Comparing the Impact of Financial Knowledge Graphs from Financial Reports and Wikidata in Asset Recommendation

Lubingzhi Guo, **Javier Sanz-Cruzado**, **Richard McCreadie**
University of Glasgow
l.guo.1@research.gla.ac.uk,{javier.sanz-cruzadopuig, richard.mccreadie}@glasgow.ac.uk

Abstract

Financial asset recommender (FAR) systems suggest investment assets to customers based on past market information. Many of these models choose those securities which they estimate to be more profitable for customers. Financial knowledge graphs (KGs)– data structures containing information about assets and their relations to other involved entities (companies, people) – have been one of the data sources exploited to drive asset selection. Although the construction of knowledge graphs from different sources (news, reports) has previously been investigated, there has been limited analysis of the effect these construction strategies have for FAR. In this work, we compare two different knowledge graphs representing U.S. stocks under a unified FAR framework: a knowledge graph crawled from a general knowledge base, Wikidata, and a knowledge graph built by extracting entities and relations from 10K financial reports using the GoLLIE open information extraction model. We show that integrating these KGs in FAR can lead up to 10.7% improvements in monthly ROI. However, the nature of these graphs makes algorithms prone to bias the recommendations towards different asset types.

1 Introduction

Financial asset recommender systems are tools to assist investors in making informed investment decisions [McCreadie *et al.*, 2022; Sanz-Cruzado *et al.*, 2022]. These technologies aim to produce a ranking of financial securities (e.g. stocks) on which a customer might invest. As one of the main targets of these methods is to help customers increase their wealth, the majority of the methods proposed in this field rely on historical pricing information of assets to predict stock price movement [Zhang *et al.*, 2017; Nelson *et al.*, 2017]. This core signal is then often augmented through the integration of evidence from external information sources such as textual data from news and social media [Hu *et al.*, 2018; Chen, 2021; Cheng *et al.*, 2020].

A *financial knowledge graph* (KG) is a data structure that can be used to store such external information [Wang *et al.*, 2023; Deng *et al.*, 2019; Cheng *et al.*, 2020; Zhao *et al.*, 2023]. In such a graph, nodes represent entities (companies, people), while edges represent relations between them (e.g. between-company or board member relationships). Multiple methods have been proposed for the creation of financial KGs, including crawling general knowledge bases like Wikidata [Feng *et al.*, 2019] or extracting facts and events from financial reports [Kertkeidkachorn *et al.*, 2023; Pujara, 2017] or news [Cheng *et al.*, 2020; Elhammadi *et al.*, 2020].

Different KG creation techniques have advantages and disadvantages. On the one hand, graphs extracted from financial documents (such as news or financial reports) include timely information about companies and related entities, which is specific to the financial domain. However, they are generated by applying information extraction techniques over (typically) unstructured text documents [Bach and Badaskar, 2007], and as such are prone to hallucinating incorrect facts [Pejić Bach *et al.*, 2019]. On the other hand, graphs produced from general knowledge bases like Wikidata or DBPedia have their factoids assessed by human annotators to ensure their correctness [Vrandečić and Krötzsch, 2014], but often incorporate connections that are irrelevant for the financial domain and are updated infrequently. While both approaches have been tested in isolation previously, no prior works have quantitatively compared these two different strategies. This is an important gap, as intuitively the graphs produced by these two strategies are likely to benefit different types of assets being recommended, introducing a structured bias. For instance, general knowledge bases result in larger graphs with imbalanced coverage toward well-known/long standing companies. Meanwhile, news-based graphs are smaller and more focused on companies that are newsworthy.

Hence, in this work, we tackle the above gap by analysing the impact that these two knowledge graph construction strategies have when predicting the future profitability of U.S. stocks. First, we produce a financial knowledge graph containing information about companies from Wikidata. Second, we build a knowledge graph by applying automated information extraction techniques over 10K reports using large language models (LLMs) [Sainz *et al.*, 2024]. As required by the U.S. Securities and Exchange Comission, these annual reports disclose detailed financial performance and announcements about publicly traded companies to stock investors and unsurprisingly trigger immediate market responses [Griffin, 2003]. Finally, we compare the utility of these knowledge graphs under a unified profitability prediction framework integrating knowledge graph embeddings [Wang *et al.*, 2021] as features for the task.

Specifically, the primary contributions of this work are three-fold:

- We construct a financial KG from 10K reports using fine-tuned LLMs for open information extraction.
- We crawl a financial KG from Wikidata as a general knowledge base.
- We compare the effect these knowledge graphs have on profitability prediction over the U.S. stock market, demonstrating that integrating these KGs can lead up to 10.7% higher in monthly ROI. We also demonstrate that different KG construction strategies bias their results towards separate sets of assets.

2 Related Work

To integrate external evidence into an asset recommendation system using a knowledge graph as an intermediate representation, we need two technologies: 1) a graph generator, that takes information about financial topics and extracts associated entities as well as their relations that form the graph; and 2) an entity embedder, which given an asset produces a vector embedding that encodes information related to the asset from the graph. We introduce past works regarding each below:

2.1 Knowledge Graph Generation

The first step in knowledge graph generation is to select the type of content that you want to extract information from. If producing a new knowledge graph from scratch, the most popular data source to use is financial news articles, as intuitively significant events that affect an asset will have associated news content, but conversely much irrelevant information will also be captured [Elhammadi *et al.*, 2020]. Instead, a clean data source such as financial filings can be used, which is more targeted and in-depth, but are published infrequently [Pujara, 2017; Kertkeidkachorn *et al.*, 2023]. Alternatively, rather than building a completely new graph, some works have bootstrapped from an existing knowledge-base, such as DBPedia, where a large general knowledge graph needs to be filtered down to a useful subset for the financial domain [Vrandečić and Krötzsch, 2014]. To contain the scope for this initial work, we compare approaches to model what a company (that can be invested in) *is and does*. As such, we use U.S. 10K financial reports as a company provided overview of their operations and compare it to company information from WikiData.

Once we have selected our data sources, we next need to extract the entities and relationships that will form each graph. Depending on the type of data being used, the approach here will differ. If using an existing knowledge graph, then a set of filtering rules needs to be defined, as well as potentially entity disambiguation performed. However, for text-based data sources, Information Extraction (IE) techniques need to be applied to the raw text. This is usually comprised of two components: 1) entity identification (and linking), which identifies financial entities (companies, people, places) in the text; and 2) relationship extraction, which generates likely relationship tags between pairs of entities [Pujara, 2017][1]. For instance:

$$\text{Entity1}_{\text{`Nik Jhangiani'}}, \text{Relationship}_{\text{`CFO'}}, \text{Entity2}_{\text{`Diageo'}} \quad (1)$$

Of note is that for relationship extraction, approaches can be either closed domain (a set of target relationship types are defined beforehand) or open domain (any relationship tag can be generated) [Kaur *et al.*, 2023]. While most works focus on closed-domain extraction, the emergence of effective large language models has opened the door to less error-prone open-domain extraction than was previously possible, with models such as GoLLIE [Sainz *et al.*, 2024]. In this work, we use GoLLIE over our 10K filings perform open-domain extraction, where the model is guided to look for either business, transaction or personnel-related relationships.

2.2 Entity Embedding

Once we have a financial knowledge graph, given a financial asset representing a company that we want to recommend, we need to produce an embedding representing what the knowledge graph has about that company. To do this, knowledge graph embedding (KGE) models are used, which produce a low-dimensional vector representation given a starting graph node or edge to represent [Wang *et al.*, 2021]. There are three families of KGE models:

Translation-based These techniques represent entities as points and relationships as translations in vector spaces. An early model in this category is TransE [Bordes *et al.*, 2013], followed by models like TransH [Rossi *et al.*, 2021] and TransR [Lin *et al.*, 2015] and RotatE [Sun *et al.*, 2018].

Factorization-based Factorization methods estimate the plausibility of triplets through semantic similarity, focusing on the latent semantics between entities and relations [Rossi *et al.*, 2021]. These models include RESCAL [Nickel *et al.*, 2011], DistMult [Yang *et al.*, 2015], TuckER [Balazevic *et al.*, 2019], and HolE [Nickel *et al.*, 2016].

Neural Network-based Neural networks, with their capacity to learn complex patterns through a large number of parameters, are considered as a promising approach across various domains [Rossi *et al.*, 2021]. Several KGE methods have taken these algorithms as a basis, like ConvE [Dettmers *et al.*, 2018], RGCN and KGAT [Wang *et al.*, 2019].

In our later experiments, we compare asset embeddings produced by a range of algorithms across these three types for both 10K filings and Wikidata-based knowledge graphs.

2.3 Financial Asset Recommendation

Finally, having produced a knowledge graph embedding for a company/asset, we can then use that embedding to augment a downstream task. In this work, we target Financial Asset Recommendation as that task, where given a day, we want to rank assets on that day such that the future return-on-investment of the top ranked assets is maximised [Feng *et al.*, 2019;

[1]Relationships may also have extracted properties/qualifiers, such as an indicated date for when the relationship was formed.

Alzaman, 2024; Alsulmi, 2022]. For this, we rely on regression models like the ones used by [Sanz-Cruzado *et al.*, 2022; Rather *et al.*, 2015]

Similarly to our work, several models have integrated pricing and knowledge graph information for stock predictions [Feng *et al.*, 2019; Zhang *et al.*, 2018]. These models either exploit similarities between assets [Zhang *et al.*, 2018; Long *et al.*, 2020; Wang *et al.*, 2023] or integrate KGs as features [Deng *et al.*, 2019; Cheng *et al.*, 2020; Zhao *et al.*, 2023]. We explore the second way. However, previous feature-based approaches need specific KG structures or data sources to build those KGs. Differently, we propose a simple framework which integrates knowledge graph embeddings as features. This allows the use of any financial KG as input to our models – something that we can use to compare the effect that different knowledge graph structures have on the recommendations.

3 Knowledge Graph Construction

In this work, we construct two financial knowledge graphs from two different sources to compare the performance for financial asset recommendation / stock recommendation.

3.1 Graph Definition

Our constructed knowledge graphs contain a set of hyper-relational facts $f = (e_h, r, e_t, q, q_v)$, following the definition provided by Chia et al. [2022]. Each fact consists of a head entity e_h, a relation r , a tail entity e_t, as well as an optional qualifier label q and its respective value q_v. For instance, specific organizations and people represent entities, relations detail connections between those entities (i.e. ownership or employment) and qualifiers represent additional values associated to a link (for example, the date on which the link occurs).

3.2 Wikidata graph

Firstly, we extract a financial knowledge graph from a general knowledge base, Wikidata [Vrandečić and Krötzsch, 2014]. Wikidata includes information about entities in the financial domain that we can integrate into a graph. However, as Wikidata contains a broad range of information beyond the financial domain, we need to filter and retrieve the relevant data for our knowledge graph, following the procedure we detail next.

Seed Entity Matching As a first step, we need to identify some seed entities in Wikidata. For this, we take the stocks trading in NASDAQ, NYSE and AMEX at December 2021 as our seed entities, since we aim to predict the future pricing of these stocks. We employ a semi-automated three-step process to match those companies with entities in the Wikidata knowledge base:

First, we use the SPARQL Wikidata Query Service[2] to filter and retrieve entities related to the stock exchanges of interest (NASDAQ, NYSE, AMEX) and gather their identifiers, names, and aliases in multiple languages. If a direct match between the company tickers and Wikidata entries is found, we link them automatically. Second, for assets without direct matches, we use DBpedia Spotlight [Mendes *et al.*, 2011] for entity recognition and linking to entries in DBPedia (another public knowledge base), which are then cross-referenced to Wikidata identifiers. We verify the results manually to ensure accuracy. Finally, for any unmatched entities, we conduct a manual search in Wikidata. If a company does not match any entity, it is excluded from our dataset, assuming no association with Wikidata exists. We started with a total of 5,823 assets from NASDAQ, NYSE and AMEX. Via entity matching we mapped 3,370 of these (57.9%) to Wikidata pages.

[2]https://query.wikidata.org/

Element	Valid types
Entity	Organization, Person, Location, Market, Product, Material, Activity, Award, Legal form, Form of government, Gender, Health problem
Relation	Ownership, Employment, Part of, Creation, Award, Location, Material, Skills, Condition, Sequence
Qualifier	Time, Position, Location, Item or service, Amount

Table 1: Entity, relation and qualifier types in the Wikidata graph.

Entity and Relation Extraction Starting with mapped Wikidata entries, we extract metadata, relations, and properties for each entry, including any available temporal information for the relations. Our crawling method follows a breadth-first search algorithm, beginning with seed entities and expanding outward in a first-found, first-served manner. To avoid crawling information outside the financial domain, we cap the search depth from the seed entities. Furthermore, we have specified a list of valid financial relations and entity types to guide our crawler. The broad types of those entities and relations are summarized in Table 1.

3.3 10K reports graph

Second, we create a graph from financial texts using automated relation extraction. In this work, we construct our automated knowledge graph using some of the most comprehensive and official financial reports: the 10K annual filings. We next provide details of our information extraction procedure.

Pre-processing Considering that financial reports are lengthy and exceed the context window of the LLM we use, we initially perform sentence segmentation on each report, converting it into a list of sentences using Stanford NLP [Qi *et al.*, 2020]. This enables us to generate hyper-relational facts. Furthermore, as in most reports, terms such as 'we', 'the company' and 'the corporate' refer to the reported company, as such we resolve/replace these with their corresponding company names.

Entity and Relation Extraction Our information extraction pipeline is based on GoLLIE [Sainz *et al.*, 2024]. GoLLIE is a recent LLM-based model for zero-shot information extraction (IE). This model has been successfully applied to multiple IE tasks across multiple domains, so we use it in our work to extract financial entities and relations from the 10K filings. This model is based on Code-LLaMA [Roziere *et al.*, 2023] and represents both input and output using Python

Entity	Valid Type
Head Entity & Tail Entity	Organization, Person
Qualifier Label	Time, Position, Location, Item or service, Amount

Table 2: Entity types in the 10K graph.

classes. It receives two inputs: first, a text from which to extract information, a second, a list of Python class definitions describing the information to extract. For each label, there is a Python class detailing its structure (where class attributes represent specific information pieces to extract). Extraction guidelines are embedded as comments in the Python class to assist the model. The output of the model is a list of instances of the pre-defined Python classes, each containing a pair of entities, a relation and any relevant properties.

In our information extraction procedure, we perform event argument extraction to produce hyper-relational facts using event templates provided for GoLLIE[3]. We define three types of events to extract:

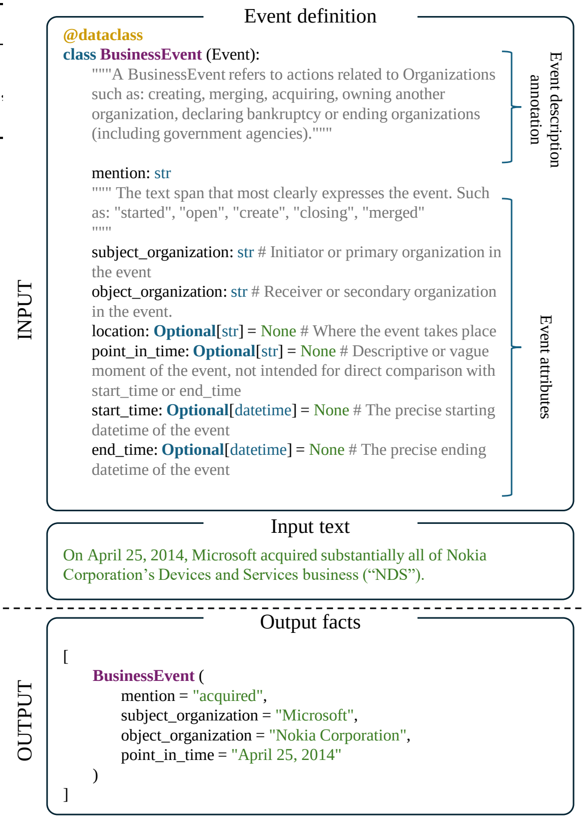
- **Business event:** actions related to organisations. For instance, creating, acquiring or ending other organisations as well as declaring bankruptcy.
- **Transaction event:** this event type refers to exchanges between organisations, either of artefacts, people or money.
- **Personnel event:** interactions between people and organisations. For instance, foundation of a company, or election of a person for a position.

All of these events are represent by a hyper-relational fact, where we define the types of the entities, and allow the model to extract the relation types. We broadly classify the qualifiers into five groups: ‘time’, ‘position’, ‘location’, ‘item or service’ and ‘amount’.

Figure 1 shows an example of the input and output of the model. In the predefined ’BussinessEvent’ data class, we first provide a text describing the types of events we want to extract as a comment. In order to capture the business facts related to organizations, we define the head entity as ’subject organization’, and tail entity as ’object organization’, adding location and time as qualifiers. As it is common that the extracted facts may not always include all the components we thus define Optional typings to handle diverse matching patterns. This example demonstrates an extracted ’BusinessEvent’ instance from the input text, with ’Microsoft’ as the head entity and ’Nokia Corporation’ as the tail entity. The relationship between them is described by the term ’acquired’, with ’April 25, 2014’ serving as the only time-related qualifier.

Relation Clustering The identified relational phrases are continuous text spans directly extracted from the sentence, which tend to be noisy. Therefore, additional steps are needed to match different relation types. Inspired by Hu et al. [2020], we use clustering to group similar relational facts in an unsupervised manner.

[3]https://github.com/hitz-zentroa/GoLLIE/blob/main/notebooks/Event%20Extraction.ipynb



```
@dataclass
class BusinessEvent (Event):
    """A BusinessEvent refers to actions related to Organizations
    such as: creating, merging, acquiring, owning another
    organization, declaring bankruptcy or ending organizations
    (including government agencies)."""

    mention: str
    """ The text span that most clearly expresses the event. Such
    as: "started", "open", "create", "closing", "merged"
    """
    subject_organization: str # Initiator or primary organization in
    the event
    object_organization: str # Receiver or secondary organization
    in the event.
    location: Optional[str] = None # Where the event takes place
    point_in_time: Optional[str] = None # Descriptive or vague
    moment of the event, not intended for direct comparison with
    start_time or end_time
    start_time: Optional[datetime] = None # The precise starting
    datetime of the event
    end_time: Optional[datetime] = None # The precise ending
    datetime of the event
```

On April 25, 2014, Microsoft acquired substantially all of Nokia Corporation’s Devices and Services business (“NDS”).

```
[
    BusinessEvent (
        mention = "acquired",
        subject_organization = "Microsoft",
        object_organization = "Nokia Corporation",
        point_in_time = "April 25, 2014"
    )
]
```

Figure 1: An example of extracted hyper-relational facts

We first perform lemmatization on the extracted relations to reduce variations in their word representation. To perform the clustering, we represent the relation mentions as vectors using the Sentence-BERT pretrained language model [Reimers and Gurevych, 2019]. This enables us to establish distances between relation phrases, which we can use to perform the clustering. Then, we use agglomerative clustering [Murtagh and Legendre, 2014] to group the vectors and combine similar relations – thus shrinking the number of different relations in our knowledge graph. We use this simple and hierarchical algorithm as it allows us to establish a distance threshold for grouping instead of a number of clusters – something that aligns well with our open information extraction task. In our configuration, we consider that two vectors with distance (within-cluster variance) smaller than 1 belong to the same cluster. Finally, to further refine our dataset with higher qualified relations, we exclude relations that occurred with a frequency below the 75th percentile threshold for the entire collection (6 times in our data).

Entity Linking Similar to extracted relations, the entities extracted from texts are text spans that need to be unified and linked to real-world entities. Given the difficulty in accurately mapping individual names, we only focus on organisations. We apply two methods. First, we use a zero-shot entity linking method based on BERT known as BLINK [Wu *et al.*, 2020]: this approach matches spans of text to entities in Wikipedia. Second, we use the company name nor-

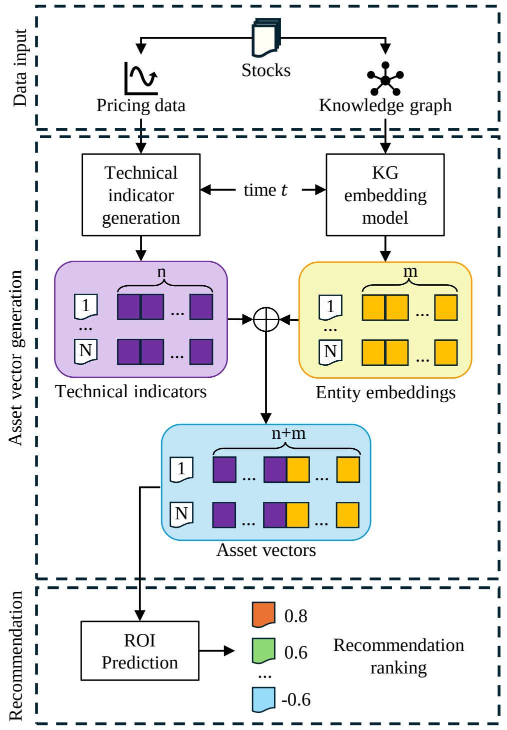


Figure 2: Profitability Prediction Architecture

malisation functionality of the John Snow Labs NLP library[4] This feature maps extracted company names to the name registered with the SEC in the Edgar Database, which is useful when handling 10K filings and aims to enhance the accuracy of linking organisational entities.

Both methods provide, as outputs, an entity name and a confidence score. To ensure reliability of the collected entities, for each method, we only keep those text-entity pairs with confidence scores above the median score obtained for all the analysed text spans. We keep the matching if the confidence score for one of the two methods is above the median for the entity linker. In case we have positive matchings for a company in both methods, we keep the one provided by BLINK.

4 Profitability Prediction with Knowledge Graph Embeddings

After constructing our financial knowledge graphs, we aim to use them to improve price prediction accuracy, where future predicted prices are used to recommend financial assets. We therefore define a simple aggregation framework that enables us to combine knowledge graph information with temporal pricing information, depicted in Figure 2.

Asset vector generation For each time point t, we remove those facts in the KG happening after t, and feed the remaining graph into a knowledge graph embedding (KGE) model to obtain entity embeddings for specific financial assets targeted for prediction. KGE models encode the information

[4]https://github.com/JohnSnowLabs/johnsnowlabs

Property	Wikidata	10K
Number of entities	102,739	8,380
Number of relation types	114	450
Number of links	457,758	36,973

Table 3: Graph Properties

in the knowledge graph into a compact, low dimensional space, while preserving the important properties of the graph [Cai *et al.*, 2018]. These embeddings consider information about each entity and its relationships with other entities in the graph.

Separately, for the same time point t, the historical price data is processed to calculate the technical indicator vector for each asset/stock. Technical indicators, also referred to as key performance indicators (KPIs), are heuristics that encode signals from past pricing information of a financial asset. These indicators have been shown to be beneficial in predicting future profitability and are widely used in training price prediction models [Neely *et al.*, 2014; Sanz-Cruzado *et al.*, 2022; Naik and Mohan, 2019]. Notably, these asset vectors vary over time due to frequent updates in asset pricing and timely revisions of company information within the constructed graph.

Profitability prediction Once we have the vector representation for the assets at time t, we train a regression model to predict the future profitability of the assets and then rank them by their prediction in descending order. Specifically, given a prediction time t, we train a model with asset vectors from time points preceding t to avoid leaking future information. The target of the regression model is to predict the return on investment (ROI) over a specified interval Δt, i.e. the percentage change in closing price. As the loss function for our regression algorithms, we use the squared error.

5 Experiment setup

To assess the effectiveness of the two constructed knowledge graphs in predicting stock market profitability, we carry out experiments using U.S. stock market data. Here, we detail the dataset and the experimental setup used for our analysis.

5.1 Dataset

To conduct our experiments, we collected a dataset from three major U.S. stock exchanges: NASDAQ, NYSE, and AMEX.

Pricing data: We collect daily pricing data from Yahoo! Finance[5] including open, close, high, low, and volume prices for 5,823 assets from January 2018 to September 2022.

Wikidata graph: Using the approach described in Section 3.1, we collected 3,370 assets entities from Wikidata, resulting in more than 100k entities and 450k relations crawled for our knowledge graph. Table 3 summarises the total properties of the crawled Wiki graph.

10K graph: We successfully retrieved 2,264 assets with 10K reports[6] based on the linked assets in the Wikidata graph, re-

[5]http://finance.yahoo.com

[6]https://sec-api.io/

sulting in 5,399 filings from 2017-2022. We incorporated an additional year of data prior to 2018 to ensure a rich dataset for constructing the initial knowledge graph. Table 3 summarizes the global 10K graph properties.

Dataset split: For each time point, technical indicators and knowledge graph versions from prior dates are used as input to predict price changes over a six-month horizon (Δt). For our experiments, time points on or before the 31st of December 2019 are used for training and the six months following the 30st of June 2020 are used for testing, maintaining a six-month gap between those sets to avoid data leakage. Within the dataset, time points are spaced 1 week apart, selecting the Monday of each week as t. In total, the training set includes 73 time points, while the test set contains 25.

Dataset post-processing: To ensure data consistency and reduce discrepancies, we take the intersection of assets listed in the pricing data, Wikidata entities and 10K reports. This results in 2,042 assets for our training set and 2,096 assets for our testing set. We also exclude 421 upper outliers from our test set when profits exceed 1.5 times the interquartile range above the third quartile, which indicate unusually high profits. These outliers include penny stocks, companies coming back from bankruptcy, and phenomena like the 2021 meme stock trading (e.g., GameStop). Including these assets leads to unstable evaluations, as their presence among the top-ranked assets can significantly skew metrics like ROI@10.

5.2 Metrics

In order to evaluate our predictions, we consider two different metrics: 1) a ranking-oriented metric, monthly return on investment (ROI), and 2) a global error metric, root mean squared error (RMSE). We summarise each below:

- **Monthly return on investment (ROI@k):** we analyse the average monthly return on investment over the top k ranked assets. In our experiments, we take $k = 10$ and compute the ROI over the 6 months following the date of the recommendations.
- **Root mean-squared error (RMSE):** To understand the model accuracy, we compute the square root of the average squared difference between predicted and real ROI.

5.3 Model Configuration

Technical indicators: In our experiments, we use 16 different KPIs derived from the pricing time series as technical indicators, summarized in Table 4. In order to generalise the comparison of two knowledge graphs, and reduce the influence of KPIs on the comparison result, we have chosen two groups of technical indicators:

- **BasicKPIs**: average price, return on investment, and volatility.
- **AdvKPIs**: all KPIs in Table 4.

Knowledge graph embeddings: We select a set of both popular and state-of-art KGE models for the experiment, specifically, 9 KGE models are tested:

- **Translation-based embeddings**: TransE [Bordes *et al.*, 2013], TransH [Wang *et al.*, 2014], TransR [Lin *et al.*, 2015] and RotatE [Sun *et al.*, 2019]

Table 4: Summary of financial technical indicators

Indicator (financial days)	Time period Δt
Average price	28, 63, 126
Return on investment	28, 63, 126
Volatility	28, 63, 126
Momentum	14, 21, 28
Moving average convergence divergence	26
Rate of change	14, 21, 28
Relative strength index	14
Detrended close oscillator	22
Force index	1
Minimum	14, 21, 28
Maximum	14, 21, 28
Chaikin oscillator	10
Average true range	14
Average directional index	14
Vortex indicator	14

- **Factorization-based embeddings**: RESCAL [Nickel *et al.*, 2011], HolE [Nickel *et al.*, 2016] and TuckER [Balazevic *et al.*, 2019]
- **Neural network-based embeddings**: ConvE [Dettmers *et al.*, 2018] and RGCN [Schlichtkrull *et al.*, 2018]

We use the PyKeen library [Ali *et al.*, 2021] to generate 50-dimensional embeddings for entities (companies) associated with each asset. We repeat embeddings generation 5 different times to analyse the variability across the runs. However, we use just one random seed for the RGCN model due to its high computational cost.

Regression Model We opt to use a Random Forest regression algorithm with 100 trees as our prediction model.

6 Results

In this section we compare the impact of incorporating two distinct knowledge graphs sourced from Wikidata and 10K reports on the prediction of asset profitability. In particular, we investigate the following research questions:

- **RQ1:** How does the use of the Wikidata and 10K graphs affect the effectiveness of profitability prediction?
- **RQ2:** How different are the profitable assets recommended for each knowledge graph?

6.1 RQ1: Graph Performance Comparison

We begin by examining the core question posed in this work: how do knowledge graphs derived from financial reports vs. a knowledge base affect financial asset recommendation (FAR) effectiveness? In particular, we would like to know whether one knowledge graph provides more useful information than the other, and whether the approach used to embed the graph for each company impacts performance.

To answer this question, we compare FAR approaches with and without knowledge graph embeddings. In particular, we start with a price-prediction-based baseline referred as *Baseline (Only KPIs)*, which uses past pricing data to predict the future price of an asset. For a day, all assets are ranked by their predicted return-on-investment (RoI) after 6 months.

		BasicKPIs				AdvKPIs			
		ROI@10		RMSE		ROI@10		RMSE	
Group	Algorithm	10K Graph	Wikidata	10K Graph	Wikidata	10K Graph	Wikidata	10K Graph	Wikidata
Baseline (Only KPIs)		0.0410		0.4574		0.0466		0.4299	
Translation-based models	KPIs + TransE	**0.0454***	0.0368	0.4439†	0.4415†	0.0474*	0.0393	0.4277†	0.4289
	KPIs + TransH	0.0409	0.0419	0.4408†	0.4383†	0.0467	0.0434	0.4257†	0.4275
	KPIs + TransR	0.0435	0.0422	0.4442†	0.4385$^{\dagger *}$	0.0442	0.0454	0.4276†	0.4259†
	KPIs + RotatE	0.0427*	0.0385	0.4423†	0.4371†	0.0472*	0.0403	0.4254†	0.4256†
Factorization-based models	KPIs + RESCAL	0.0418	0.0418	0.4474†	0.4439$^{\dagger *}$	0.0451	0.0447	0.4329	0.4243$^{\dagger *}$
	KPIs + HolE	0.0418	0.0407	0.4448†	0.4353$^{\dagger *}$	0.0441	0.0442	0.4285	0.4239$^{\dagger *}$
	KPIs + TuckER	0.0400	0.0409	0.4442†	0.4334$^{\dagger *}$	0.0412	0.0445*	0.4298	0.4226$^{\dagger *}$
Neural network models	KPIs + ConvE	0.0407	0.0450*	0.4433†	0.4304$^{\dagger *}$	0.0435	0.0451	0.4269†	0.4207$^{\dagger *}$
	KPIs + RGCN	0.0423	0.0447	0.4479†	**0.4192**$^{\dagger *}$	0.0423	**0.0502***	0.4322	**0.4126**$^{\dagger *}$
Market		0.0345		-	-	0.0345		-	-
S&P 500		0.0266		-	-	0.0266		-	-

Table 5: Performance of random forest regression methods with assets embeddings derived from two knowledge graphs, when predicting six months into the future. Cell colours go from red (lower) to blue (higher values). Blue cells represent improvements over the baseline model (only using KPIs) whereas red cells do not improve it. The best value for each algorithm and metric is highlighted in bold. † denotes significant improvements (Wilcoxon test $p < 0.05$) with respect to the only KPIs baseline. * indicates significant improvements compared to the corresponding graph model for the other graph.

To evaluate performance, we report both error between the prediction and actual RoI (RMSE, lower is better) and the actual (monthly) RoI of the top 10 recommendations (RoI@10, higher is better). We have two baseline variants, denoted BasicKPIs and AdvKPIs, where the latter includes more technical indicators. As we can see from Table 5, the baseline models achieve an RoI@10 of 4.1% (BasicKPIs) to 4.66% (AdvKPIs), which is higher than both the market average (Market) and S&P 500 (a common index benchmark) for the same period (also reported at the bottom of Table 5).

Having established our baseline, we now contrast this baseline to the same model when augmented with the embeddings derived from our two knowledge graphs. In Table 5, for each metric, we include two columns (10K Graph and WikiData) reporting performance when the baseline is augmented by each knowledge graph. As there are a range of possible graph embedding techniques (see Section 2.2), we include one row for each embedding technique tested, denoted *KPIs + <KGE>* (where <KGE> is a knowledge graph embedding approach, e.g. TransE). For each column, cells are coloured from red (worst values) to blue (best values). Blue values indicate improvements with respect to the baseline, whereas red values indicate a decline in performance. The best metric values are highlighted in bold, and statistically significant increases (pairwise Wilcoxon test at $p < 0.05$) in comparison to the Baseline (Only KPIs) model is denoted $\dagger$. We also highlight significance differences between the application of the same model on the two knowledge graphs as $*$.

The first observation is that integrating KGE for profitability prediction generally results in RMSE reductions with respect to the baselines (33/36 times). In 30 cases, this reduction is significant, thus showcasing the capability of knowledge graph information to generate more accurate predictions. When comparing both graphs, the Wikidata KG obtains lower errors in 15 out of 18 cases (with 11 of them showing a significant difference) – therefore showing that this graph provides more accurate results than the 10K graph.

We observe a different pattern when we study the return on investment over the top-10 ranked results however: even when most methods using KGE reduce the prediction error, this fact does not necessarily result in more profitable recommendation rankings. This is particularly notable for the methods using the larger set of indicators, where only four models beat the baseline (TransE, TransH and RotatE for the 10k graph and RGCN for the Wikidata graph). However, for both baselines, it is possible to find at least one model for each graph improving its profitability. In the case of the BasicKPIs baseline, the best models are TransE for the 10K graph (4.54% ROI@10) and ConvE for the Wikidata graph (4.47% ROI@10). For AdvKPIs, TransE is again the best for the 10K graph (4.74% ROI@10), whereas RGCN is the best for Wikidata (5.02% ROI@10). This illustrates that both knowledge graphs are capable of providing a useful profitability signal for the task.

When we compare the effectiveness of the graphs in terms of ROI@10, we also see that there is a different relationship between the complexity of the embedding approach and ROI gain across the two graphs. Specifically, the 10k graph yields higher ROI for the translation-based algorithms (particularly the simpler TransE and RotatE models) that perform poorly when applied on Wikidata. Meanwhile, for the most complex of tested algorithms (TuckER, and both neural network approaches, ConvE and RGCN), the Wikidata graph provides a stronger profitability signal. According to Table 3, the Wikidata graph contains approximately ten times the number of entities and links as the 10K graph, indicating a greater complexity and graph size. Although the simple knowledge graph embedding models are capable of providing useful summaries of the 10k graph information, we hypothesize that the more complex knowledge graph embedding models (specially those based on neural networks) need a much larger number of links to learn how to extract stronger profitability signals from knowledge graphs – hence why RGCN performs well on Wikidata but not the 10K filings.

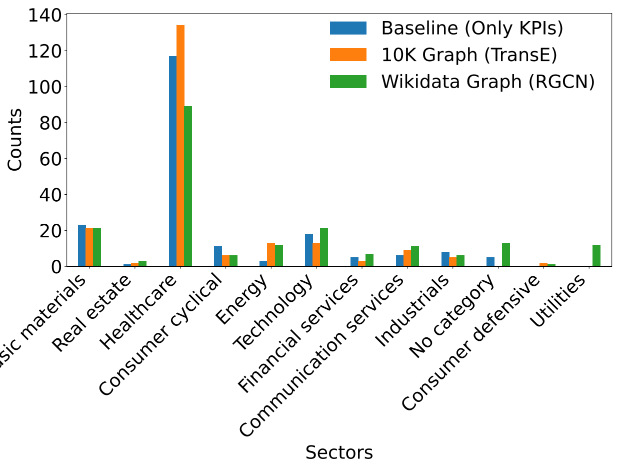


Figure 3: Distribution of profitable assets in the top-10 recommendation rankings across sectors.

To answer RQ1: *Both knowledge graphs are capable of enhancing the accuracy of the predictions – with Wikidata achieving better results. If we look at returns, however, it highly depends on the embedding method used. The simpler translation-based methods favour the use of the smaller 10K graph, whereas the most complex neural-based methods require more information to work, which they can obtain from the Wikidata graph.*

6.2 RQ2: Profitable Asset Sector Analysis

Besides raw algorithm performance, we hypothesize that different knowledge graph construction methodologies lead to the promotion of specific types of assets in the recommendations. For instance, general knowledge graphs might promote well-known companies as they have more information about them. As studying these differences is important to understand the inner workings of these methods, we provide a preliminary analysis where we study the distribution of recommended profitable assets across sectors.

To perform this analysis, we identify assets with positive ROI in the top-10 of the asset rankings and count how many times each sector is represented. We compare two top performing models using the basic indicators: TransE for the 10K graph, and RGCN for the Wikidata graph. Although ConvE provides slightly better performance when using the basic KPIs for the Wikidata graph, we choose RGCN as it is the best overall method for this KG. Both BasicKPIs + TransE (10k) and BasicKPIs + RGCN (WikiData) provide similar ROI@10 values (4.54% vs. 4.47%), but we hypothesise that source of that profitability might be different.

Figure 3 displays the results of our experiment, where the x axis shows the different sectors and the y axis shows the number of profitable assets for each algorithm and sector. In the plot, we also include the baseline using only technical indicators as features, for comparison. For many of the sectors (basic materials, consumer cyclical, energy), similar numbers of profitable assets are selected by both graphs, but there are sectors which highlight the differences between both knowledge bases.

The most important is the healthcare sector. In our data, the studied test period (June-December 2020) runs during the Covid-19 pandemic. Due to the pandemic, the value of healthcare in this period rose. This is observable in our results, as it is the sector counting the biggest number of profitable assets for the three compared models. However, it is the models using the 10K graph that recommends more assets from this sector. Considering that 10K filings contain company projection information, in this case, they enable the model to capture company outlooks on the Covid-19 pandemic and exploit them – something that the Wikidata graph, containing more general information, does not, as it even reduces the number of profitable healthcare recommended assets with respect to the baseline. Instead, the Wikidata graph takes its improvements from other sectors, like technology or utilities, for which the graph might contain more data.

To answer RQ2: *The knowledge graph construction strategy markedly impacts the types of assets recommended and this appears to be driven by the types of relationships and properties captured within each graph, although further investigation will be needed to conclusively show this.*

7 Conclusion & Future Work

In this work, we have explored the impact that two KG construction strategies have when predicting the future returns of U.S. stocks. For this, we collected a Wikidata subgraph and a built a graph by automatically extracting factoids from annual 10K filings. We have compared these methods under a unified FAR model that estimates the profitability of stocks. This method integrates price technical indicators with asset KG vectors extracted from the graphs by 9 different knowledge graph embedding models.

Our findings show that both graph types can improve the profitability of recommendations with respect to only using price information by up-to 10.7%. However, different graphs favour different embedding strategies: graphs extracted from financial reports tend to be smaller, and therefore benefit from translation-based models like TransE [Bordes *et al.*, 2013] or RotatE [Sun *et al.*, 2019], whereas the bigger Wikidata graph favours complex neural network models like RGCN [Schlichtkrull *et al.*, 2018].

We have also analysed the distribution of the profitable assets recommended by the models across sectors, showing that different knowledge graph construction strategies might present biases towards certain types of assets. In our experiments, the 10K graph has been able to leverage the information regarding global events (in particular, the Covid-19 pandemic) available in the reports to promote profitable healthcare stocks, while the more static Wikidata graph has identified profitable assets in sectors like utilities.

As future work, we aim to compare these knowledge graphs with others that include other types of financial information, such as news or press releases. We also aim to further analyse the properties of the assets recommended by different graphs, so we can prevent potential undesired algorithmic behaviours. Finally, as in this work we have only used random forests, we aim to test other FAR algorithms, including those directly targeting asset ranking [Alsulmi, 2022].

References

[Ali *et al.*, 2021] Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings. *Journal of Machine Learning Research*, 22(82):1–6, 2021.

[Alsulmi, 2022] Mohammad Alsulmi. From Ranking Search Results to Managing Investment Portfolios: Exploring Rank-Based Approaches for Portfolio Stock Selection. *Electronics*, 11(23):4019, 2022.

[Alzaman, 2024] Chaher Alzaman. Deep learning in stock portfolio selection and predictions. *Expert Systems with Applications*, 237:121404, 2024.

[Bach and Badaskar, 2007] Nguyen Bach and Sameer Badaskar. A review of relation extraction. *Literature review for Language and Statistics II*, 2:1–15, 2007.

[Balazevic *et al.*, 2019] Ivana Balazevic, Carl Allen, and Timothy Hospedales. TuckER: Tensor Factorization for Knowledge Graph Completion. In *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 5185–5194, Hong Kong, China, 2019. Association for Computational Linguistics.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In *27th Conference on Neural Information Processing Systems (NeurIPS 2013)*, Stateline, Nevada, USA, 2013. Curran Associates, Inc.

[Cai *et al.*, 2018] HongYun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1616–1637, 2018.

[Chen, 2021] Qinkai Chen. Stock movement prediction with financial news using contextualized embedding from BERT. *CoRR*, abs/2107.08721, 2021.

[Cheng *et al.*, 2020] Dawei Cheng, Fangzhou Yang, Xiaoyang Wang, Ying Zhang, and Liqing Zhang. Knowledge Graph-based Event Embedding Framework for Financial Quantitative Investments. In *43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2020)*, pages 2221–2230, Online, China, 2020. ACM.

[Chia *et al.*, 2022] Yew Ken Chia, Lidong Bing, Sharifah Mahani Aljunied, Luo Si, and Soujanya Poria. A dataset for hyper-relational extraction and a cube-filling approach. In *2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022)*, pages 10114–10133, Abu Dhabi, 2022. Association for Computational Linguistics.

[Deng *et al.*, 2019] Shumin Deng, Ningyu Zhang, Wen Zhang, Jiaoyan Chen, Jeff Z. Pan, and Huajun Chen. Knowledge-Driven Stock Trend Prediction and Explanation via Temporal Convolutional Network. In *The Web Conference 2019 (WWW 2019 Companion)*, pages 678–685, San Francisco, CA, USA, 2019. ACM.

[Dettmers *et al.*, 2018] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2D Knowledge Graph Embeddings. In *32nd of the AAAI Conference on Artificial Intelligence (AAAI 2018)*, pages 1811–1818, New Orleans, LA, USA, 2018. AAAI Press.

[Elhammadi *et al.*, 2020] Sarah Elhammadi, Laks V.S. Lakshmanan, Raymond Ng, Michael Simpson, Baoxing Huai, Zhefeng Wang, and Lanjun Wang. A High Precision Pipeline for Financial Knowledge Graph Construction. In *28th International Conference on Computational Linguistics (COLING 2020)*, pages 967–977, Online, Barcelona, Spain, 2020. International Committee on Computational Linguistics.

[Feng *et al.*, 2019] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. Temporal Relational Ranking for Stock Prediction. *ACM Transactions on Information Systems*, 37(2):1–30, 2019.

[Griffin, 2003] Paul A Griffin. Got information? investor response to form 10-k and form 10-q edgar filings. *Review of Accounting Studies*, 8:433–460, 2003.

[Hu *et al.*, 2018] Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. Listening to Chaotic Whispers: A Deep Learning Framework for News-oriented Stock Trend Prediction. In *11th ACM International Conference on Web Search and Data Mining (WSDM 2018)*, pages 261–269, Los Angeles, CA, USA, 2018. ACM.

[Hu *et al.*, 2020] Xuming Hu, Lijie Wen, Yusong Xu, Chenwei Zhang, and Philip Yu. SelfORE: Self-supervised relational feature learning for open relation extraction. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 3673–3682, Online, November 2020. Association for Computational Linguistics.

[Kaur *et al.*, 2023] Simerjot Kaur, Charese Smiley, Akshat Gupta, Joy Sain, Dongsheng Wang, Suchetha Siddagangappa, Toyin Aguda, and Sameena Shah. REFinD: Relation Extraction Financial Dataset. In *the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2023)*, pages 3054–3063, Taipei, Taiwan, 2023. ACM.

[Kertkeidkachorn *et al.*, 2023] Natthawut Kertkeidkachorn, Rungsiman Nararatwong, Ziwei Xu, and Ryutaro Ichise. FinKG: A Core Financial Knowledge Graph for Financial Analysis. In *17th IEEE International Conference on Semantic Computing (ICSC 2023)*, pages 90–93, Laguna Hills, CA, USA, 2023. IEEE.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *29th AAAI Conference on Artificial Intelligence (AAAI 2015)*, pages 2181–2187, Austin, TX, USA, 2015. AAAI Press.

[Long *et al.*, 2020] Jiawei Long, Zhaopeng Chen, Weibing He, Taiyu Wu, and Jiangtao Ren. An integrated framework of deep learning and knowledge graph for prediction of stock price trend: An application in Chinese stock exchange market. *Applied Soft Computing*, 91:106205, 2020.

[McCreadie *et al.*, 2022] Richard McCreadie, Konstantinos Perakis, Maanasa Srikrishna, Nikolaos Droukas, Stamatis Pitsios, Georgia Prokopaki, Eleni Perdikouri, Craig Macdonald, and Iadh Ounis. Next-Generation Personalized Investment Recommendations. In John Soldatos and Dimosthenis Kyriazis, editors, *Big Data and Artificial Intelligence in Digital Finance: Increasing Personalization and Trust in Digital Finance using Big Data and AI*, pages 171–198. Springer, 2022.

[Mendes *et al.*, 2011] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. DBpedia Spotlight: Shedding Light on the Web of Documents. In *7th International Conference on Semantic Systems (I-Semantics 2011)*, page 1–8, Graz, Austria, 2011. ACM.

[Murtagh and Legendre, 2014] Fionn Murtagh and Pierre Legendre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification*, 31:274–295, 2014.

[Naik and Mohan, 2019] Nagaraj Naik and Biju R. Mohan. Stock Price Movements Classification Using Machine and Deep Learning Techniques-The Case Study of Indian Stock Market. In *20th International Conference on Engineering Applications of Neural Networks (EANN 2019)*, pages 445–452, Hersonissons, Crete, Greece, 2019. Springer.

[Neely *et al.*, 2014] Christopher J. Neely, David E. Rapach, Jun Tu, and Guofu Zhou. Forecasting the Equity Risk Premium: The Role of Technical Indicators. *Management Science*, 60(7):1772–1791, 2014.

[Nelson *et al.*, 2017] David M. Q. Nelson, Adriano C. M. Pereira, and Renato A. Oliveira. Stock market's price movement prediction with LSTM neural networks. In *2017 International Joint Conference on Neural Networks (IJCNN 2017)*, pages 1419–1426, Anchorage, AK, USA, 2017. IEEE.

[Nickel *et al.*, 2011] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A Three-Way Model for Collective Learning on Multi-Relational Data. In *28th International Conference on Machine Learning (ICML 2011)*, pages 809–816, Bellevue, WA, USA, 2011. Omnipress.

[Nickel *et al.*, 2016] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic Embeddings of Knowledge Graphs. In *30th AAAI Conference on Artificial Intelligence (AAAI 2016)*, pages 1955–1961, Phoenix, AZ, USA, 2016. AAAI Press.

[Pejić Bach *et al.*, 2019] Mirjana Pejić Bach, Živko Krstić, Sanja Seljan, and Lejla Turulja. Text mining for big data analysis in financial sector: A literature review. *Sustainability*, 11(5):1277, 2019.

[Pujara, 2017] Jay Pujara. Extracting Knowledge Graphs from Financial Filings: Extended Abstract. In *3rd International Workshop on Data Science for Macro–Modeling with Financial and Economic Datasets (DSMM 2017), colocated with the 2017 International Conference on Management of Data (SIGMOD/PODS 2017)*, pages 5:1–5:2, Chicago, IL, USA, 2017. ACM.

[Qi *et al.*, 2020] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A Python natural language processing toolkit for many human languages. In *58th Annual Meeting of the Association for Computational Linguistic: System Demonstrations (ACL 2020)*, pages 101–108, Online, 2020. Association for Computational Linguistics.

[Rather *et al.*, 2015] Akhter Mohiuddin Rather, Arun Agarwal, and Vinjamuri Narshima Sastry. Recurrent Neural Network and a Hybrid Model for Prediction of Stock Returns. *Expert Systems with Applications*, 42(6):3234–3241, 2015.

[Reimers and Gurevych, 2019] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 3982–3992, Hong Kong, China, 2019. Association for Computational Linguistics.

[Rossi *et al.*, 2021] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. *ACM Transactions on Knowledge Discovery from Data*, 15(2):14:1–14:49, 2021.

[Roziere *et al.*, 2023] Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

[Sainz *et al.*, 2024] Oscar Sainz, Iker García-Ferrero, Rodrigo Agerri, Oier Lopez de Lacalle, German Rigau, and Eneko Agirre. GoLLIE: Annotation guidelines improve zero-shot information-extraction. In *The 12th International Conference on Learning Representations (ICLR 2024)*, Vienna, Austria, 2024. OpenReview.

[Sanz-Cruzado *et al.*, 2022] Javier Sanz-Cruzado, Richard McCreadie, Nikolaos Droukas, Craig Macdonald, and Iadh Ounis. On Transaction-Based Metrics as a Proxy for Profitability of Financial Asset Recommendations. In *3rd International Workshop on Personalization & Recommender Systems in Financial Services (FinRec 2022), colocated with the 16th ACM Conference on Recommender Systems (RecSys 2022)*, pages 1–9, Seattle, WA, USA, 2022.

[Schlichtkrull *et al.*, 2018] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling Relational Data with Graph Convolutional Networks. In *15th Europan Semantic Web Conference (ESWC 2018)*, pages 593–607, Heraklion, Greece, 2018. Springer.

[Sun *et al.*, 2018] Yunchuan Sun, Mengting Fang, and Xinyu Wang. A Novel Stock Recommendation System Using Guba Sentiment Analysis. *Personal and Ubiquitous Computing*, 22(3):575–587, 2018.

[Sun *et al.*, 2019] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *7th International Conference on Learning Representations (ICLR 2019)*, New Orleans, LA, USA, 2019. OpenReview.

[Vrandečić and Krötzsch, 2014] Denny Vrandečić and Markus Krötzsch. Wikidata: A Free Collaborative Knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[Wang *et al.*, 2014] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge Graph Embedding by Translating on Hyperplanes. In *28th AAAI Conference on Artificial Intelligence (AAAI 2014)*, pages 1112–1119, Québec City, Québec, Canada, 2014. AAAI Press.

[Wang *et al.*, 2019] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. KGAT: Knowledge Graph Attention Network for Recommendation. In *25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 2019)*, pages 950–958, Anchorage, AK, USA, 2019. ACM.

[Wang *et al.*, 2021] Meihong Wang, Linling Qiu, and Xiaoli Wang. A Survey on Knowledge Graph Embeddings for Link Prediction. *Symmetry*, 13(3):485:1–485:29, 2021.

[Wang *et al.*, 2023] Ting Wang, Jiale Guo, Yuehui Shan, Yueyao Zhang, Bo Peng, and Zhuang Wu. A knowledge graph-GCN-community detection integrated model for large-scale stock price prediction. *Applied Soft Computing*, 145:110595, 2023.

[Wu *et al.*, 2020] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. Zero-shot Entity Linking with Dense Entity Retrieval. In *2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 6397–6407, Online, 2020. Association for Computational Linguistics.

[Yang *et al.*, 2015] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*, San Diego, CA, USA, 2015.

[Zhang *et al.*, 2017] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. Stock Price Prediction via Discovering Multi-Frequency Trading Patterns. In *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2017)*, pages 2141–2149, Halifax, Nova Scotia, Canada, 2017. ACM.

[Zhang *et al.*, 2018] Yuxin Zhang, Kunlin Yang, Wei Du, and Wei Xu. Predicting Stock Price Movement Direction with Enterprise Knowledge Graph. In *22nd Pacific Asia Conference on Information Systems (PACIS 2018)*, page 237, Yokohama, Japan, 2018.

[Zhao *et al.*, 2023] Yu Zhao, Huaming Du, Ying Liu, Shaopeng Wei, Xingyan Chen, Fuzhen Zhuang, Qing Li, and Gang Kou. Stock Movement Prediction Based on Bi-Typed Hybrid-Relational Market Knowledge Graph via Dual Attention Networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):8559–8571, 2023.