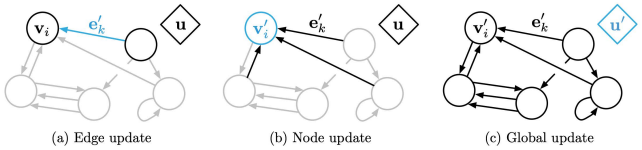
$$\mathbf{h}'_n = \phi^v(\bar{\mathbf{e}}'_n, \mathbf{h}_n) \tag{10}$$

where ϕ is the learnable updater and ρ is the aggregator. Eq. 8 extracts the edge information with connected node information, Eq. 9 aggregates the edge node information into the receiver node, Eq. 10 update the node status. ϕ is a single layer neural network in the implementation. Assume the input are concatenated as a single vector $\hat{\mathbf{X}}$, which is equivalent to individual input. Then Eq. 8 and Eq. 10 can be simplified as:

$$\phi(\hat{\mathbf{X}}) = {}_r(\text{BatchNormalization}(\hat{\mathbf{X}})\mathbf{W}_\theta) \tag{11}$$

$$\text{BatchNormalization}(\hat{\mathbf{X}}) = \Gamma\frac{\hat{\mathbf{X}} - \mathbb{E}[\hat{\mathbf{X}}]}{\sqrt{\text{Var}[\hat{\mathbf{X}}]}} + \Omega \tag{12}$$

where Γ and Ω is the learnable parameters for batch normalization, batch normalization replaces the data normalization, and dropout is applied to avoid overfitting with rate r [11][7].



(a) Edge update (b) Node update (c) Global update

Figure 2: Updates in three levels: edge, node, global. Blue element is being updated, and black elements are being involved in the update. [1].

Figure 2 indicates how the three types of elements update. One deficiency is that directed graph does not diffuse the information. For example, most of the nodes only receive information not send information, which obstructs the information diffusion. To solve this, add an edge with inverse direction to the original direction edge.

Finally, take the final layer's node embedding $\mathbf{H}^{(t)}$ to to classification:

$$\bar{\mathbf{Y}} = softmax(\mathbf{H}^{(t)}\mathbf{W}_\theta) \tag{13}$$

where $\mathbf{W}_\theta \in \mathbb{R}^{D_t \times 2}$ transforms the embedding into 2 kinds of probability. After predicting M&A probability $\bar{\mathbf{Y}}$, use Eq. 2 to calculate the masked cross entropy loss with back propagation updating weight.

5.4 Graph Attention Neural Network

According to traditional GAT Eq. 14 15, the attention mechanism is only applied in the aggregation process without considering informative edge vectors. Moreover, current SOTA attention models mainly employ self-attention mechanism [13]. Because self-attention could capture distant dependency. In the adopted GAT model, Eq. 8 and Eq. 9 are designed to blend edge information:

$$\mathbf{h}_v^{(k)} = \sigma\Big(\sum_{u\in\mathcal{N}_v} a_{vu}\mathbf{W}^{(k-1)}\mathbf{h}_u^{(k-1)}\Big) \tag{14}$$

$$a_{vu} = softmax(g(\mathbf{a}^T[\mathbf{W}^{(k-1)}\mathbf{h}_v || \mathbf{W}^{(k-1)}\mathbf{h}_u])) \tag{15}$$

$$\mathbf{e}'_m = \phi^e(\mathbf{e}_m, \mathbf{h}_{r_m}, \mathbf{h}_{s_m}) = (\mathbf{e}_m || \mathbf{h}_{r_m} || \mathbf{h}_{s_m})\mathbf{W}^{value} \tag{16}$$

$$\bar{\mathbf{e}}'_n = \rho^{e\to v}(E'_n) = \sum_{\mathbf{e}'_{(u,v)}\in E'_n} a_{uv}\mathbf{e}'_{(u,v)} \tag{17}$$

$$a_{uv} = \frac{\exp\Big((\mathbf{h}_u\mathbf{W}^{key})^T(\mathbf{h}_v\mathbf{W}^{query})\Big)}{\sum_u \exp\big((\mathbf{h}_u\mathbf{W}^{key})^T(\mathbf{h}_v\mathbf{W}^{query})\big)} \tag{18}$$

$$\tag{19}$$

where $\mathbf{W}^{value}, \mathbf{W}^{key}, \mathbf{W}^{query}$ are learnable parameters for self attention [13]. Self-attention mechanism treats both neighbor nodes and corresponding edge as key and value simultaneously, the center node as the query. Then, calculate the attention weight for each contributor and aggregate the weighted sum instead of the sum. The node updater Eq. 10 remains the same. Moreover, it could be extended to multi-head mechanism. Regarding above equation $(\mathbf{e}'_m, \bar{\mathbf{e}}'_n, a_{uv})_i$ as a single head of self attention mechanism with index i, the output is the concatenation of weighted values from all the head. However, multi-head self attention mechanism requires high hardware specification and large dataset. This research cannot support such model and only applies single head.

6 EXPERIMENT RESULTS

This section lists the result of different models. Firstly, this section argues the metrics of the task and the previous best record. Then, the result of different models is shown, with analyzing the advantage and disadvantage of each model. Finally, this ssection explores the potential influence factor to M&A prediction.

6.1 Indicators and Records

We use true positive rate (TPR) and false positive rate (FPR) as the main evaluation indicators. We also considered the area under the ROC curve (AUC) as a metric. However, TRP and FPR cannot

accurately measure the performance of the model, which leads to errors in CMU baseline model Table 3 lists the model performance under the "computer" industry category. There are two main problems in model training: 1) The baseline cannot train LR and SVM well. 2)The topic model cannot improve the BN model because TPR and FPR can mislead performance. This study proves the first problem. The specific results will be shown in the next section. The first problem may be caused by abnormal data points and no data standardization.

Model	TPR	FPR	**AUC**	F_1 score	precision
LR	2.8%	0.3%	not given	0.053	not given
SVM	39.6%	0.1%	not given	0.564	not given
BN	59.9%	2.2%	0.882	0.677	not given
BN+5 topic	70.9%	10.6%	0.888	0.559	not given

Table 2: The model performance from former record [16].

In the second problem, it is observed that AUC still remains the same, where AUC is the most accurate metrics to evaluate model. It could be proved in another way. In the 'Computer' category, there are 2,668 positive samples and 20,777 negative samples that $\frac{\#\text{positive}}{\#\text{negative}} = 0.128$. 0.128 is the accuracy baseline that if someone predicts the M&A of all companies success, then he can at least achieve accuracy $\frac{0.128}{0.128+1} = 11.35\%$.

	Label True	Label False
Predict True	1598	457
Predict False	1070	20320
	Label True	Label False
Predict True	1892	2202
Predict False	778	18575

Table 3: The top confusion table is plain BN result and the bottom one is BN with topic model result. The plain BN model predicts 1598 True Positive (TP) cases, 1070 False Negative (FN) cases, 20320 True negative (TN) cases, 457 False Positive (FP) cases, while the BN model with topic model predicts 1892 TP cases, 778 FN cases, 18575 TN cases, 2202 FP cases.

According to Table 3, within topic model, TP cases increases 294 and FP cases increases 1745 that $\frac{\#TP}{\#FP} = 0.168$. It means it is only little better than predicting M&A of all companies success. Hence, topic model does not extremely improve the model but improve the TPR. The precision is the metrics drawing greatest attention of investors, because they are not interested in companies predicted as unsuccessful. However, the precision depends on the threshold. With different threshold, the same result would produce the different precision, TPR, FPR metrics. Although all company M&A probabilities would be computed, the investors only care about the most potential company indeed. In another word, investors only focus on the companies with highest probability. Therefore, this research chooses precision as the main metrics with 0.95 threshold. Besides, AUC score does not depend on threshold value. If the model has a higher AUC score, then tuning threshold value could always achieve a higher score on other metrics.

This research mainly adopts AUC and precision (0.95 threshold) as the main metrics, and also considers F_1 score, TPR, FPR (0.5 threshold) as the auxiliary metrics.

6.2 Baseline model

Figure 3 and Table 4 list the confusion matrix and performance of the three benchmark models.

Model	TPR	FPR	**AUC**	F_1 score	precision
LR	54.5%	0.6%	0.863	0.684	0.556
LR+5 topic	57.8%	1.7%	0.853	0.675	0.778
SVM	51.0%	0.4%	0.808	0.655	0.968
SVM+5 topic	55.6%	1.0%	0.865	0.677	0.923
XGBoost	69.9%	3.1%	0.909	0.716	0.885
XGBoost+5 topic	66.5%	2.6%	0.899	0.710	0.867

Table 4: Baseline model performance from this research.

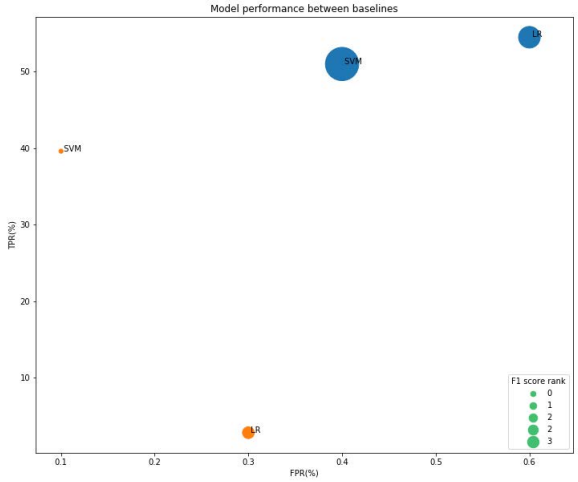
LR and SVM models have an extreme low FPR, which means both of them predict M&A with a prudent investment style. LR tends to over-fitting on the outlier and get a low precision value. In the meanwhile, SVM is not sensitive to the outlier value, and is suitable for classification task. Then SVM acquires a low AUC score but high precision value. The cost is to neglect more positive samples. Based on ensemble algorithm, XGBoost achieves the benchmark level performance that it makes an extraordinary AUC score, 0.909. XGBoost also predicts M&A precisely with high precision 0.885, which is a balanced investment investment style. Figure 4 lists the feature importances of XGBoost algorithm. According to the feature importance, it concludes following important factors for a company to get M&A: the company experience, the received investment, and the environment market (conclusion from the currency type feature).

Compared with this research baseline in Figure 3 (a), [16] baseline does not gain the best performance. It was mainly caused by feature engineering on abnormal values and no data normalization. Since this research LR achieves TPR/FPR/AUC/F_1 score 54.5%/0.6%/0.863/0.684, their BN model achieves TPR/FPR/AUC/F_1 score 59.9%/2.2%/0.882/0.677 that does not improve too much.

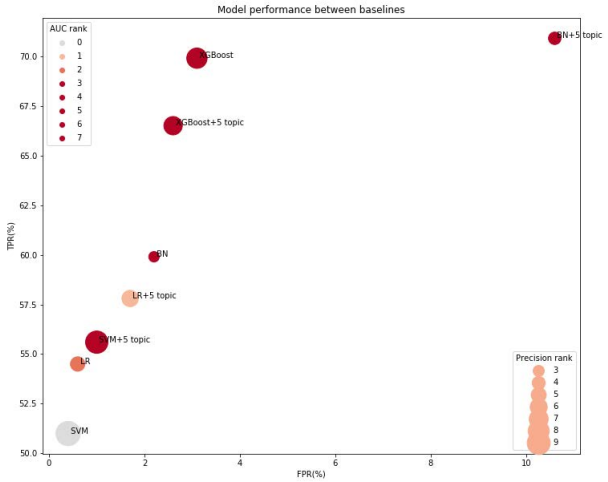
In Figure 3 (b), it shows that topic model does not improve the model that sometime it even hurts the model. In fact, it is proved in last section that topic model does not improve their BN model but only increase TPR and FPR meaninglessly. It also convinces the metrics choice of this research.

6.3 Graph Neural Network

Compared with the baseline XGBoost, 2 layers spectral-based GNN model (Model 1) does not improve AUC score or precision. Spectral-based GNN does not gain the effect of the neighbor nodes information. However, regarding plain neural network is a weak learner, Model 1 competitor should be LR and SVM. In that view, GNN architecture takes slight effect of neighbor nodes information in this task. In addition, spectral-based GNN does not use the edge information. As its control group, 2 layers spatial-based GNN model (Model 2) has a higher AUC score, which confirms edge information is helpful. Due to under-fitting, Model 2 has a lower precision. It is proved in the following comparisons.



(a) The baseline model LR, SVM trained this research (blue) and trained by [16] (orange). X axis means FPR and Y axis means TPR. The point size means the F_1 score.



(b) The comparison between plain baseline model and baseline model integrated with topic model. X axis means FPR and Y axis means TPR. The point size means the precision score rank. The point color means the AUC rank.

Figure 3: The comparison between all baseline models.

This research set 1 layer, 2 layers, 4 layers spatial-based GNN models together to validate whether information diffusion improves the result. Those three models means the central node could receive the information from 1st order, 2nd order, 4th order neighbor nodes respectively. With number of layers increases and information diffusion strengthens, the result shows that 4-layers spatial-based GNN achieves the best AUCs core and precision. It explains the high order neighbor nodes do affect the central nodes.

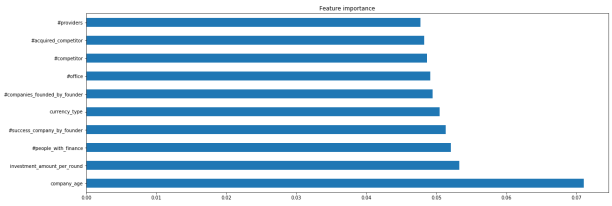


Figure 4: The feature importance of XGBoost algorithm. XGBoost algorithm sorts company age as the most important features, which is the most related to acquisition.

Index	Model	TPR	FPR	AUC	F_1 score	precision
1	spectral: 2 layers	76.2%	12.4%	0.885	0.556	0.917
2	spatial: 0 layer	84.4%	14.0%	0.919	0.565	0.817
3	spatial: 2 layers	87.8%	15.5%	0.937	0.572	0.841
4	spatial: 4 layers	81.5%	7.0%	0.943	0.683	0.890
5	spatial: 4 layers, 5-topics	82.0%	7.3%	0.935	0.689	0.930
6	spatial: 4 layers, 5-topics, inverse edge	86.5%	11.3%	0.952	0.638	0.946
7	GAT: 1 layers, 5-topics, inverse edge	82.0%	9.3%	0.938	0.646	0.900

Table 5: Graph Neural Network performance table.

Index	Dataset	Dropout	Node dimension	Edge dimension
1	$\mathbf{A}, \mathbf{X}^{all}, \mathbf{Y}^{graph}, \mathbf{M}$	0	[100,100,2]	[]
2	$\mathbf{E}^{di}, \mathbf{X}^{all}, \mathbf{Y}^{graph}, \mathbf{M}$	0.2	[2]	[2]
3	$\mathbf{E}^{di}, \mathbf{X}^{all}, \mathbf{Y}^{graph}, \mathbf{M}$	0.2	[200,100,2]	[100,50,50]
4	$\mathbf{E}^{di}, \mathbf{X}^{all}, \mathbf{Y}^{graph}, \mathbf{M}$	0.2	[80,80,80,80,2]	[30,30,30,30,50]
5	$\mathbf{E}^{di}, \mathbf{X}^{all}_{topic}, \mathbf{Y}^{graph}, \mathbf{M}$	0.2	[80,80,80,80,2]	[30,30,30,30,50]
6	$\mathbf{E}^{multi}, \mathbf{X}^{all}_{topic}, \mathbf{Y}^{graph}, \mathbf{M}$	0.2	[80,80,80,80,2]	[30,30,30,30,50]
7	$\mathbf{E}^{multi}, \mathbf{X}^{all}_{topic}, \mathbf{Y}^{graph}, \mathbf{M}$	0	[40,2]	[20,50]

Table 6: Graph Neural Network hyper parameters setting.

The above experiment only explores single direction information diffusion, which information is allowed to flow from neighbor nodes to central node but not allowed to flow from central node to neighbor nodes. Therefore, inversed edge mentioned is applied. Compared with Model 5, Model 6 with inverse edge improves both of AUC score and precision. Then inverse edge could enhance and speed up the information diffusion. In addition, it is observed that Model 5 with topic model does not improve the AUC score from Model 4. It is proved repeatedly that topic model cannot benefit the performance and sometimes hurts the performance.

In the Model 7, due to limitation of computation resource and small scale dataset, its performance is close to the plain spatial-based GNN model. In fact, the result is reasonable that attention mechanism does not bring new information. Observing the Table 5, every time performance boost is caused by adding new information. It adds neighbors node information from baseline to spectral-based GNN model, adds edge information from spectral-based GNN model to spatial-based GNN, adds higher order neighbors node information from Model 2 to Model 4, brings the inverse direction neighbor node information from Model 5 to Model 6. Among those influence factor, the edge information from spectral-based GNN model to spatial-based GNN increases the highest score.

In the dataset, 23.9% company entities do not have any neighbors. The GNN improves the rest 76.1% company entities. 89.2% company entities has less than 10 neighbors, which most of the edges are inverse edge with no information. The validate edge amount for each company entities is only 1.95. Although the average edge amount is not adequate, edge information and inverse edge still boost significant performance. The neighbor node does not boost too much score as expected. The main reason is that most of neighbor nodes are non-company entities. According to Data Chapter, except company entities, there are low quality data for other entities. Hence, with the low quality data, the neighbor nodes cannot offer enough effective information.

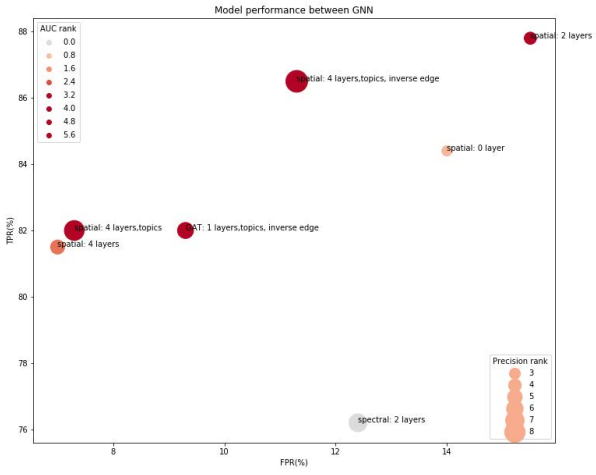


Figure 5: The comparison between all GNN models. X axis means FPR and Y axis means TPR. The point size is the precision score rank. The point color means the AUC rank.

In general, GNN Model 6 achieves the best performance in both of AUC score and precision. Although SVM obtains the same level precision, it gives up the recall rate (TPR). Compared with SVM, GNN has a higher TPR, which provides the investor with a wider choices.

7 CONCLUSION

This research firstly selects AUC score and precision as main metrics and explains why other metrics is not reliable to measure the model. Then list the performance of 6 baseline models and 7 GNN models. Among those models, the baseline XGBoost achieves benchmark level performance, and all GNN models raises the varying degrees of AUC scores as expected. Based on experiment result, it concludes: • Topic model based on company article cannot boost the performance. • GNN architecture helps company entities prediction by adding other entities information. • Neighbor node information, edge information, information diffusion, and inverse edge are all helpful to aggregate new information on company entities. • In this dataset and this task, edge information is the most essential influence factor.

REFERENCES

[1] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, H. Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey R. Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Manfred Otto Heess, Daan Wierstra, Pushmeet Kohli, Matthew Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. 2018. Relational inductive biases, deep learning, and graph networks. *ArXiv* abs/1806.01261 (2018).

[2] Peter W Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. 2016. Interaction networks for learning about objects, relations and physics. *arXiv preprint arXiv:1612.00222* (2016).

[3] Siheng Chen, Rohan Varma, Aliaksei Sandryhaila, and Jelena Kovačević. 2015. Discrete Signal Processing on Graphs: Sampling Theory<? Pub _newline=""? *IEEE transactions on signal processing* 63, 24 (2015), 6510–6523.

[4] Xinyu Chen, Zhaocheng He, and Lijun Sun. 2019. A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation. *Transportation research part C: emerging technologies* 98 (2019), 73–84.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. 2017. Protein Interface Prediction using Graph Convolutional Networks. *Advances in Neural Information Processing Systems* 30 (2017), 6530–6539.

[7] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *ArXiv* abs/1502.03167 (2015).

[8] Zhijian Li, Xiyang Luo, Bao Wang, Andrea L Bertozzi, and Jack Xin. 2019. A study on graph-structured recurrent neural networks and sparsification with application to epidemic forecasting. In *world congress on global optimization*. Springer, 730–739.

[9] Aliaksei Sandryhaila and José MF Moura. 2013. Discrete signal processing on graphs. *IEEE transactions on signal processing* 61, 7 (2013), 1644–1656.

[10] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine* 30, 3 (2013), 83–98.

[11] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15 (2014), 1929–1958. http://jmlr.org/papers/v15/srivastava14a.html

[12] Shayan Tabe-Bordbar, Amin Emad, Sihai Dave Zhao, and Saurabh Sinha. 2018. A closer look at cross-validation for assessing the accuracy of gene regulatory networks and models. *Scientific reports* 8, 1 (2018), 1–11.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *NIPS*.

[14] David H Wolpert and William G Macready. 1997. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation* 1, 1 (1997), 67–82.

[15] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

[16] Guang Xiang, Zeyu Zheng, Miaomiao Wen, Jason Hong, Carolyn Rose, and Chao Liu. 2012. A supervised approach to predict company acquisition with factual and topic features using profiles and news articles on techcrunch. In *Sixth International AAAI Conference on Weblogs and Social Media*.

[17] Kun Xu, Lingfei Wu, Zhiguo Wang, Yansong Feng, Michael Witbrock, and Vadim Sheinin. 2018. Graph2seq: Graph to sequence learning with attention-based neural networks. *arXiv preprint arXiv:1804.00823* (2018).

[18] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems* 32 (2019).

[19] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81.