CONSTRUCTING A SEARCHABLE KNOWLEDGE BASE FROM FINANCIAL TEXT USING INFORMATION EXTRACTION

by

VALARY LIM WAN QIAN

A THESIS SUBMITTED FOR THE DEGREE OF

BACHELOR OF SCIENCE

in

BUSINESS ANALYTICS

in the

UNDERGRADUATE DIVISION

of the

NATIONAL UNIVERSITY OF SINGAPORE

2022

Supervisor:
Assistant Professor Stanley Kok

Project No:
H246110

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Valary Lim Wan Qian

April 5, 2022

Contents

Abstract

Constructing a Searchable Knowledge Base from Financial Text using Information Extraction

by

Valary Lim Wan Qian

Bachelor of Science in Business Analytics

National University of Singapore

The volume of unstructured data available to financial market participants is increasing at a rapid rate, making manual tracking of financial literature an extremely challenging and time consuming task. Instead, recent developments in relation extraction now enable us to programmatically extract complex relational information from financial text and filter for relevant financial information. In this paper, we propose and implement an end-to-end pipeline that extracts relational information, and populates a financial knowledge base, which is connected to an intuitive and efficient financial search engine, FinSearch. In the process, we provide scrapers to retrieve financial abstracts from a wide range of online publications, and introduce a unified schema for coarse and granular relations across various financial activities, functions and influences. We train a state-of-the-art joint-entity information extraction model, DyFinIE, and apply DyFinIE to a large corpus of financial abstracts to construct FinKB, a structured knowledge base of over a million coarse and granular relational triplets. This knowledge base can be accessed by the public through FinSearch, a financial search engine we developed that allows users to efficiently query for and filter relevant financial abstracts. Our experiments show that users are able to retrieve accurate and relevant relational information from FinSearch 84.2% and 69.8% of the time for coarse and granular relations respectively.

Subject Descriptors:

H.3.3: Information Search and Retrieval

I.6.5: Model Development

I.2.7: Natural Language Processing

Keywords:

Information extraction, Joint relation extraction, Financial knowledge base, Semantic search

Implementation Software and Hardware:

Ubuntu 20.04.3 LTS (Focal Fossa), Apache 2.4.41, Python 3.7, Flask 2.0.2, JavaScript ES6, Vue 4.1.2, NUS School of Computing cluster

Acknowledgments

This work would not have been possible without the support of many individuals. I would like to thank my advisor, Dr. Stanley Kok, for his constant support, guidance, and assistance. I have benefited greatly from his wealth of knowledge and our meetings have been vital in providing me with a clear research direction.

I would also like to extend my gratitude to Dr. Huang Ke-Wei for his valuable and objective feedback, and Lai Yan Jean, for her time and assistance in evaluating my research output.

Finally, I could not have completed this dissertation without the support from my friends and family. I would like to specially thank my partner, Ayush Chattoraj, for his unwavering encouragement and support at every step of way.

List of Figures

List of Tables

Chapter 1

Introduction

In the last two decades, the volume of unstructured data available to financial market participants has increased rapidly (Lewis and Young, 2019). Examples of unstructured financial data include analyst reports, academic journals, social media posts by various financial stakeholders, news and media commentary, corporate reporting packages, etc. This expansion in the availability of data presents an opportunity for market participants to engage in more data-driven decision making and support economic decision-making (Zhou and Zhang, 2018). However, the significant volumes of fragmented data also poses an information overloading problem – manually tracking information across multiple sources and extracting the relevant content is challenging and time consuming (Lewis and Young, 2019), and the growth rate of financial text data far exceeds the pace at which we can manually process it (Zhou and Zhang, 2018).

Existing machine learning studies in financial analytics often focus on applying sentiment analysis or topic modelling. However, recent developments in relation extraction now enable us to extract more complex relational information from financial text. For example, given the sentence '*An increase in risk aversion reduces wages, unemployment, and investment.*', the triplets (*risk aversion*; `reduces`; *wages*), (*risk aversion*; `reduces`; *unemployment*), and (*risk aversion*; `reduces`; *investment*) can be extracted, where the relational phrase '`reduces`' indicates the semantic relationship between the entities '*risk aversion*' and '*wages*', '*unemployment*', or '*investment*'.

Through relation extraction, we can algorithmically extract entity-relation triplets

and populate a financial knowledge base. This knowledge base can serve as an asset in summarising how financial instruments and players correlate or affect the movements of financial markets, and can then be utilised to explore specific financial entities and trends or be applied to downstream tasks of natural language processing (NLP) such as question-answering, the formation of a knowledge graph, or recommendation systems to explore specific financial entities and trends.

In this paper, we propose a novel neural network-based information extraction (IE) model for financial relation extraction, adapted from DyGIE++ (Wadden et al., 2019). We employ Financial Bidirectional Encoder Representations from Transformers (FinBERT), a BERT model pre-trained for financial language to enhance model performance on texts in the financial domain, for token encoding, allowing us to derive contextualised embeddings of our financial text without the need for significant amounts of data. This consideration is important as there are no existing publicly-available corpora of labelled financial entity-relation data that can be used to train a fully supervised relation extraction model. Unlike the joint-entity relation extraction algorithms that have previously been implemented in the finance space, DyGIE++ (Wadden et al., 2019) propagates global (cross-sentence) information, which could be useful as we are dealing with abstracts that contain multiple related sentences.

Our proposed model is trained on FinMechanic, an annotated dataset of 290 abstracts from financial academic papers, with 2953 instances of entity-relations. While unstructured financial information is readily available from a variety of sources, this paper will focus on extracting entity-relations from the abstracts of published financial papers, because academic papers have been vetted through peer reviews and accepted by publishers, and are therefore more likely to reflect reliable or "ground truth" financial reasoning suitable for the initial development of a knowledge base. As financial research deals with a wide variety of disciplines, including, but not limited to, accounting, marketing, organisational behaviour, economics, and statistics, it is difficult to develop a fine-grained relation schema that captures the relations available in all financial academic papers. To consolidate this diverse information, we modify

and adopt the coarse-grained unified schema proposed by Hope et al. (2021) in our annotated dataset, FinMechanic, where relations are divided into only 2 broad categories, `DIRECT` and `INDIRECT` mechanisms. We also test a more granular relation schema consisting of `ATTRIBUTE` and `FUNCTION` which are `DIRECT`, and `POSITIVE`, `NEGATIVE`, `NEUTRAL`, `NONE`, `CONDITION`, `COMPARISON`, `UNCERTAIN` which are `INDIRECT`.

We then apply the DyFinIE model to a large corpus of 87,895 financial abstracts to construct FinKB (Financial Open Mechanism Knowledge Base). We showcase the utility of FinKB through the development of FinSearch (Financial Search Engine), a financial search engine service that provides an interface through which users can easily query and retrieve data from FinKB. We train a custom embedding model for financial semantic search (FinMultiQA), and modify an approximate nearest neighbour descent search algorithm to efficiently identify relevant relational triplets.

In summary, we make the following key contributions:

- **Unified Schema for Relations.** We introduce a modified schema for both coarse and granular relations that generalises across the many activities, functions, and influences found in financial abstracts.
- **DyFinIE (Dynamic Financial IE).** We modify and train an IE model using the DyGIE++ approach (Wadden et al., 2019), to develop DyFinIE, a state-of-the-art model for extracting financial entity-relations.
- **FinKB (Financial Open Mechanism Knowledge Base).** We apply DyFinIE to a large dataset of financial abstracts to construct a structured knowledge base of financial relational triplets.
- **FinSearch (Financial Search Engine).** We build a financial search engine that allows users to easily search for combinations of entities and retrieve relevant abstracts from FinKB.

Chapter 2

Related Work

2.1 Supervised Relation Extraction

2.1.1 Pipeline Extraction Models

Traditional relation extraction models (Chan and Roth, 2011; Hoffmann et al., 2011; Mintz et al., 2009; Zelenko et al., 2002; D. Zeng, K. Liu, et al., 2014) adopt a pipeline approach for relation extraction that consists of two stages: (1) a named entity recognition (NER) system is used to identify entities in a text, (2) a classifier is used to identify and classify relations between pairs of entities. However, these works focus on relation extraction and assume entities to be known, and therefore depend heavily on the performance of an external NER system to successfully identify the entities in a text (Nayak and Ng, 2020). In addition, the complete separation of entity detection and relation classification ignores the possible interaction and correlation between the two sub-tasks, and is therefore susceptible to cascading errors (Q. Li and H. Ji, 2014).

2.1.2 Parameter-Sharing Models

To address the problem of error propagation, Katiyar and Cardie (2017), Miwa and Bansal (2016), and Nguyen and Verspoor (2018) use a parameter-sharing mechanism that brings the entity recognition and relation extraction tasks closer together by sharing their parameters and optimising them together. While this mechanism represents entities and relations as a single model with shared parameters,

it still requires the identification of all entities in a sentence first, before relations can be extracted among all possible pairs of entities. The mechanism therefore omits possible interactions among relation triplets present in a sentence, and does not truly join the entity and relation identification tasks. In addition, such methods extract all possible entities and generate `None` relations that signify that no relation can be found, and are redundant (Zheng et al., 2017).

2.1.3 Encoder-Decoder Models

Zheng et al. (2017) proposed a unified tagging strategy that transforms the joint extraction into a sequence tagging problem. However, this method is unable to identify overlapping triplets, where more than one triplet share the same common entity. To extract overlapping triplets, X. Zeng et al. (2018) proposed an encoder-decoder model, CopyRe, based on sequence-to-sequence (Seq2Seq) learning with a copying mechanism that allows words to be copied multiple times and participate in multiple relation triplets. A major shortcoming, however, is that their model can only copy and generate single-token (one word) entities. Nayak and Ng (2020) improved the model by introducing a representation scheme for relation tuples that tags the start and end tokens of an entity (instead of the entity itself), to allow the encoder-decoder model, which extracts only one word at a time, to identify multi-token entities. D. Zeng, Zhang, et al. (2019) also proposed CopyMTL to address the issues of CopyRe by using a sequence tagging approach to extract full entity names.

The Seq2Seq models by X. Zeng et al. (2018) and Nayak and Ng (2020) rely heavily on the representation of "meaning", which might not be sufficiently accurate in cases where the system needs to refer to sub-sequences of input such as entity names or dates (Gu et al., 2016). Instead, Gu et al. (2016) propose CopyNet, a Seq2Seq learning model that incorporates a copying mechanism. The copying mechanism locates a certain segment of the input sentence, and places the segment into the output sentence to mimic rote memorisation in human language processing.

The performance of these Seq2Seq models rely on being able to successfully

convert relation triplets into a sequence at the training phase. To avoid this conversion, Sui et al. (2020) propose a bipartite matching loss in the encoder-decoder network which considers the group of relation triplets as a set instead of a sequence. While conventional encoder-decoder models use only one encoder, J. Wang and Lu (2020) propose casting the joint extraction task as a table-filling problem that uses two distinct encoders, a table encoder and a sequence encoder, to capture the different types of information for each entity identification and relation extraction sub-task.

2.1.4 Decomposition-Based Models

X. Li et al. (2019), Yu et al. (2019), and Wei et al. (2020) use decomposition-based models that first distinguish all candidate head-entities that may be involved with target relations, then label corresponding tail-entities and relations for each extracted subject to extract overlapping triplets. X. Li et al. (2019) casts the joint extraction task as a multi-turn question-answering problem to leverage on important information for entity and relation classes within the question query and exploit well developed machine reading comprehension models. Yu et al. (2019) represents the task as an end-to-end sequence labeling framework with a novel decomposition strategy that further breaks the two decomposition sub-tasks into several sequence labelling problems.

Wei et al. (2020) introduced a cascade binary tagging framework (CasRel) that models relations as functions that map head-entities to tail-entities in a sentence, instead of treating relations as discrete labels. Takanobu et al. (2018) demonstrates a novel hierarchical reinforcement learning (RL) deep neural network model where related entities are regarded as arguments of a relation to enhance the interaction between entity mentions and relation types. The hierarchical RL system iteratively extracts relational triplets by first identifying relations based on relation-specific tokens in a high-level RL, and subsequently extracting entities associated with the relation using a low-level RL.

2.1.5 Span-Based Models

The aforementioned models consider only information at the local (within-sentence) context. However, Luan et al. (2019) demonstrates that there could be value in considering information beyond the local context by propagating global (cross-sentence) information. Luan et al. (2019) propose Dynamic Graph IE (DyGIE), a framework for information extraction using dynamically constructed span graphs that propagates global context through coreference and relation links. Wadden et al. (2019) modified DyGIE to create DyGIE++, a framework that uses contextualised embeddings like BERT for token encoding to better capture relationships among entities in the same or adjacent sentences. This is in contrast with the context-independent bidirectional LSTM and GloVe embeddings used for token representation in DyGIE. We have chosen to adopt the DyGIE++ framework for its integration of global information that could be useful when looking at abstracts with multiple related sentences, and its use of a pre-trained contextualised embedding that can generate more contextualised embeddings suitable for a domain-specific model.

More recently, Y. Lin et al. (2020) propose OneIE, a joint neutral framework that incorporates global features such as the probability of two entities being involved in the same two relation types simultaneously to capture the cross-subtask and cross-instance interactions. However, this method is more data-intensive, as it requires additional entity labels. Eberts and Ulges (2019) also propose Span-based Entity and Relation Transformer (SpERT), a simple but effective attention-model for span-based joint extraction that uses BERT as a backbone and two feed forward neural networks (FFNNs) to classify spans and relations. Unlike DyGIE++, SpERT reduces model complexity by omitting any graph propagation and using shallow entity and relation classifications, instead focusing on using negative sampling to improve the model. B. Ji et al. (2020) improve on SpERT by adding an attention mechanism to obtain span-specific and contextual semantic representations. We have chosen not to adopt both OneIE and SpERT as these models are more data-intensive, and are unliekly to provide signifiacnt improvements in training

efficiency, at the cost of lower model performance in light of our relatively small training dataset.

2.2 Unsupervised Relation Extraction

Unlike supervised models which are restricted to specific relation models that are provided in the training data, pure unsupervised relation extraction methods enable the extraction of relations not yet seen in the knowledge base (open relations). Unsupervised models generally utilise an encoder to extract sentence embeddings, which are subsequently fed into a clustering algorithm to generate relation labels. This method of unsupervised relation extraction was first proposed by Hasegawa et al. (2004). Their work employs a named entity tagger to create context vectors of named entity pairs. These context vectors are then clustered using hierarchical clustering and common words in the context of all entity pairs within a cluster were selected to describe each relation.

Ali et al. (2021) propose US-BERT, a framework built on a pretrained BERT-based model to encode sentences. Instead of hierarchical clustering, Ali et al. (2021) use affinity propagation, a clustering method that does not require the number of clusters (and hence relations) to be fixed, to identify patterns in encoded sentences. Each cluster's centroid is treated as a different relation type, and new relations are extracted by computing the cosine similarity between the query vector and all available centroids with a confidence value above a certain threshold to avoid semantic drift.

Some works also explore using a probabilistic generative approach to cluster and extract similar relations. Yao et al. (2011) applied Latent Dirchlet Allocation (LDA) models that clustered equivalent textual expressions using entity type constraints within a relation and features on the dependency path between entity mentions. Under this framework, each relation tuple is drawn from a relation type topic distribution selected by a latent relation type indicator variable. Lopez de Lacalle and Lapata (2013) applied this method to general domain knowledge, first encoding

domain knowledge using First Order Logic rules and then integrating them with an LDA model to produce clusters.

Marcheggiani and Titov (2016) argued that previous generative and agglomerative clustering models rely heavily on independence assumptions. Instead, they implement a two-part variational autoencoder (VAE) model consisting of a relation extractor that predicts the semantic relation between two entities, and a factorization model that reconstructs entities relying on the predicted relation. The two components are jointly estimated by minimising the error in entity recovery. However, this model still relies on hand-crafted features extracted by natural language processing tools that might contain errors and are unable to discover new patterns, which might hinder the model's performance. To overcome this limitation, Simon et al. (2019) propose a relation classifier that employs a piecewise convolutional neural network that does not require hand-crafted features. The relation extraction model incorporates two loss functions that enforces the distribution over relations to be uniform.

Yuan et al. (2020) highlighted that the models proposed by Marcheggiani and Titov (2016) and Simon et al. (2019) do not explicitly use the correlation between sentences with the same entity pair. Their work introduced a Clustering-based Unsupervised generative Relation Extraction (CURE) framework that considers this correlation by reading multiple sentences with the same entity pairs as inputs and uses self-supervised learning to predict the shortest path between entity pairs on the dependency graph of one of the input sentences. Relation information is then extracted and entity pairs that share the same relation are clustered, with each cluster assigned a label based on the words in the shortest paths corresponding to the entity pairs in each cluster.

In this paper, we compare our supervised IE model trained on the DyGIE++ framework with two unsupervised baselines, and find that the supervised method still performs better despite our smaller training dataset.

2.3 Relation Extraction in Finance

In the field of financial relation extraction, Vela and Declerck (2009) proposed a feature-based approach on the basis of lexical and syntactic properties to uncover ontological (part-of and subfield-of) relations in German financial text. However, their work is limited to identifying the structures of ontological relations in German text, which might not translate to the discovery of both ontological and non-ontological relations in English text. In addition, due to the large linguistic diversity within the English language, there are a multitude of ways to express a given relation. For example, the following 3 sentences all express relations between a **stock** and **financial instrument**, but display varying syntactic structures.

- "Some examples of **financial instruments** are cheques, shares, **stocks**, bonds, futures, and option contracts."
- "Securities under equity-based **financial instruments** are **stocks**."
- "**Stocks**, bonds, securities, futures - essentially any form of capital that can be packaged and traded can be considered a **financial instrument**."

Using a rule-based approach is therefore unlikely to generalise well to all sentences, resulting in the construction of a knowledge base with high precision but low recall.

Repke and Krestel (2021) propose a pipeline that uses off-the-shelf open-source libraries such as NLTK, spaCy, and OpenNLP to extract financial entities from text. The pipeline then utilises Stanford's OpenIE to split the sentence into sets of clauses, which are grouped with entities to form relational triplets. These relational triplets are refined and used to populate a knowledge graph. In a similar vein, Gupta et al. (2021) presents a zero-shot open information extraction technique that utilises off-the-shelf machine reading comprehension model, Flair NER, to extract entities. They model relation generation as a Question-Answer problem, and create a custom noun question phrase for each pair of predicted entities. These question phrases are passed into the Flair NER model, and the response with the highest confidence is taken as the final relation. However, the off-the-shelf models used by Repke and

Krestel (2021) and Gupta et al. (2021) were trained for open-domain purposes, and are not specifically tuned for financial relation extraction. Their approaches are therefore unlikely to capture the idiosyncrasies and nuances of financial language and text. In addition, they depend heavily on the performance of an external NER system to successfully identify the entities in a text (Nayak and Ng, 2020), and ignore the possible interaction and correlation between the entity and relation retrieval sub-tasks, making them more susceptible to cascading errors (Q. Li and H. Ji, 2014).

Based on an analysis of financial and economic textual data, Zhou and Zhang (2018) report that financial texts tend to have long sentences with redundant information. Thus, they proposed a BiGru Attention Joint Model (BGAJM) for financial news, that incorporates an attention mechanism into a BiGru (Bidirectional Gated Relational Unit) model. The mechanism involves using a word-level attention model to assign higher weights to words that contribute more to the relation prediction. The BGAJM approach was compared with standard pipeline extraction approaches using BiGru, and was found to achieve a greater area under the precision-recall curve (AUC-PR). However, no comparison was made between BGAJM and other joint entity-relation approaches, or with other approaches that use incorporated attention mechanisms.

Reyes et al. (2021) represented the relation extraction problem as a binary classification problem, and utilised BERT to extract entities and relations. The model was developed for the financial texts in Portuguese. While Reyes et al. (2021) suggests the efficacy of using BERT to process entities and identify relations, the proposed model is unable to highlight specific parts of the sentence that represent the relation between two entities. In addition, we hypothesise that the incorporation of a language model specifically designed to tackle NLP tasks in the finance domain, such as FinBERT, a financial language model based on BERT (Araci, 2019), could further improve the performance of financial relation-extraction.

Chapter 3

Data

3.1 Data Collection

Financial abstracts were retrieved from various publishers and Semantic Scholar, a free AI-powered database of scientific literature developed by the Allen Institute of AI. Due to the differences in the publishers, different procedures were used to scrape and filter relevant financial literature papers. We detail these procedures in the following subsections. Abstracts with fewer than 5 words were also filtered out as irrelevant, since these abstracts are unlikely to contain meaningful relations. After scraping and filtering for relevant abstracts, a total of 506597 sentences from 87895 abstracts were collected.

3.1.1 American Economic, American Accounting, Wiley, and Oxford Academic

Scientific papers on these publisher websites are organised by journal. We identified journals relevant to the financial domain and scraped all papers of these financial journals. Journals with the word "Economic" and "Accounting" in their name for American Economic, American Accounting and Oxford Academic publications were deemed to be relevant, alongside journals that were tagged as "Accounting" or "Business & Management" on Wiley. After filtering out irrelevant abstracts, we were able to retrieve 3657 abstracts from American Accounting, 3416 abstracts from American Economic, 10846 abstracts from Wiley, and 20 abstracts from Oxford Academic.

3.1.2 Springer and Elsevier

Both the *Springer API* and *Elsevier API* allow us to query and retrieve all papers based on specific search terms. However, a random sampling of 100 articles from each publisher revealed that the search function in the API were returning a significant percentage of irrelevant papers (25% and 37% relevance scores for Springer and Elsevier respectively). For example, a biomedical paper that mentions the cost of producing a vaccine would also be flagged as a financial paper by the two APIs, despite not having any financial entity-relations in the paper abstract. To filter the dataset and form a collection of abstracts that is more relevant to the financial domain, we filtered for articles with the root word 'finance' in the title. Another random samplings of 100 articles showed that the filtered dataset had a 91% relevance score for Springer and 99% relevance score for Elsevier. In total, 4917 abstracts and 6571 abstracts were retrieved from Springer and Elsevier respectively.

3.1.3 Semantic Scholar

The Semantic Scholar database tags papers by their field of study, such as 'Business', 'Economics', 'Engineering', and 'Mathematics'. For our database, we extracted 50226 scientific papers with the tag 'Business' and 'Economics'.

3.2 Data Annotation

3.2.1 Relation Schema

We formally define a relational triplet as (E_1; `relation`; E_2), where each entity (E_1 and E_2) is a text span and `relation` indicates the type of mechanism relation between them. We allow an entity to take part in multiple relations within a given text. Mechanisms are categorised into two coarse-grained relations, `DIRECT` and `INDIRECT`, adopted from Hope et al. (2021) and modified to fit the financial context:

- `DIRECT`: attributes (e.g. the economic environment consists of the available technology, incentives, constraints, and institutions) and functions (e.g. the

dividend discount model is used to compute stock price)

- `INDIRECT`: influences and associations (e.g. effects of an increase in interest rates), conditions (e.g. the market is well-functioning if market prices equal the costs of producing the housing unit) and comparisons (e.g. which financial metrics are more reflective of a firm's financial health)

We further breakdown the two relations into more granular relations, defined in Table 3.1. We also provide an example of each relation in Appendix B. It should be noted that our relation schema does not differentiate between causation and correlation; For cases where the relation tagged should reflect a causal relationship, we use E_1 to signify the cause, and E_2 to represent the effect. For cases of correlation, we label the entity that comes first in the sentence as E_1 and the other as E_2.

Table 3.1: Relation Definitions

Coarse Relation	**Granular Relation**	**Definition**
`Direct`	`Attribute`	E_1 is an attribute of E_2, E_1 is a subset of E_2.
	`Function`	E_1 is used for E_2, E_1 is a model of E_2.
`Indirect`	`Positive`	E_1 and E_2 are positively related or E_1 increases E_2.
	`Negative`	E_1 and E_2 are negatively related or E_1 decreases E_2.
	`Neutral`	E_1 and E_2 are correlated or E_1 causes E_2, but no clear direction as specified.
	`None`	E_1 and E_2 have no relation. Note that unlike the Parameter Sharing models (Section 2.1.2), `None` relations in our dataset are utilised only if the paper explicitly mentions that two entities are unrelated.
	`Condition`	If E_1, then E_2. E_2 occurs in the event of E_1.
	`Comparison`	E_1 is different from E_2, E_1 is better than E_2.

	`Uncertain`	Relationship between E_1 and E_2 is being studied, unsure if there is a relation between E_1 and E_2.

3.2.2 Coreference

Coreference occurs when two or more entities in an abstract refer to the same person or object. The DyGIE++ model (Wadden et al., 2019) is able to perform coreference resolution that groups spans referring to the same entity into one cluster, allowing the knowledge learned from one relational triplet to benefit another. Table 3.2 illustrates an example of the potential benefits of coreference contexts. It is impossible to identify the entity that '*It*' refers to from within-sentence context alone. However, coreference resolution allows us to identify '*It*' as an entity belonging to a coreference group consisting of '*Fiscal discipline*' and '*It*'.

Table 3.2: Example of Coreference

Example	Fiscal discipline is essential to improve and sustain economic performance. It also maintains macroeconomic stability.
Ideal	(*Fiscal discipline*; `POSITIVE`; *economic performance*) (*Fiscal discipline*; `POSITIVE`; *macroeconomic stability*)
Ideal	(*Fiscal discipline*; `POSITIVE`; *economic performance*) (*It*; `POSITIVE`; *macroeconomic stability*) (*Fiscal discipline*; `COREFERENCE`; *It*)

3.2.3 Annotation Collection

We label our data in accordance to the relation schema and coreference definition provided in Sections 3.2.1 and 3.2.2 using Prodigy (Montani and Honnibal, 2018), an annotation platform that allows us to easily select span boundaries and relations. We then export and perform the following data processing steps on the exported data:

1. Prodigy (Montani and Honnibal, 2018) does not differentiate between relations and coreferences. We therefore perform a processing step to extract the

relations and coreferences separately.

2. Prodigy (Montani and Honnibal, 2018) utilises spaCy (Honnibal and Montani, 2017) to split abstracts by sentences. As the sentence-splitting by spaCy (Honnibal and Montani, 2017) is not entirely accurate, we found that some entities or relations previously labelled had been split into separate sentences. Since our DyGIE++ model (Wadden et al., 2019) can only identify entities and relations belonging to the same sentence, we remove all entities and relations that are part of more than one tokenized sentence.

3.3 FinMechanic

To construct FinMechanic, we randomly sampled 300 abstracts from the pool of abstracts collected and manually annotated them for relational triplets. During the labelling process, we also removed abstracts that were not financial-related. In total, our final annotated sample consists of 290 abstracts and a total of 2953 relational triplets. We further split the sample set randomly into train (173 abstracts/1700 relations), development (62 abstracts/684 relations), and test (55 abstracts/573 relations) sets using an approximate 60-20-20 ratio. This split was executed using a random sampling of abstracts instead of sentences, as splitting by sentence would expose us to potential information leakages in the event that sentences of the same abstract are present in both the train and test sets. Table 3.3 show the distribution of each relation label in the train and test sets.

3.4 Limitations

3.4.1 Data Collection

During the construction of FinMechanic, we sampled 300 abstracts, of which only 290 (96.7%) were determined to be finance-related. This suggests that our full

Table 3.3: Distribution of Relations in Annotated Dataset

Relation (Coarse)	**Sub-Relation (Granular)**	**Train**	**Dev**	**Test**	**All**
Direct	Attribute	578	201	215	994
	Function	334	132	115	581
Indirect	Positive	156	53	50	259
	Negative	85	50	35	170
	Neutral	300	125	86	511
	None	42	17	21	80
	Condition	72	33	12	117
	Comparison	34	27	13	74
	Uncertain	99	42	26	167

corpus of abstracts still consists of abstracts from irrelevant papers, which might contribute noise to our knowledge base.

3.4.2 Relational Triplet Representation

Our relation schema is unable to reflect the magnitude of a relation. For instance, the sentence 'The individual mandate's exemptions and penalties had little impact on coverage rates.' suggests that there is some degree of relation between '*individual mandate's exemptions and penalties*', and '*coverage rates*'. However, the magnitude of the relation, *little*, cannot be expressed through our relation labels. In addition, in sentences like 'The tendency toward default is governed more by systematic risk than by idiosyncratic risk.', we are able to express that there is a relation between '*systematic risk*' and '*default*', as well as '*idiosyncratic risk*' and '*default*', but are unable to express their relative difference in magnitude through the relation labels (both will be given a `NEUTRAL` tag).

The relation schema used also leaves room for some ambiguity in labelling. For instance, the sentence 'A greater sense of understanding of investment risk was associated with lower risk ratings.', we can choose to derive either the relational triplet (*greater sense of understanding of investment risk*; `NEUTRAL`; *lower risk ratings*) or (*greater sense of understanding of investment risk*; `NEGATIVE`; *risk ratings*). Additionally, we observed instances where the second entity in a relational triplet was implied and could not be expressed in the form of a relational triplet. From

the example in Table 3.4, we observe that '*the impact of the flat-tier*' refers to a flat earnings tier's impact on private savings. However, the entity *private savings* is purely implied.

Table 3.4: Example of Incontiguous Span

Example	The earnings-related tier of the pension scheme is found to have a negative impact on private savings. The impact of the flat-tier is not significantly different from 0.
Ideal	(*earnings-related tier of the pension scheme*; `NEGATIVE`; *private savings*) (*flat-tier*; `NEGATIVE`; *private savings*)
Actual	(*earnings-related tier of the pension scheme*; `NEGATIVE`; *private savings*) (*flat-tier*; `NEGATIVE`; **?**)

3.4.3 Annotation and Model

Due to the limitations of our annotation software, we were unable to label overlapping entities, and have chosen to treat entities as non-overlapping text spans; That is, each word in the abstract will belong to a maximum of one entity. In addition, we observed that certain abstracts contained entities that were best represented by incontiguous spans of text. However, as Prodigy (Montani and Honnibal, 2018) and DyGIE++ (Wadden et al., 2019) are unable to label and handle incontiguous entities respectively, we have chosen to deal with instances of incontiguous entities by selecting only one of the two relevant spans of text as the entity. An example of this is shown in Table 3.5.

Table 3.5: Example of Incontiguous Span

Example	Social welfare in the monopoly is lower than the duopoly.
Ideal	(*social welfare in the duopoly*; `COMPARISON`; *Social welfare in the monopoly*) The `COMPARISON` is between social welfare in the monopoly and duopoly, but we are unable to create an entity *social welfare in the duopoly* as the term *social welfare* is not directly followed by *the duopoly*.
Actual	(*the duopoly*; `COMPARISON`; *Social welfare in the monopoly*)

Chapter 4

Pipeline

Our project pipeline (Figure 4.1) can be split into two broad components, (1) extracting a knowledge base of financial mechanisms (FinKB) using our unified schema, and (2) FinKB usage. We describe our approach for both components in the sections below.

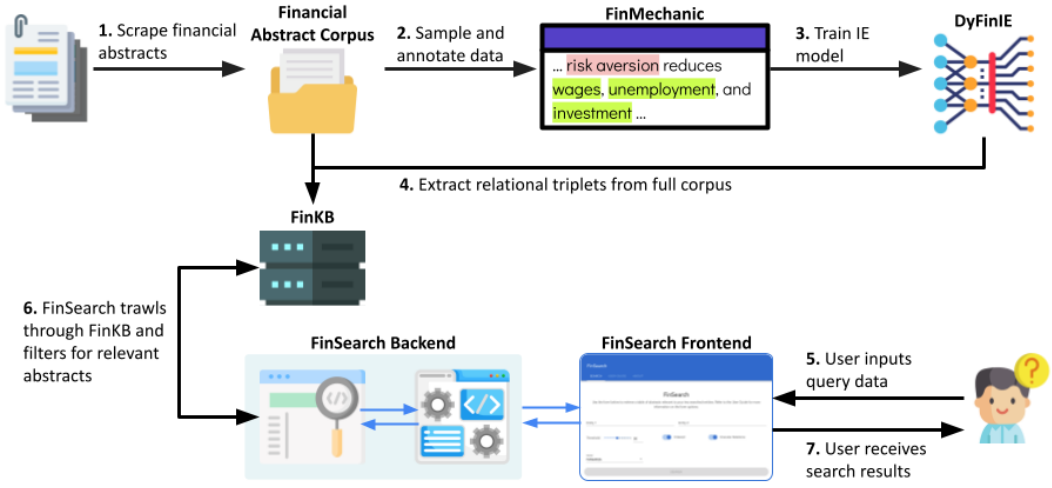


Figure 4.1: Overview of Full Pipeline

4.1 FinKB Construction

To construct FinKB, we first scrape financial abstracts from various publications to form a financial abstract corpus of more than 87,895 abstracts (Section 3.1). We then annotate a random sample of the data to form FinMechanic, a financial

dataset with labelled relational triplets (Section 3.2). This annotated financial dataset is used to train and tune our information extraction model, DyFinIE. Finally, we form our corpus-level knowledge base FinKB by applying the tuned DyFinIE model to each document in our corpus, and extract over a million financial relation-triplets (Section 6).

4.2 FinKB Usage

To allow users to easily access and interpret FinKB, we develop FinSearch, a service that accepts user queries, searches through FinKB, and displays the most relevant abstracts and relational triplets for the user. We encode all entity spans in FinKB to a 384-dimensional dense vector space using an embedding model trained specifically for financial semantic search (Section 7.3), and make use of an approximate nearest neighbour descent algorithm to efficiently identify relevant relational triplets (Section 7.3).

Chapter 5

DyFinIE

5.1 Model Architecture

We train DyGIE++ (Wadden et al., 2019), an end-to-end span-based IE model which extracts entities and relations jointly. We provide an overview of the main components of DyGIE++ in Figure 5.1. For more details, refer to Wadden et al. (2019) and Luan et al. (2019).

Token Encoding: We modify DyGIE++ to use FinBERT (Araci, 2019), instead of BERT. Token representations are generated using a "sliding window" approach, feeding each sentence to FinBERT together with a size-L neighbourhood of surrounding sentences.

Span Enumeration: For each span s_i, its vector representation g_i^0 is obtained by concatenating the tokens representing their left and right endpoints and an embedded span width feature.

Graph Propagation: The propagation process starts from the span representations g_i^0. At each iteration t, we compute an update vector u_C^t for each span s_i and use it to update the presentation of g_i^t to produce the next span representation g_i^{t+1}. This process is repeated N times and the representations g_i^N share contextual information across spans that are likely to be antecedents in the coreference graph. The outputs g_i^N are then passed as inputs to the relation propagation layer. Similarly, at iteration t, we compute an update vector u_R^t for each span s_i and use it to update the

representation of g_i^{N+t} to the next span representation g_i^{N+t+1}. This is repeated M times until we obtain the final representation g_i^{N+M}

Multi-Task Classification: Outputs of the relation graph layer are passed into a two-layer feedforward neural net (FFNN) that serves as a scoring function to predict the entity and relation labels. For entities, we compute $\text{FFFN}_{task}(g_i)$. For relations, we concatenate the relevant pair of entity embeddings and compute $\text{FFNN}_task([g_i, g_j])$.

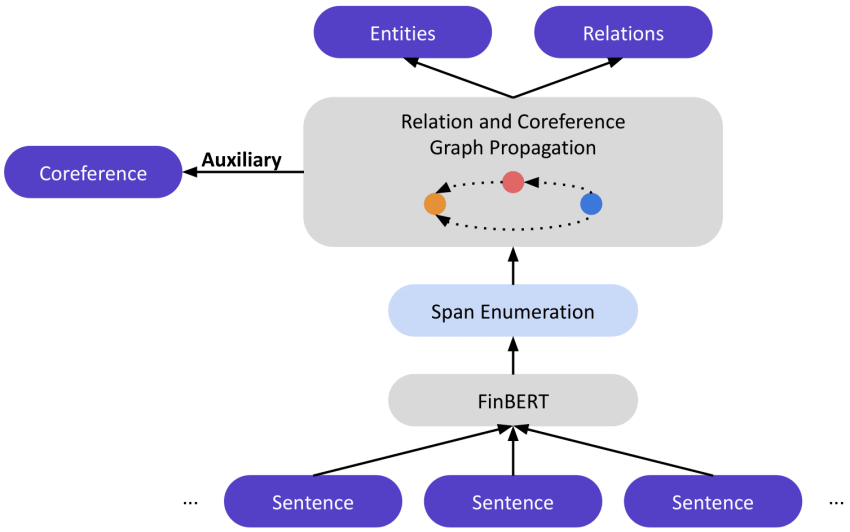


Figure 5.1: Overview of Modified DyGIE++ Framework

5.2 Evaluation Metrics

In the standard implementation of the DyGIE++ model (Wadden et al., 2019), an entity-relation triplet is correctly predicted if and only if the both entities E_1 and E_2 are correctly identified (and in the correct order), and the relation label is accurately classified (we denote this metric as ExactRel). For evaluation, we introduce several relation-based (represented by the infix **Rel**) and span-based (represented by the infix **Span**) computation methods of determining if an entity-relation triplet is correctly extracted. The postfix **ND** represents a metric that has

no direction (ND); In other words, the relation triplet is correct as long as both entities are extracted correctly, and the order of these entities (E_1 or E_2) is ignored.

The prefix **Exact** indicates that a predicted entity span will only be classified as correct if the predicted entity span is exactly the same as the gold entity span, while the prefix **Fuzzy** labels a predicted entity span as correct as long as the predicted entity span contains the gold entity span, and does not exceed the gold entity span by more than 5 tokens. The prefix **Rouge** (Recall-Oriented Understudy for Gisting Evaluation) considers a predicted entity span to be correct if the ROUGE-1 F_1 score between the predicted entity and gold entity is greater than a threshold of 0.5 (C.-Y. Lin, 2004). ROUGE-1 is a widely used partial-matching similarity function that counts the number of overlapping unigrams, word sequences, and word pairs between the predicted entity and gold-standard entity (Equation 5.3).

$$precision^{rouge} = \frac{|S_i \cap S_i^{gold}|}{|S_i|} \tag{5.1}$$

$$recall^{rouge} = \frac{|S_i \cap S_i^{gold}|}{|S_i^{gold}|} \tag{5.2}$$

$$F_1^{rouge} = 2 \cdot \frac{precision^{rouge} \cdot recall^{rouge}}{precision^{rouge} + recall^{rouge}} \tag{5.3}$$

where S_i and S_i^{gold} refer to the set of tokens in E_i and E_i^{gold} respectively.

The prefix **Jaccard** considers a predicted entity span to be correct if the Jaccard index between the predicted entity span and gold-standard entity span is greater than 0.5. In this context, the Jaccard index is defined as the size of the intersection divided by the size of the union of the predicted entity token set and the gold entity token set (Equation 5.4).

$$J(S_i, S_i^{gold}) = \frac{|S_i \cap S_i^{gold}|}{|S_i \cup S_i^{gold}|} \tag{5.4}$$

where S_i and S_i^{gold} refer to the set of tokens in E_i and E_i^{gold} respectively.

We also introduce a token-based evaluation metric (ExactToken) that compares the total number of correctly extracted entity tokens with the total number of

extracted entity tokens (precision) and total number of gold standard tokens (recall). For each of the aforementioned computation methods, we compute the precision, recall and F_1 scores. More detailed explanations of each evaluation metric, as well as positive and negative examples of predictions are provided in Appendix C.

5.3 Baselines

We compare the performance of our model against two baselines:

- OpenIE: A pre-trained deep BiLSTM sequence prediction model (Stanovsky et al., 2018) implemented on the AllenNLP framework. OpenIE formulates the relation extraction problem as a sequence tagging problem, and uses a BERT model to generate word and part of speech embedding. These embeddings are fed into a Bidirectional Long-Short Term Memory (BiLSTM) network which computes contextualized output embeddings. The outputs are used in softmaxes for each word, producing indepdendent probability distributions over possible entity or relation tags.
- SRL: A pre-trained BERT-based Semantic Role Labelling (SRL) model (Shi and J. Lin, 2019), also implemented on the AllenNLP framework. SRL first tokenizes the input sentence and feeds the tokenized input into a BERT model to obtain context embeddings for each token. The context embeddings are concatenated with position embeddings and passed into a BiLSTM network. The final hidden states in each direction of the BiLSTM are used for prediction with a one-hidden-layer MultiLayer Perceptron (MLP).

For both models, we extracted relations of the form (`ARG0`; `VERB`; `ARG1`) as (E_1; `RELATION`; E_2) and evaluated the performance using our span-based and token-based metrics detailed in Section 5.2. As these models do not generate relation labels in accordance to our schema, we will only evaluate them using span-based metrics (infix **Span**) and the token-based metric (infix **Token**).

5.4 Hyperparameter Search

We plan to perform hyperparameter search over the following set of parameters, comparable to the search done by Hope et al. (2021):

- **Dropout** is randomly selected from intervals [0, 0.5].
- **Learning Rate** is randomly selected between [1e-5, 1e-2].
- **Hidden Size** is randomly selected from the interval [64, 512].

The hyperparameter search is implemented using the Optuna package (Akiba et al., 2019) that uses the Tree-structured Parzen Estimator (TPE) algorithm (Bergstra et al., 2011) to sample optimal hyperparameters. For each experiment, we set the search space to be among 30 total samples in the hyperparameter space. In accordance with the standard DyGIE++ model (Wadden et al., 2019), we train our model to maximise performance based on the **ExactRel** F_1 score and selected the set of parameters that achieved the best **ExactRel** F_1 on the development set. The training and validation curves of the best models for DyFinIE can be found in Appendix D.3.

5.5 Results

Tables 5.1 - 5.5 and Tables 5.6 - 5.10 show the results of our DyFinIE model compared to the baselines OpenIE and SRL on the FinMechanic test set with coarse and granular relation labels respectively. We find that across both coarse and granular datasets, our model outperforms the baselines for all relation-based and span-based metrics, with DyFinIE improving 35.1% on OpenIE and 35.2% on SRL in the strictest **ExactSpan** metric. We note that DyFinIE achieves a lower recall and F_1 score than both baselines for the token-based metric. We provide several cases that might illustrate this poorer token-identification performance in Appendix D.4, along with other cases where our model performs well or fails. We also test the performance of DyFinIE against the baselines on an external dataset

labelled by another volunteer who is an expert in finance (Appendix E). We also provide comparisons between FinBERT and BERT (Appendix D.1) and DyFinIE and off-the-shelf NER models (Appendix D.2).

5.5.1 FinMechanic (Coarse Relations)

Table 5.1: Performance on FinMechanic Dataset (Coarse Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	34.9	18.7	24.3	37.5	20.1	26.1	39.4	21.1	27.5	42.0	22.5	29.3

Table 5.2: Performance on FinMechanic Dataset (Coarse Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	48.2	25.8	33.6	53.7	28.8	37.5	42.0	22.5	29.3	45.9	24.6	32.0

Table 5.3: Performance on FinMechanic Dataset (Coarse Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	1.4	1.0	1.2	2.1	1.6	1.8	8.4	6.3	7.2	10.0	7.5	8.6
SRL	1.3	0.9	1.1	1.4	1.1	1.2	8.5	6.4	7.3	11.2	8.4	9.6
DyFinIE	36.5	19.5	25.5	39.7	21.3	27.7	41.0	22.0	28.6	44.6	23.9	31.1

Table 5.4: Performance on FinMechanic Dataset (Coarse Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	12.6	9.4	10.8	15.4	11.5	13.2	8.9	6.6	7.6	10.7	8.0	9.2
SRL	12.9	9.7	11.0	17.8	13.4	15.3	8.3	6.2	7.1	11.8	8.8	10.1
DyFinIE	50.8	27.2	35.5	58.6	31.4	40.9	44.0	23.6	30.7	49.8	26.7	34.8

Table 5.5: Performance on FinMechanic Dataset (Coarse Relations) (5)

	Token		
Model	P	R	F1
OpenIE	52.4	68.6	59.4
SRL	54.1	69.2	60.8
DyFinIE	86.4	34.3	49.1

5.5.2 FinMechanic (Granular Relations)

Table 5.6: Performance on FinMechanic Dataset (Granular Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	29.3	13.6	18.6	31.2	14.5	19.8	33.1	15.4	21.0	35.3	16.4	22.4

Table 5.7: Performance on FinMechanic Dataset (Granular Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	40.2	18.7	25.5	45.1	20.9	28.6	35.0	16.2	22.2	38.7	18.0	24.6

Table 5.8: Performance on FinMechanic Dataset (Granular Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	1.4	1.0	1.2	2.1	1.6	1.8	8.4	6.3	7.2	10.0	7.5	8.6
SRL	1.3	0.9	1.1	1.4	1.1	1.2	8.5	6.4	7.3	11.2	8.4	9.6
DyFinIE	38.0	17.6	24.1	41.7	19.4	26.5	42.1	19.5	26.7	46.2	21.5	29.3

Table 5.9: Performance on FinMechanic Dataset (Granular Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	12.6	9.4	10.8	15.4	11.5	13.2	8.9	6.6	7.6	10.7	8.0	9.2
SRL	12.9	9.7	11.0	17.8	13.4	15.3	8.3	6.2	7.1	11.8	8.8	10.1
DyFinIE	51.5	23.9	32.7	60.2	27.9	38.1	44.0	20.4	27.9	50.4	23.4	31.9

Table 5.10: Performance on FinMechanic Dataset (Granular Relations) (5)

	Token		
Model	P	R	F1
OpenIE	52.4	68.6	59.4
SRL	54.1	69.2	60.8
DyFinIE	89.4	31.0	46.0

5.6 Limitations

We note that DyFinIE is a fully-supervised model, and might be unable to extract relation patterns not yet seen in the training set. This is particularly significant as we have an extremely small training sample of only 174 abstracts and 1,700 relations, which is unlikely to capture all relation patterns of the 87,895 financial abstracts in our corpus that cover a broad range of financial topics.

In addition, we observe instances where DyFinIE correctly identifies entity spans, but fails to tag the right relation labels between entities (Appendix D.4). This is a more apparent limitation for the granular relation set as opposed to the coarse relation set, which is understandable given the greater complexity of classifying granular relations as the granular relation set contains 9 possible labels, compared to 2 possible labels for the coarse relation set. The limited training sample is unlikely to reflect all text patterns for granular relation labels, in particular the None and Condition relation labels that only have 42 and 34 relational triplets in the train set respectively.

Chapter 6

FinKB

We run DyFinIE on the full corpus of financial abstracts to generate FinKB, a financial mechanism knowledge base. FinKB consists of relational triplets of both the coarse relation and granular relation set.

6.1 FinKB Analysis

We provide an analysis of the distribution of relational triplets across relation labels for both coarse and granular relation sets. In total, DyFinIE extracted 646,977 coarse relational triplets and 391,843 granular relational triplets. Figures 6.1 and 6.2 show the count of coarse and granular relation labels in FinKB respectively. We observe that the distribution of granular relation labels is similar to that in FinMechanic, suggesting that the training set used for training of DyFinIE was representative of the larger corpus.

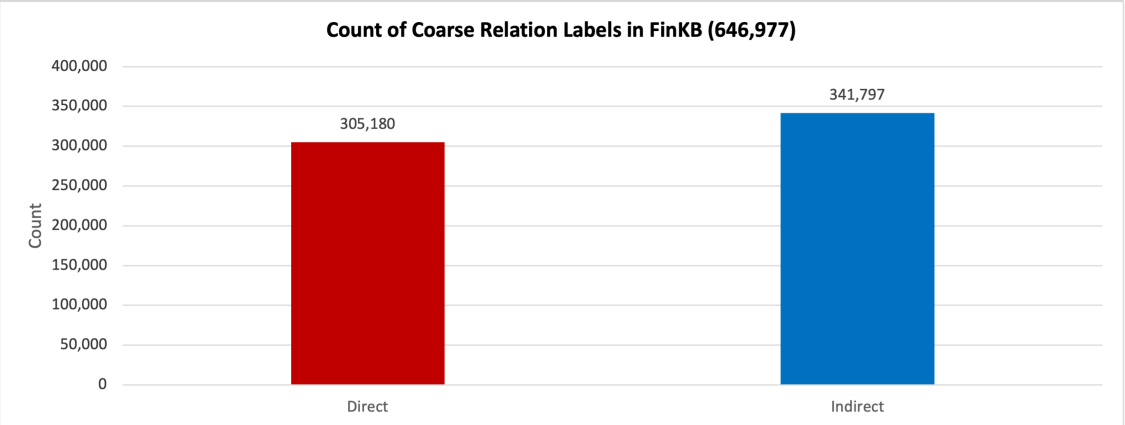


Figure 6.1: Distribution of Coarse Relations in FinKB

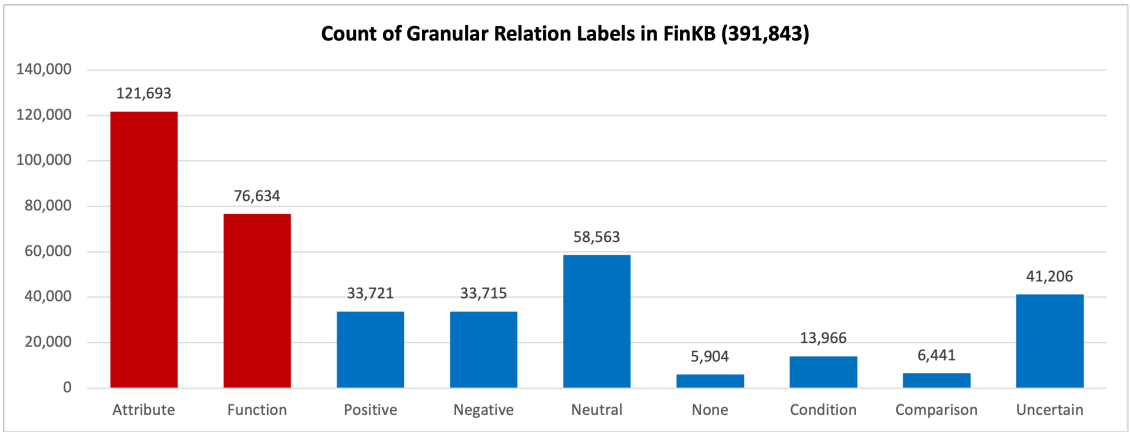


Figure 6.2: Distribution of Granular Relations in FinKB

6.2 Limitations

We note that the range of relational triplets in FinKB is limited to the 87,895 financial abstracts that were initially scraped, and the existing pipeline does not include an automated scraping process to retrieve new articles and extract their relations. Thus, FinKB might have diminishing utility over time, as the body of financial literature expands and new financial research and relations are available.

Chapter 7

FinSearch

FinSearch is a service that allows users to query for relevant abstracts from our financial knowledge base, FinKB. Users enter two keywords or phrases, and the service trawls through its corpus of over 80,000 abstracts from the world of finance to return the most relevant results. Users can also specify the embedding model they wish to leverage for the search, indicate whether they wish to extract relations with a coarse or granular label, filter for ordered (direction-dependent) or unordered (direction-independent) relations, and set a threshold relation score for how strict they want the search to be.

Direction dependence refers to whether users wish to restrict their search entities to entity 1 and entity 2. For instance, if a user queries for '*economic growth*' as search entity 1 and '*stock prices*' as search entity 2 with the directed relations filter, the user will filter for relational triplets where entity 1 is similar to '*economic growth*' and entity 2 is similar to '*stock prices*'. This is useful if the user wishes to find the causal effect of '*economic growth*' on '*stock prices*', since our relation schema is defined in such a way that E_1 affects or causes E_2. However, if the user only wishes to establish correlation and not causation, or does not wish to evaluate the order of entities, he can opt for the undirected relations setting, and search for cases where '*economic growth*' and '*stock prices*' appear in either entity 1 or entity 2.

7.1 Service Architecture

The FinSearch service has a separate frontend and backend system, with the backend consisting of two separate microservices, a querier and an embedder.

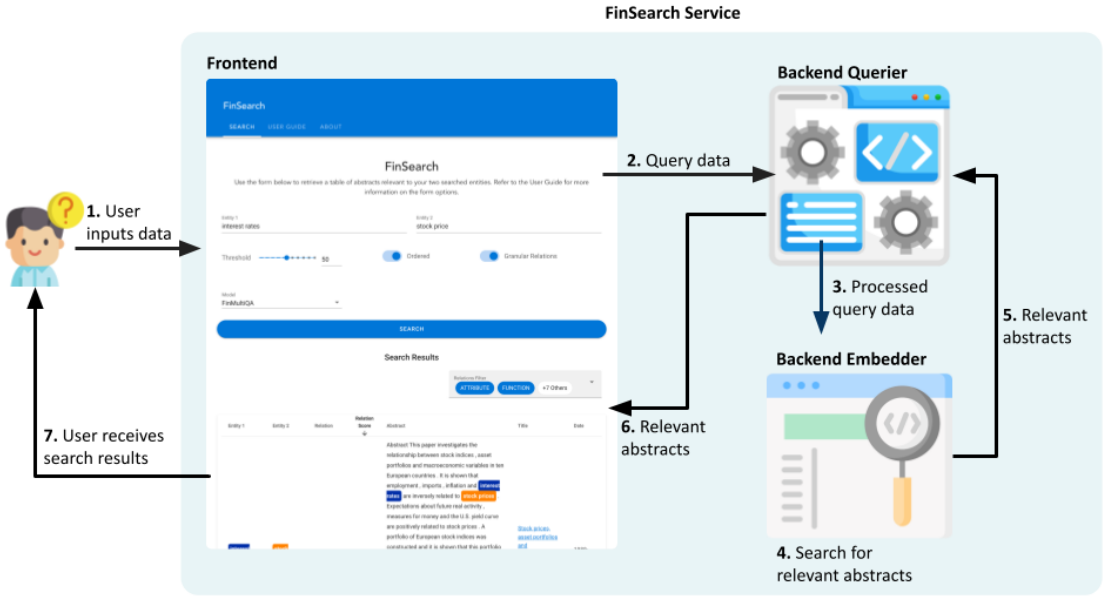


Figure 7.1: Overview of FinSearch Service Pipeline

7.1.1 FinSearch Frontend

The FinSearch Frontend is deployed using surge at finsearch.surge.sh. It is written in JavaScript using the Vue.js framework. Having a separate frontend application provides a responsive and aesthetically pleasing user interface that users can interact with to fetch data from FinSearch. The frontend application is also intuitive, allowing users to identify what parameters the service can accept and tweak.

In addition, FinSearch frontend provides client-side form validation, and helps ensure that the data submitted by the user matches the requirements of the Querier. Some examples of checks include whether the user has entered both entities, and whether the user has selected a valid threshold score and embedder model. The form validation serves as an initial check that catches and indicates invalid data entries on the client-side, allowing the user to fix their queries immediately. If an invalid query is allowed to get to the server before being rejected, there would be a noticeable delay caused by a round trip to the server and back to the client-side to tell the user to fix their data. The separation of frontend and backend also enables

us to display an error message at the frontend if the backend is experiencing outages, and makes it easier for us to ascertain the point of failure if the FinSearch service is down.

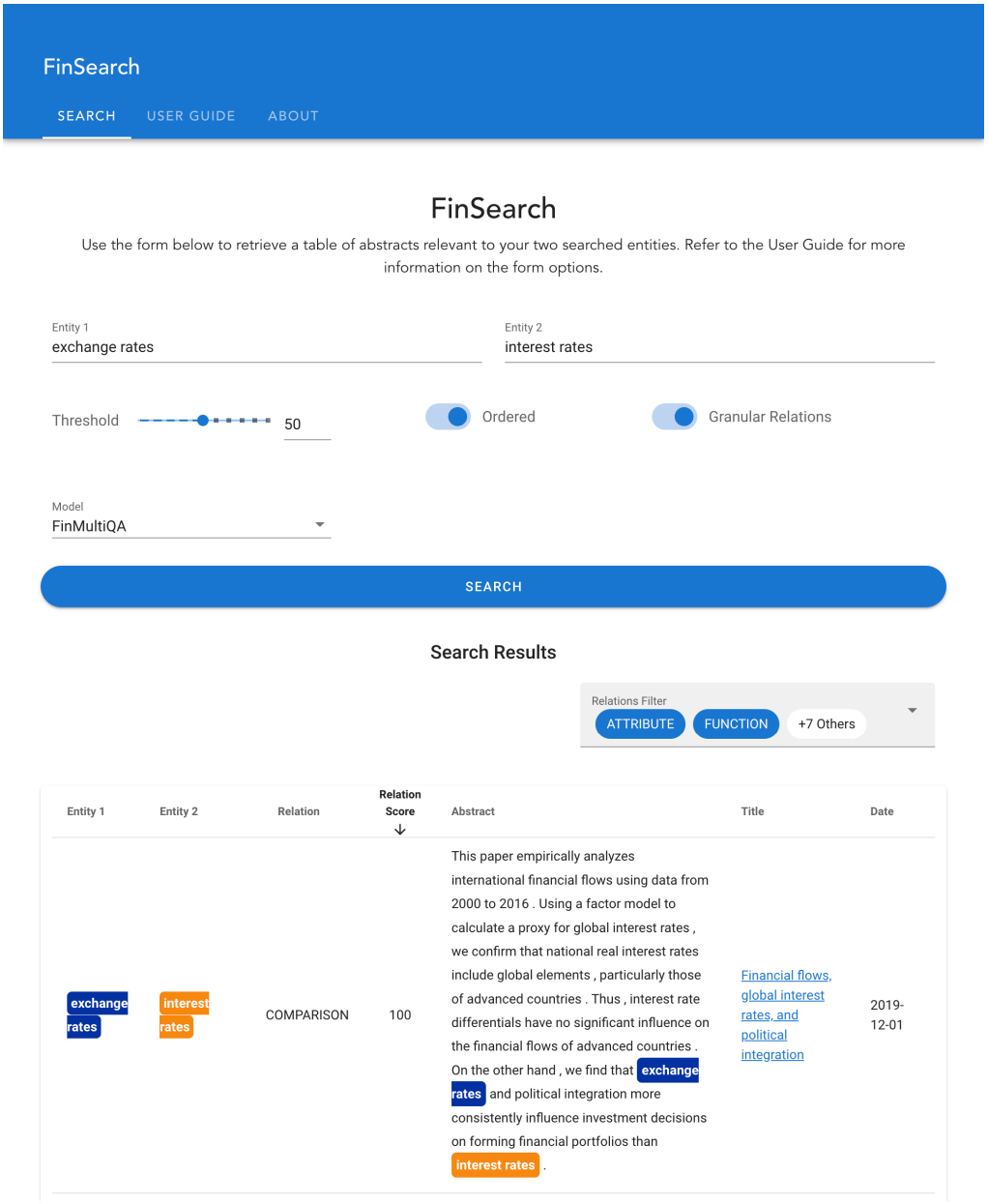


Figure 7.2: Screen Capture from FinSearch Frontend

7.1.2 FinSearch Backend

The FinSearch Backend relies on a layered architecture, and is composed of two separate microservices, each fulfilling a different function.

7.1.2.1 Microservice 1: Querier

The Querier service exposes REST API endpoints for the frontend services to access data, and is served using Apache. It is deployed on a server from the NUS School of Computing, and is written in Python, relying on the Flask framework. The Querier service parses the API requests received and generates the queries in a format that can be used by our Embedder microservice.

7.1.2.2 Microservice 2: Embedder

The Embedder microservice can only be queried via the Querier microservice, and is completely inaccessible by external sources. This microservice is run locally on a server from the NUS School of Computing, and is also developed with the Python web framework, Flask. It is responsible for loading the embedders and data into the system memory, receiving the formulated queries from the Querier service, and trawling through the database to return the most applicable abstracts. A more detailed explanation of the Embedder Microservice pipeline can be found in Section 7.3.

As our embedders and data are memory intensive, the backend takes a significant amount of time for a cold start. As Apache only launches the backend server on request and does not run the actual backend perpetually, hosting the full backend service on Apache would be too slow for any practical purposes. We therefore separate the Querier from the Embedder, and the Embedder is kept running to prevent slow queries due to a cold start, with the embedders and data preloaded into the embedder at all times to keep it highly performant. In addition, Apache can only be loaded with one compiled mod_wsgi module, and therefore one Python environment, at a time. Separating the Querier and Embedder microservices allows us to create a custom Python environment for our Embedder.

7.2 Embedder Microservice

The Embedder microservice is preloaded with embedder models and datasets containing information of all relational triplets extracted, each entity's token embeddings generated using the respective embedder models, and details about its corresponding abstract. On receiving a query, the microservice embeds both search entities using FinMultiQA (Section 7.3, or an alternative selected embedder, if the user chooses. The Embedder then utilises an Approximate Nearest Neighbour Descent Semantic Search algorithm (Section 7.4) to retrieve an approximation of the most semantically similar relational triplets in the dataset by cosine similarity score (Section 7.2.1). The Embedder microservice then aggregates the top relations by abstract, and assigns an abstract similarity score to the abstract (Section 7.2.2). Finally, the microservice returns the top 1000 abstracts by abstract similarity score to the Querier.

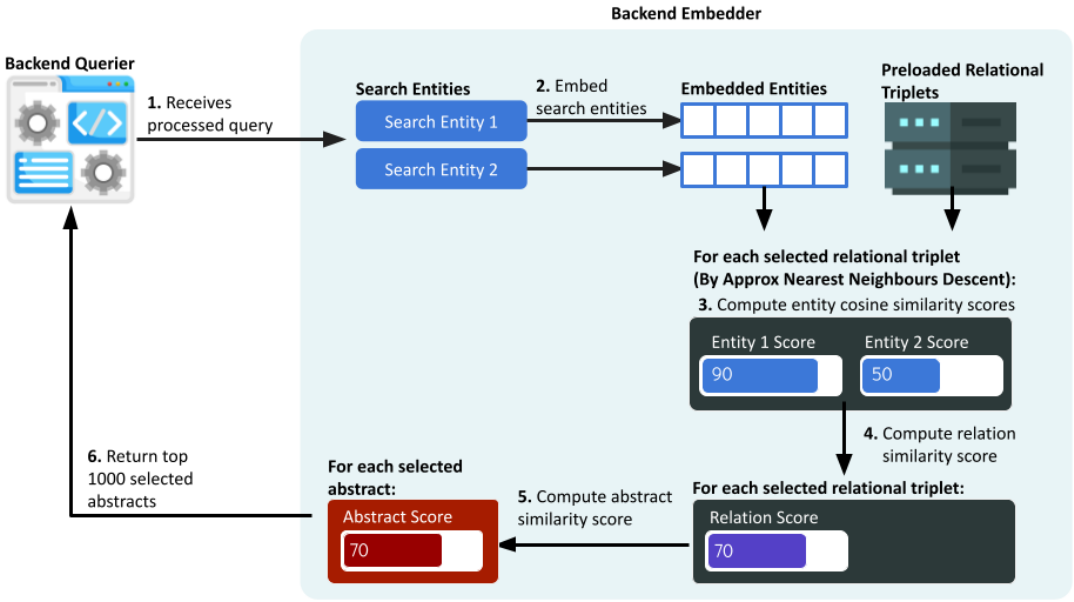


Figure 7.3: FinSearch Backend Embedder Pipeline

7.2.1 Relation Similarity Score Computation

The relation semantic similarity score is computed differently based on whether the user wishes to query for ordered relations or unordered relations.

For ordered relation queries, we calculate the cosine similarity scores between the first and second entity and query respectively (Equations 7.1-7.2), and take the average of the two to be the relation score (Equation 7.3).

$$score_{E1} = cosineSimilarity(E_1, Q_1) \tag{7.1}$$

$$score_{E2} = cosineSimilarity(E_2, Q_2) \tag{7.2}$$

$$score_{REL} = \frac{score_{E1} + score_{E2}}{2} \tag{7.3}$$

where $score_i$ refers to the similarity score for entity or relation i, and $cosineSimilarity$ refers to the cosine similarity value between two entities.

For undirected relation queries, we first calculate the cosine similarity score for all 4 possible pairs of entities and queries (Equations 7.4-7.7). We then compute 2 relation scores, the one where the first entity corresponds to the first query and the second to the second (Equation 7.8), and one for where the first entity corresponds to the second query, and the second entity to the first query (Equation 7.9), as we do not know the correct pairing a priori. The final relation score will be the max of both relation scores computed (Equation 7.10).

$$score_{E1_A} = cosineSimilarity(E_1, Q_1) \tag{7.4}$$

$$score_{E2_A} = cosineSimilarity(E_2, Q_2) \tag{7.5}$$

$$score_{E1_B} = cosineSimilarity(E_1, Q_2) \tag{7.6}$$

$$score_{E2_B} = cosineSimilarity(E_2, Q_1) \tag{7.7}$$

$$score_{REL_A} = \frac{score_{E1_A} + score_{E2_A}}{2} \tag{7.8}$$

$$score_{REL_B} = \frac{score_{E1_B} + score_{E2_B}}{2} \tag{7.9}$$

$$score_{REL} = \max\{score_{REL_A}, score_{REL_B}\} \tag{7.10}$$

7.2.2 Abstract Similarity Score Computation

DyFinIE extracts multiple relational triplets per abstract, where each relational triplet has an individual relation similarity score. We take the maximum relation similarity score to be the abstract similarity score (Equation 7.11).

$$score_{ABSTRACT} = \max_{i \in R}\{score_i\} \tag{7.11}$$

where R refers to the set of relations in the abstract, and $score_i$ refers to the relation score of relation i in the relation set.

7.3 Embedding Model

The computation of relation and abstract similarity scores rely on being able to embed query and retrieved entities into a vector that can then be compared using the cosine similarity function. We therefore explore various pre-trained and custom embedding models.

7.3.1 Pre-Trained Models

We consider three different pre-trained embedding models:

- **FinBERT.** A BERT-based language model developed by (Araci, 2019) and pre-trained on financial data, specifically designed to tackle NLP tasks in the finance domain. The FinBERT model is currently used as the embedder within our IE model, DyFinIE, and has been found to outperform state-of-the-art machine learning methods even with small training sets, but is not tuned specifically for semantic search tasks.

- **MsMarcoQA.** The MsMarcoQA model is an embedding model that maps text to a 384-dimensional dense vector space. The model is built on Microsoft's MiniLM model (W. Wang et al., 2020), which is a deep self-attention distillation model that mimics the self-attention module of the last layer in pre-trained Transformer models. The model is designed specifically for information retrieval

by semantic search, and is trained using the MS Marco Passage Ranking dataset.

- **MultiQA.** The MultiQA model is similarly built on Microsoft's MiniLM model (W. Wang et al., 2020), but is trained on over 215 million different question-answer pair data, including question-answer pairs from the MS Marco Passage Ranking dataset used for MsMarcoQA.

7.3.2 Custom Models

Both MsMacroQA and MultiQA are domain-agnostic, as they are trained on general question-answer pairs provided by sources like Microsoft, WikiAnswers, Google, and Bing, instead of financial question-answer data. We therefore explore the option of further training these models on financial question-answer pairs, to better fit our model to embedding tokens for the financial domain.

7.3.3 Data

To the best of our knowledge, there are no publicly available financial question-answer datasets that we could use to further train MsMarcoQA and MultiQA. To create our custom semantic search dataset for the financial-domain, we annotate a total of 300 entity sets, with each set consisting of 3 entities (E_1, E_2, E_3), where E_1 and E_2 are expected to have high semantic similarity, and E_1 and E_3 are expected to have low semantic similarity. In total, these 300 entity sets make up 600 entity pairs. These entities are retrieved from online financial texts and financial textbooks, and while non-exhaustive, aim to cover a wide range of common financial terms. We will subsequently refer to this dataset as FinSemantic.

We split the sample set randomly into train (150 entity sets/300 entity pairs), development (75 entity sets/150 entity pairs), and test (75 entity sets/150 entity pairs) sets. We then train the MsMarcoQA and MultiQA models on our train set and validate the model using our development set, to produce FinMsMarcoQA and FinMultiQA.

7.3.4 Evaluation Metric

The goal of our similarity score is to ensure that the most relevant abstracts and relational triplets are flagged out to the user. It is therefore more valuable to compare the differences in cosine similarity score obtained from entities with high similarity versus entities with low similarity. For instance, given the pairs of entities $pair_1$ = (*fiscal shocks*, *fiscal shock*) and $pair_2$ = (*fiscal shocks*, *manufacturing firm*). We would prefer a model that obtains a cosine similarity score of 75 for $pair_1$ and 25 for $pair_2$, as opposed to a model that identifies that the entities in $pair_1$ are equivalent and returns a cosine similarity score of 100 for $pair_1$, but 90 for $pair_2$, where $pair_1$ and $pair_2$ are expected to have high and low semantic relation scores respectively.

We therefore set our evaluation metric to be the average difference in cosine similarity score between a set of entities with high expected semantic similarity, and a set of entities with low expected semantic similarity. (Equations 7.12-7.15).

$$cosine\_similarity_{high_i} = cosineSimilarity(E_{1i}, E_{2i}) \tag{7.12}$$

$$cosine\_similarity_{low_i} = cosineSimilarity(E_{1i}, E_{3i}) \tag{7.13}$$

$$metric_i = cosineSimilarity_{high_i} - cosineSimilarity_{low_i} \tag{7.14}$$

$$metric = \frac{1}{|S|} \sum_{i \in S} \{metric_i\} \tag{7.15}$$

where S refers to the set of all pairs in the test set.

7.3.5 Results

Table 7.1 shows the results of our custom FinMsMarcoQA and FinMultiQA models compared to the baselines Finbert, MsMarcoQA, and MultiQA on our test dataset. We observe that both FinMsMarcoQA and FinMultiQA performed better than their baseline counterparts MsMarcoQA and MultiQA, suggesting that even a small training set could result in significant improvements in domain-specific performance. In addition, we found that FinMultiQA achieved the highest average difference in cosine similarity score of 0.487. We therefore set our default and recommended embedder model in our FinSearch service to FinMultiQA. However, we provide the option for other users to select the baseline models if they wish to use a more established or domain-agnostic embedder.

Table 7.1: Performance on FinSemantic Dataset

Model	**Average Difference in Cosine Similarity Score**
FinBERT	0.170
MsMarcoQA	0.433
MultiQA	0.465
FinMsMarcoQA	0.443
FinMultiQA	0.487

7.4 Semantic Search Algorithm

In this section, we explain how we arrived at the final semantic search algorithm, Approximate Nearest Neighbour Descent (NNDescent) search, to select candidate relations for computation of similarity scores. With the NNDescent algorithm, we were able to reduce the average query time from the FinSearch frontend to only 1.35 seconds.

7.4.1 Naive Search

Our initial naive search algorithm iterated through all relations in the dataset and computed their relation similarity score (Section 7.2.1) to retrieve the top relational triplets. Performing this search sequentially would therefore take linear time (O(n), where n refers to the number of relational triplets in our dataset), which was significant given our large knowledge base of relational triplets (646,977 triplets for the coarse relation set and 391,843 triplets for the granular relation set). On average, we found that a standard search request using the naive algorithm took about 354.15 seconds (5m54s). We felt that this delay was too long to be considered acceptable in a product, and that most users, who have grown accustomed to the search speeds of engines like Google's, would not wait for the results, and were more likely to infer that our service had either broken down or entered an infinite loop.

We identified two main bottlenecks in this approach: (1) the computation of relation similarity score, and (2) the iteration through all relations in our knowledge base. The subsequent improvements aim to target these bottlenecks.

7.4.2 Naive Search with Process-Based Parallelism

As the computation of similarity score of each relational triplet to the search query is independent of other relational triplets, there is no need to run the computation sequentially. Instead, we parallelise the relation similarity score computation step to leverage the 24 CPU cores available on our workstation. We make use of Python's multiprocessing package to allocate and offload the computation tasks to different

worker processes. We found that an average search request using the naive search with process-based parallelism took about 155 seconds (2m35s), 56% less time than the serially executed naive search. While we had 24 CPU cores available, we found that there were only minor improvements in query times beyond 16 worker processes, likely due to the additional costs incurred with parallelism in terms of time taken to set-up and communicate between different worker processes.

7.4.3 Naive Search with Process-Based Parallelism and Numba Compilation

We found that the cosine similarity calculations were the most time-intensive steps when computing relation similarity scores. Our codebase originally performed these calculations using the cosine similarity function implemented by PyTorch, which provides an easy-to-use function, but is not optimised at runtime. Instead, we develop a runtime-optimised cosine similarity function by implementing our own cosine similarity function and adding a Numba decorator to the function. This optimised function utilises the open-source JIT compiler, Numba, to translate the Python cosine similarity function to an optimised machine code at runtime, using the industry-standard LLVM compiler library. We observed that Numba compilation achieved a 17 second improvement – an average search request using the naive search with parallelism and numba compilation took approximately 138 seconds (2m18s).

7.4.4 Approximate Nearest Neighbours Descent Search

Both process-based parallelism and Numba compilation tackle Bottleneck 1 by increasing the speed at which relation similarity score is computed. While these modifications have resulted in significant improvements to query time, we feel that 138 seconds remains too long to be considered acceptable in a public search engine. Instead, we pivot our efforts toward Bottleneck 2, and eliminate the need to perform a linear search of our dataset by using the NNDescent algorithm proposed by Dong et al. (2011) to filter for candidate relevant relational triplets. We provide further explanations of the NNDescent algorithm (Section 7.4.4.1) and our modification

to the algorithm to consider similarities between relations (Section 7.4.4.2). Our average query time with NNDescent is reduced to 1.35 seconds.

7.4.4.1 NNDescent Algorithm

The NNDescent algorithm makes use of an approximate k-nearest neighbour graph to efficiently find good candidates for nearest neighbours of query points from a large training set. Each query is represented as a node in the k-neighbour graph, and the relations between nodes are represented by their edge, where a lower edge weight represents that the nodes are more similar, and conversely, a higher edge weight represents that the nodes are dissimilar.

The algorithm is similar to a breadth-first search algorithm on the k-nearest neighbour graph, and utilises a priority queue sorted by relation score. Upon receiving a query, we choose a random starting candidate node in the k-neighbour graph, and add this node to a priority queue. We then look at the best untried (highest similarity score) candidate node from the priority queue, and add all its k neighbours into the priority queue of potential candidates. Next, we truncate the priority queue and keep only the k best candidates, and repeat the process of looking at the best untried candidate node and adding its neighbours. We repeat this process until we have tried all candidates in the candidate priority queue, to retrieve the top k nodes to the query point.

To construct the approximate k-nearest neighbour graph, we start with a random graph where each node is connected to k random nodes. For each node, we compute the similarity from the node to its neighbour node's neighbours, and update the graph to keep only the k closest neighbours. We perform this continuously until we reach an equilibrium where there are no more updates to the k-nearest neighbour graph. Finally, we perform a pruning step on this intermediate graph to remove all redundant long edges. For each node i, we retain its closest neighbour, node j. For all other neighbour nodes n, we compute the d(n, i) and d(n, j), where d(i, j) is the distance measure between i and j, and remove the edge connecting node i and node n if $d(n, i) > d(n, j)$, i.e., node i has a neighbour that is closer to its retained first

nearest neighbour.

7.4.4.2 Application of NNDescent Algorithm

The algorithm is designed to query for similar entities, with each node in the k neighbour graph being an embedded entity, and the edges between nodes representing entity similarities. As our search query task involves retrieving similar relational triplets which consist of two query entities, we instead extend the algorithm and apply it onto relational triplets. We do so by considering each node in the k neighbour graph as the pair of entities in the relational triplet, and the edges as the relation similarity score between each relational triplet.

We utilise the PyNNDescent Python package to implement this algorithm in our code. As the PyNNDescent package was written to accept entities and an entity similarity scoring function instead of accepting pairs of entities, we perform a preprocessing step to modify our inputs and customise our scoring function. Our preprocessing step involves concatenating the two entities in a relational triplet together, such that they can be passed into PyNNDescent as a single entity. Our customised relation similarity scoring function then unpacks the concatenated embedding into two separate entities, and computes the relation similarity score in accordance with Section 7.2.1.

We retain the process-based parallelism and Numba compilation improvements in this algorithm. The similarity score computations used to construct the k-nearest neighbours graph and candidate priority queue are parallelised. In addition, the custom scoring function is written with a Numba decorator to continue benefiting from runtime-level optimisation. With these improvements, we successfully reduce our average query time to 1.35 seconds.

7.5 Evaluation Metrics

Our main goal is to ensure that relations extracted and retrieved through our FinSearch service are of high quality. To do so, we adopt a similar search quality evaluation approach as to Hope et al. (2021), and will be assessing the FinSearch service in terms of its relevance and correctness. We construct a dataset of 120 different search queries, and retrieve the entities, relation, and text of the top abstract from FinSearch by relation score. This dataset is constructed for both the coarse and granular relation searches. Scores are then assigned to each retrieved relation based on the following criteria:

1. **The extracted entities are semantically similar to search entities.** This condition checks for the relevance of our retrieved relation to the users' desired search. We assign a score of 1 if both entities are relevant, and 0 if any entity is irrelevant. Table 7.2 provides example queries and retrieved entities, as well as their corresponding labels.

Table 7.2: Examples of Assigned Entity Scores

Query Entities	Retrieved Entities	Entity Score
E1: credit score **E2:** risk assessment	**E1:** credit rating **E2:** risk measure	1
E1: abnormal returns **E2:** optioned stocks	**E1:** abnormal returns **E2:** stock picking	0 We assign an entity score of 1 as the query E2 and retrieved E2 are not semantically similar.
E1: green policies **E2:** stock price	**E1:** monetary policy **E2:** stock market	0

2. **The extracted relational triplet accurately reflects a relation mentioned in the text.** This condition checks for the correctness of our retrieved

relation. We assign a score of 1 if the text reports a relation between the two retrieved entities, and if the retrieved relation label correctly represents the reported relation. Table 7.3 provides both positive and negative examples of retrieved relations, where RELG represents the granular relation label, and RELC represents the coarse relation label.

Table 7.3: Examples of Assigned Relation Scores

Retrieved Relation	**Text**	**Granular Score**	**Coarse Score**
E1: leverage **E2:** firm performance **RELG:** `NEUTRAL` **RELC:** `INDIRECT`	Corporate governance theory predicts that **leverage** affects agency costs and thereby influences **firm performance**.	1	1
E1: shareholder value **E2:** M&A acquisition deals **RELG:** `POSITIVE` **RELC:** `INDIRECT`	The acquiring firms are more likely to experience decreased **shareholder value** through **M&A acquisition deals**.	0 The correct relation label is `NEGATIVE`.	1 Both `POSITIVE` and `NEGATIVE` are `INDIRECT`.
E1: market makers **E2:** lion's share of HFT trading volume **RELG:**`NEUTRAL` **RELC:**`INDIRECT`	We find that **market makers** constitute the **lion's share of HFT trading volume** and limit order traffic	0 The correct relation label is `ATTRIBUTE`.	1 The correct relation label is `DIRECT`.

In addition, we compute an overall relational triplet score, calculated by multiplying the score from Criterion 1 with the score from Criterion 2. This overall

relational triplet score will therefore only reflect 1 if both criteria are met, and 0 if either criterion is not met in its entirety.

7.6 Results

We report two different sets of evaluation results in Table 7.4. The first set of results (Resercher) refer to our self-evaluation of the performance of FinSearch, and the second set of scores were generated with the help of a volunteer with a Financial Analytics and Economics background, but who has not worked on this project, but was briefing on the evaluation criteria.

Overall, FinSemantic achieved an average score of 84.2% for coarse relations and 69.8% for granular relations, showcasing the effectiveness of our approach in searching for relational triplets. The lower overall precision score for granular relations compared to coarse relations can be attributed to the greater complexity of classifying granular relations as the granular relation set has a higher number of labels (9 possible labels, compared to 2 for the coarse relation set). We find that in the majority of our dataset (95.8%), the top relation extracted from FinSemantic for the coarse relation set and granular relation set are identical, with the only differentiator being the relation label.

Table 7.4: Average Search Quality Scores

Annotator	**Entity Score (1)**	**Coarse Score (2A)**	**Granular Score (2B)**	**Overall Score (Coarse)**	**Overall Score (Granular)**
Researcher	0.900	0.942	0.775	0.850	0.704
Volunteer	0.879	0.958	0.775	0.833	0.692
Average	0.890	0.950	0.775	0.842	0.698

We provide some positive and negative query examples of FinSearch, as well as some common problems in Appendix F. We find that FinSearch is able to retrieve relevant abstracts as long as (1) the search terms are unabbreviated and more general, (2) relevant relational triplets are available in the database, and (3) the relational triplets are correctly predicted by DyFinIE.

7.7 Limitations

Our financial embedding model is trained on FinSemantic, a small dataset of only 300 entity pairs, which is unlikely to be representative of the wide range of activities, functions, and influences found in the larger financial corpus. The embedding model might therefore perform more poorly for less-common financial terms, or financial terms outside of our training set. We also observe that our financial embedding model performs poorly on acronyms or abbreviations of financial terms. For instance, FinSearch is unable to retrieve abstracts on '*Modern Portfolio Theory*' when given the search entity '*MPT*' (Appendix F.2.2).

In addition, we find that more specific terms, for instance '*manufacturing firms*' or '*technology firms*' as opposed to just '*firms*' tend to return few relevant search results (Appendix F.2.1). This problem can be attributed to FinKB having abstracts that cover more general concepts, instead of more industry-specific information. Scraping financial abstracts from a wider set of sources could alleviate this issue.

Chapter 8

Conclusions

8.1 Summary

In this project, we introduced a unified schema for coarse and granular relations across various financial activities, functions, and influences from the abstracts of published financial papers. We trained an IE model using the DyGIE++ approach (Wadden et al., 2019) to build DyFinIE, a state-of-the-art joint-entity extraction model for retrieving financial relational triplets. In doing so, we demonstrate that labelling a small dataset can allow us to train a supervised model that performs significantly better than a fully unsupervised IE model.

In addition, we apply DyFinIE to a large financial abstract corpus to construct FinKB, a structured knowledge base of over a million coarse and granular relational triplets. Finally, we developed FinSearch, a financial search engine that showcased the ability of FinKB and allowed users to quickly search and retrieve relevant financial abstracts. Our search engine service involves a novel embedding model designed specifically for financial semantic search, and introduces a method to extend an approximate nearest neighbours descent search algorithm (Dong et al., 2011) to relational triplets.

8.2 Recommendations for Further Work

In this section, we provide some recommendations to address the limitations mentioned within our report, as well as additional points of research.

8.2.1 Data

Most of our limitations centre around having a limited dataset. We reiterate that the FinMechanic used to train our IE model, DyFinIE, and the FinSemantic dataset used to train our search embedding model, FinMultiQA, are both relatively small, and are unrepresentative of the wide range of financial entities and relations. It is therefore recommended for future work to construct a larger annotated dataset for model training, which we believe would improve the performance of both models.

In addition, an automated pipeline that integrates our scrapers and the DyFinIE model can be created, to retrieve newly published financial abstracts, extract their relational triplets, and populate FinKB. This would allow FinKB to be kept up-to-date with the latest financial research. Scraping financial abstracts from a wider set of sources could also be useful in expanding the FinKB to handle sector or industry-specific financial terms.

8.2.2 DyFinIE

At present, the DyFinIE model utilises only local (sentence-level) features to predict relational triplets, and does not make use of data at the abstract or corpus level. In the future, it might be useful to explore integrating global features into DyFinIE, similar to the OneIE model architecture (Y. Lin et al., 2020). For instance, we could provide the model with information on the probability of an entity belonging to two relational triplets, or the probability of a relation label occurring given other relational triplets predicted for the abstract.

8.2.3 FinSearch

We performed a preliminary experimentation with several embedding models designed specifically for either the financial domain, or semantic search. A more thorough experimentation of other available state-of-the-art semantic search models like Open AI's GPT3 embedder (Neelakantan et al., 2022) could be performed to potentially improve the search results retrieved by FinSearch.

Bibliography

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). "Optuna: A next-generation hyperparameter optimization framework". *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Ali, M., Saleem, M., & Ngomo, A.-C. N. (2021). "Unsupervised relation extraction using sentence encoding". In R. Verborgh, A. Dimou, A. Hogan, C. d'Amato, I. Tiddi, A. Bröring, S. Mayer, F. Ongenae, R. Tommasini, & M. Alam (Eds.), *The semantic web: Eswc 2021 satellite events* (pp. 136–140). Springer International Publishing.

Araci, D. (2019). "Finbert: Financial sentiment analysis with pre-trained language models". *CoRR*, *abs/1908.10063*. http://arxiv.org/abs/1908.10063

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). "Algorithms for hyper-parameter optimization". In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems*. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf

Chan, Y. S., & Roth, D. (2011). "Exploiting syntactico-semantic structures for relation extraction". *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 551–560. https://aclanthology.org/P11-1056

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). "BERT: pre-training of deep bidirectional transformers for language understanding". *CoRR*, *abs/1810.04805*. http://arxiv.org/abs/1810.04805

Dong, W., Charikar, M., & Li, K. (2011). "Efficient k-nearest neighbor graph construction for generic similarity measures". *WWW*.

Eberts, M., & Ulges, A. (2019). "Span-based joint entity and relation extraction with transformer pre-training". *CoRR*, *abs/1909.07755*. http://arxiv.org/abs/1909.07755

Gu, J., Lu, Z., Li, H., & Li, V. O. K. (2016). "Incorporating copying mechanism in sequence-to-sequence learning". *CoRR*, *abs/1603.06393*. http://arxiv.org/abs/1603.06393

Gupta, H., Badugu, A., Agrawal, T., & Bhatt, H. S. (2021). "Zero-shot open information extraction using question generation and reading comprehension". *CoRR*, *abs/2109.08079*. https://arxiv.org/abs/2109.08079

Hasegawa, T., Sekine, S., & Grishman, R. (2004). "Discovering relations among named entities from large corpora". *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, 415–422.

Hoffmann, R., Zhang, C., Ling, X., Zettlemoyer, L., & Weld, D. S. (2011). "Knowledge-based weak supervision for information extraction of overlapping relations". *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 541–550. https://aclanthology.org/P11-1055

Honnibal, M., & Montani, I. (2017). "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing". [To appear].

Hope, T., Amini, A., Wadden, D., van Zuylen, M., Parasa, S., Horvitz, E., Weld, D., Schwartz, R., & Hajishirzi, H. (2021). "Extracting a knowledge base of mechanisms from COVID-19 papers". *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4489–4503.

Ji, B., Yu, J., Li, S., Ma, J., Wu, Q., Tan, Y., & Liu, H. (2020). "Span-based joint entity and relation extraction with attention-based span-specific and contextual semantic representations". *COLING*.

Katiyar, A., & Cardie, C. (2017). "Going out on a limb: Joint extraction of entity mentions and relations without dependency trees". *ACL*.

Lewis, C., & Young, S. (2019). "Fad or future? automated analysis of financial text and its implications for corporate reporting". *Accounting and Business Research*, *49*(5), 587–615.

Li, Q., & Ji, H. (2014). "Incremental joint extraction of entity mentions and relations". *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 402–412.

Li, X., Yin, F., Sun, Z., Li, X., Yuan, A., Chai, D., Zhou, M., & Li, J. (2019). "Entity-relation extraction as multi-turn question answering". *CoRR*, *abs/1905.05529*. http://arxiv.org/abs/1905.05529

Lin, C.-Y. (2004). "ROUGE: A package for automatic evaluation of summaries". *Text Summarization Branches Out*, 74–81. https://aclanthology.org/W04-1013

Lin, Y., Ji, H., Huang, F., & Wu, L. (2020). "A joint neural model for information extraction with global features". *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7999–8009.

Liu, Z., Jiang, F., Hu, Y., Shi, C., & Fung, P. (2021). "NER-BERT: A pre-trained model for low-resource entity tagging". *CoRR*, *abs/2112.00405*. https://arxiv.org/abs/2112.00405

Lopez de Lacalle, O., & Lapata, M. (2013). "Unsupervised relation extraction with general domain knowledge". *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 415–425. https://aclanthology.org/D13-1040

Luan, Y., Wadden, D., He, L., Shah, A., Ostendorf, M., & Hajishirzi, H. (2019). "A general framework for information extraction using dynamic span graphs". *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 3036–3046.

Marcheggiani, D., & Titov, I. (2016). "Discrete-state variational autoencoders for joint discovery and factorization of relations". *Transactions of the Association for Computational Linguistics*, *4*, 231–244.

Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009). "Distant supervision for relation extraction without labeled data". *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 1003–1011. https://aclanthology.org/P09-1113

Miwa, M., & Bansal, M. (2016). "End-to-end relation extraction using LSTMs on sequences and tree structures". *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1105–1116.

Montani, I., & Honnibal, M. (2018). "Prodigy: A new annotation tool for radically efficient machine teaching". *Artificial Intelligence*, *to appear*.

Nayak, T., & Ng, H. T. (2020). "Effective modeling of encoder-decoder architecture for joint entity and relation extraction". *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(05), 8528–8535.

Neelakantan, A., Xu, T., Puri, R., Radford, A., Han, J. M., Tworek, J., Yuan, Q., Tezak, N., Kim, J. W., Hallacy, C., Heidecke, J., Shyam, P., Power, B., Nekoul, T. E., Sastry, G., Krueger, G., Schnurr, D., Such, F. P., Hsu, K., ... Weng, L. (2022). "Text and code embeddings by contrastive pre-training". *CoRR*, *abs/2201.10005*. https://arxiv.org/abs/2201.10005

Nguyen, D. Q., & Verspoor, K. (2018). "An improved neural network model for joint POS tagging and dependency parsing". *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, 81–91.

Repke, T., & Krestel, R. (2021). "Extraction and representation of financial entities from text". In S. Consoli, D. Reforgiato Recupero, & M. Saisana (Eds.), *Data science for economics and finance: Methodologies and applications* (pp. 241–263). Springer International Publishing.

Reyes, D., Barcelos, A., Vieira, R., & Manssour, I. (2021). "Related named entities classification in the economic-financial context". *HACKASHOP*.

Shi, P., & Lin, J. (2019). "Simple bert models for relation extraction and semantic role labeling".

Simon, É., Guigue, V., & Piwowarski, B. (2019). "Unsupervised information extraction: Regularizing discriminative approaches with relation distribution losses". *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 1378–1387.

Stanovsky, G., Michael, J., Zettlemoyer, L., & Dagan, I. (2018). "Supervised open information extraction". *NAACL*.

Sui, D., Chen, Y., Liu, K., Zhao, J., Zeng, X., & Liu, S. (2020). "Joint entity and relation extraction with set prediction networks". *CoRR*, *abs/2011.01675*. https://arxiv.org/abs/2011.01675

Takanobu, R., Zhang, T., Liu, J., & Huang, M. (2018). "A hierarchical framework for relation extraction with reinforcement learning". *CoRR*, *abs/1811.03925*. http://arxiv.org/abs/1811.03925

Vela, M., & Declerck, T. (2009). "Concept and relation extraction in the finance domain". In H. Bunt, V. Petukhova, & S. Wubben (Eds.), *Proceedings of the eighth international conference on computational semantics (iwcs-8). international conference on computational semantics (iwcs-8), january 7-9, tilburg, netherlands* (pp. 346–351). Tilburg University.

Wadden, D., Wennberg, U., Luan, Y., & Hajishirzi, H. (2019). "Entity, relation, and event extraction with contextualized span representations". *ArXiv*, *abs/1909.03546*.

Wang, J., & Lu, W. (2020). "Two are better than one: Joint entity and relation extraction with table-sequence encoders". *ArXiv*, *abs/2010.03851*.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers". *CoRR*, *abs/2002.10957*. https://arxiv.org/abs/2002.10957

Wei, Z., Su, J., Wang, Y., Tian, Y., & Chang, Y. (2020). "A novel cascade binary tagging framework for relational triple extraction".

Yao, L., Haghighi, A., Riedel, S., & McCallum, A. (2011). "Structured relation discovery using generative models". *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 1456–1466. https://aclanthology.org/D11-1135

Yu, B., Zhang, Z., Su, J., Wang, Y., Liu, T., Wang, B., & Li, S. (2019). "Joint extraction of entities and relations based on a novel decomposition strategy". *CoRR*, *abs/1909.04273*. http://arxiv.org/abs/1909.04273

Yuan, C., Rossi, R. A., Katz, A., & Eldardiry, H. (2020). "Clustering-based unsupervised generative relation extraction". *CoRR*, *abs/2009.12681*. https://arxiv.org/abs/2009.12681

Zelenko, D., Aone, C., & Richardella, A. (2002). "Kernel methods for relation extraction". *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, 71–78.

Zeng, D., Liu, K., Lai, S., Zhou, G., & Zhao, J. (2014). "Relation classification via convolutional deep neural network". *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2335–2344. https://aclanthology.org/C14-1220

Zeng, D., Zhang, H., & Liu, Q. (2019). "Copymtl: Copy mechanism for joint extraction of entities and relations with multi-task learning". *CoRR*, *abs/1911.10438*. http://arxiv.org/abs/1911.10438

Zeng, X., Zeng, D., He, S., Kang, L., & Jun, Z. (2018). "Extracting relational facts by an end-to-end neural model with copy mechanism". *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 506–514.

Zheng, S., Wang, F., Bao, H., Hao, Y., Zhou, P., & Xu, B. (2017). "Joint extraction of entities and relations based on a novel tagging scheme".

Zhou, Z., & Zhang, H. (2018). "Research on entity relationship extraction in financial and economic field based on deep learning". *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*, 2430–2435.

Appendix A

Code Repository

Our search engine (FinSearch), datasets (FinMechanic and FinSemantic), and code are publicly available.

Table A.1: Code Repository Links

FinSearch	http://finsearch.surge.sh
FinMechanic Dataset	github.com/ValaryLim/finsearchIE/dataset/finmechanic
FinSemantic Dataset	github.com/ValaryLim/finsearchIE/dataset/finsemantic
Code	github.com/ValaryLim/finsearchIE

Appendix B

Annotation Guidelines

B.1 Relation Examples

In Table B.1, we provide some examples of the relational triples extracted for each type of sub-relation. The text highlighted in green refers to E_1 and the text in pink refers to E_2.

Table B.1: Relation Examples

Relation	**Sub-Relation**	**Example**
Direct	Attribute	Growth of U.S. agriculture is determined by the economic environment, which consists of the available technology, incentives, and institutions. (*available technology*; `ATTRIBUTE`; *economic environment*) (*incentives*; `ATTRIBUTE`; *economic environment*) (*institutions*; `ATTRIBUTE`; *economic environment*)
	Function	*Tail events* are modelled using the *generalised Pareto distribution*. (*generalised Pareto distribution*; `FUNCTION`; *Tail events*)

Indirect	Positive	*Persistence* *is negatively related to the rate of growth in GDP per capita, and positively related to the* *size of entry barriers* *.* (*Persistence*; `POSITIVE`; *size of entry barriers*)
	Negative	*Persistence* *is negatively related to the* *rate of growth in GDP per capita* *, and positively related to the size of entry barriers.* (*Persistence*; `NEGATIVE`; *rate of growth in GDP per capita*)
	Neutral	*Our detailed empirical findings indicate that* *CEO characteristics* *and* *Top Management Team (TMT) characteristics* *show a nonlinear relationship.* (*CEO characteristics*; `NEUTRAL`; *Top Management Team (TMT) characteristics*)
	None	*Our findings show that there is no significant relation between* *interest rate uncertainty* *and* *intermediary exposure* *.* (*interest rate uncertainty*; `NONE`; *intermediary exposure*)
	Condition	*We obtain* *negatively biased brand constants* *when* *consumer heterogeneity in choice sets is ignored* *.* (*consumer heterogeneity in choice sets is ignored*; `CONDITION`; *negatively biased brand constants*)
	Comparison	*Results from a simulation study indicate that* *our tests* *significantly outperform* *competing backtests in several distinct settings* *.* (*our tests*; `COMPARISON`; *competing backtests in several distinct settings*)

	Uncertain	*We study the effects of capital account liberalization on firm capital allocation.* (*capital account liberalization*; `UNCERTAIN`; *firm capital allocation*)

B.2 Example Annotations

Figures B.1 and B.2 are screenshots of the annotated FinMechanic dataset on Prodigy. We use SpaCy's `en_core_web_sm` to tokenize the sentences and Prodigy's `relations` interface to label the entity-relation triplets. On the software, entities are represented by a closed box, while relations are represented by an arrow pointing from E_1 to E_2 and have a corresponding relation label as indicatd by the text on each arrow.

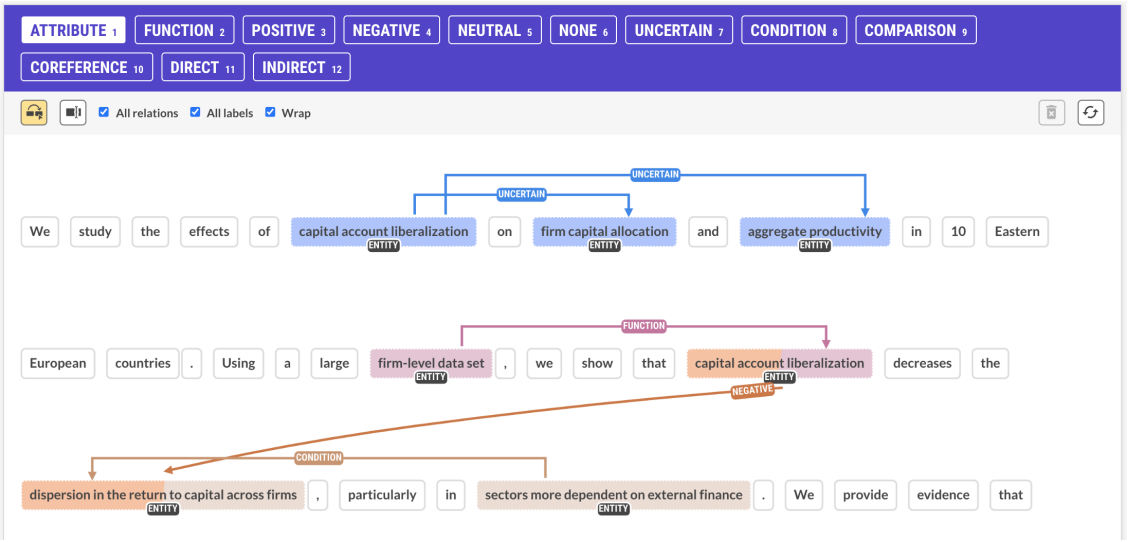


Figure B.1: Prodigy Annotation Example (1)

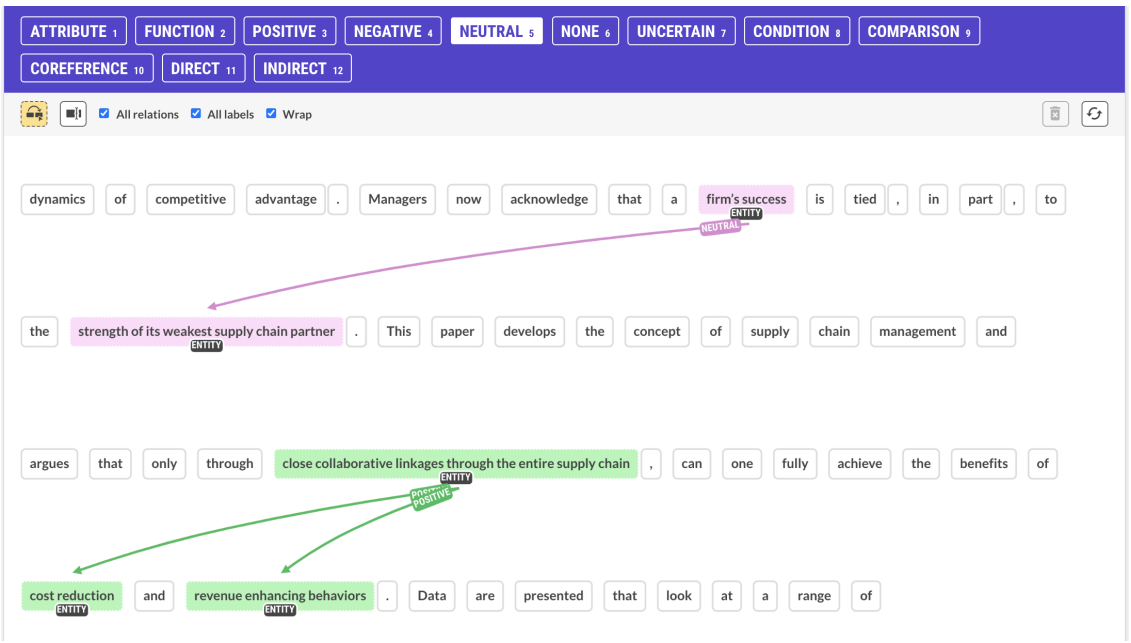


Figure B.2: Prodigy Annotation Example (2)

Appendix C

DyFinIE Evaluation Metrics

C.1 Guidelines

In Table C.1, E_i and R refer to the predicted entities and relation labels, while E_i^{gold} and R^{gold} refer to the gold standard entities and relation labels respectively. T_i refers to token i in an abstract.

Table C.1: Conditions for Prediction to be Marked Correct

Computation Method	Conditions for Correct Prediction (True Positive)
Exact Relational Triplet (ExactRel)	Correct if both predicted E_1 and E_2 are identical to their corresponding gold-standard entities, E_1^{gold} and E_2^{gold}, and the predicted relation label is the same as the gold label. $[(E_1 = E_1^{gold}) \ \& \ (E_2 = E_2^{gold}) \ \& \ (R = R^{gold})]$
Exact Relational Triplet No-Direction (ExactRelND)	Correct if the set of predicted entities is identical to the set of gold-standard entities and the predicted relation label is the same as the gold label. $[(E_1 = E_1^{gold}) \ \& \ (E_2 = E_2^{gold}) \ \& \ (R = R^{gold})]$ or $[(E_1 = E_2^{gold}) \ \& \ (E_2 = E_1^{gold}) \ \& \ (R = R^{gold})]$

Fuzzy Relational Triplet (FuzzyRel)	Correct if the gold-standard entities E_1^{gold} and E_2^{gold} are each contained within the predicted entities E_1 and E_2, the number of tokens in predicted entities do not exceed that of the gold-standard entities by more than 5 tokens, and the predicted relation label is the same as the gold label. `[(`E_1^{gold} `in` $E1$`) & (`E_2^{gold} `in` E_2`) & (`R `=` R^{gold}`) & (`$\lambda(E_1)$`-`$\lambda(E_1^{gold})$`) < 5) & (`$\lambda(E_2)$`-`$\lambda(E_2^{gold})$`) < 5)]`, where $\lambda(E_i)$ refers to the number of tokens in E_i.
Fuzzy Relational Triplet No-Direction (FuzzyRelND)	Correct if the set of gold-standard entities is contained within the set predicted entities, the number of tokens in each predicted entity does not exceed that of the gold-standard entities by more than 5 tokens, and the predicted relation label is the same as the gold label. `[(`E_1^{gold} `in` $E1$`) & (`E_2^{gold} `in` E_2`) & (`R `=` R^{gold}`) & (`$\lambda(E_1)$`-`$\lambda(E_1^{gold})$`) < `θ`) & (`$\lambda(E_2)$`-`$\lambda(E_2^{gold})$`) < `θ`)]` or `[(`E_2^{gold} `in` E_1`) & (`E_1^{gold} `in` E_2`) & (`R `=` R^{gold}`) & (`$\lambda(E_1)$`-`$\lambda(E_2^{gold})$`) < `θ`) & (`$\lambda(E_2)$`-`$\lambda(E_1^{gold})$`) < `θ`)]`
Rouge Relational Triplet (RougeRel)	Correct if the rouge score between the predicted entities and their corresponding gold-standard entities are greater than 0.5 (similar to each other) and the predicted relation label is the same as the gold label. `[rouge(`E_1`1, `E_1^{gold}`) > 0.5 & rouge(`E_2`, `E_2^{gold}`) > 0.5 & (`R `=` R^{gold}`)]`

Rouge Relational Triplet No-Direction (RougeRelND)	Correct if the rouge score between the predicted entities and either gold-standard entities are greater than 0.5 (similar to each other) and the predicted relation label is the same as the gold label. `[rouge(`$E1$`, `E_1^{gold}`) > 0.5 & rouge(`E_2`, `E_2^{gold}`) > 0.5 & (`R` = `R^{gold}`)]` or `[rouge(`E_2`, `E_1^{gold}`) > 0.5 & rouge(`E_1`, `E_2^{gold}`) > 0.5 & (`R` = `R^{gold}`)]`
Jaccard Relational Triplet (JaccardRel)	Correct if the Jaccard index between the predicted entities and their corresponding gold-standard entities are greater than 0.5 (similar to each other) and the predicted relation label is the same as the gold label. `[J(`E_1`1, `E_1^{gold}`) > 0.5 & J(`E_2`, `E_2^{gold}`) > 0.5 & (`R` = `R^{gold}`)]`
Jaccard Relational Triplet No-Direction (JaccardRelND)	Correct if the Jaccard index between the predicted entities and either gold-standard entities are greater than 0.5 (similar to each other) and the predicted relation label is the same as the gold label. `[J(`$E1$`, `E_1^{gold}`) > 0.5 & J(`E_2`, `E_2^{gold}`) > 0.5 & (`R` = `R^{gold}`)]` or `[J(`E_2`, `E_1^{gold}`) > 0.5 & J(`E_1`, `E_2^{gold}`) > 0.5 & (`R` = `R^{gold}`)]`
Exact Span (ExactSpan)	Correct if both predicted E_1 and E_2 are identical to their corresponding gold-standard entities, E_1^{gold} and E_2^{gold}. `[(`E_1` = `E_1^{gold}`) & (`E_2` = `E_2^{gold}`)]`
Exact Span No-Direction (ExactSpanND)	Correct if the set of predicted entities is identical to the set of gold-standard entities. `[(`E_1` = `E_1^{gold}`) & (`E_2` = `E_2^{gold}`)]` or `[(`E_1` = `E_2^{gold}`) & (`E_2` = `E_1^{gold}`)]`

Fuzzy Span (FuzzySpan)	Correct if the gold-standard entities E_1^{gold} and E_2^{gold} are each contained within the predicted entities E_1 and E_2, the number of tokens in predicted entities do not exceed that of the gold-standard entities by more than 5 tokens. `[(`E_1^{gold} `in` $E1$`) & (`E_2^{gold} `in` E_2`) & (`$\lambda(E_1)$`-`$\lambda(E_1^{gold})$`) <` θ`) & (`$\lambda(E_2)$`-`$\lambda(E_2^{gold})$`) <` θ`)]`
Fuzzy Span No-Direction (FuzzySpanND)	Correct if the set of gold-standard entities is contained within the set predicted entities, the number of tokens in each predicted entity does not exceed that of the gold-standard entities by more than 5 tokens. `[(`E_1^{gold} `in` $E1$`) & (`E_2^{gold} `in` E_2`) & (`$\lambda(E_1)$`-`$\lambda(E_1^{gold})$`) <` θ`) & (`$\lambda(E_2)$`-`$\lambda(E_2^{gold})$`) <` θ`)]` or `[(`E_2^{gold} `in` E_1`) & (`E_1^{gold} `in` E_2`) & (`$\lambda(E_1)$`-`$\lambda(E_2^{gold})$`) <` θ`) & (`$\lambda(E_2)$`-`$\lambda(E_1^{gold})$`) <` θ`)]`
Rouge Span (RougeSpan)	Correct if the rouge score between the predicted entities and their corresponding gold-standard entities are greater than 0.5 (similar to each other). `[rouge(`E_1`1,` E_1^{gold}`) > 0.5 & rouge(`E_2`,` E_2^{gold}`) > 0.5]`
Rouge Span No-Direction (RougeSpanND)	Correct if the rouge score between the predicted entities and either gold-standard entities are greater than 0.5 (similar to each other). `[rouge(`$E1$`,` E_1^{gold}`) > 0.5 & rouge(`E_2`,` E_2^{gold}`) > 0.5]` or `[rouge(`E_2`,` E_1^{gold}`) > 0.5 & rouge(`E_1`,` E_2^{gold}`) > 0.5]`
Jaccard Span (JaccardSpan)	Correct if the Jaccard index between the predicted entities and their corresponding gold-standard entities are greater than 0.5 (similar to each other). `[J(`E_1`1,` E_1^{gold}`) > 0.5 & J(`E_2`,` E_2^{gold}`) > 0.5]`

Jaccard Span No-Direction (JaccardSpanND)	Correct if the Jaccard index between the predicted entities and either gold-standard entities are greater than 0.5 (similar to each other). `[J(`$E1$`,` E_1^{gold}`) > 0.5 & J(`E_2`,` E_2^{gold}`) > 0.5]` or `[J(`E_2`,` E_1^{gold}`) > 0.5 & J(`E_1`,` E_2^{gold}`) > 0.5]`
Exact Token (ExactToken)	Each **token** is marked as correctly predicted if the token does not belong to any entity span in both the predicted and gold-standard labels, or the token belongs to an entity span in both the predicted and gold-standard. `[`T_i `==` T_i^{gold}`]`

C.2 Metric Examples

As an example, we will use the sentence 'We find that fiscal discipline is essential to improving and sustaining economic performance.', where the gold relation triplet to be extracted is (*fiscal discipline*; `POSITIVE`; *economic performance*). In Tables C.2 - C.3, we identify possible predictions of our IE models and indicate whether each prediction would be marked correct (✓) or incorrect (✗). The differences between the prediction and gold standard are underlined.

Table C.2: Conditions for Prediction to be Marked Correct

Prediction	ExactRel	ExactRelND	FuzzyRel	FuzzyRelND	RougeRel	RougeRelND	JaccardRel	JaccardRelND
(*fiscal discipline*; `POSITIVE`; *economic performance*)	✓	✓	✓	✓	✓	✓	✓	✓
(*economic performance*; `POSITIVE`; *fiscal discipline*)	✗	✓	✗	✓	✗	✓	✗	✓
(*fiscal discipline*; `POSITIVE`; *sustaining economic performance*)	✗	✗	✓	✓	✓	✓	✓	✓
(*sustaining economic performance*; `POSITIVE`; *fiscal discipline*)	✗	✗	✗	✓	✗	✓	✗	✓
(*fiscal discipline*; `POSITIVE`; *_ performance*)	✗	✗	✗	✗	✓	✓	✓	✓
(*_ performance*; `POSITIVE`; *fiscal discipline*)	✗	✗	✗	✗	✗	✓	✗	✓
(*fiscal discipline*; `NEGATIVE`; *economic performance*)	✗	✗	✗	✗	✗	✗	✗	✗
(*economic performance*; `NEGATIVE`; *fiscal discipline*)	✗	✗	✗	✗	✗	✗	✗	✗
(*fiscal discipline*; `NEGATIVE`; *sustaining economic performance*)	✗	✗	✗	✗	✗	✗	✗	✗
(*sustaining economic performance*; `NEGATIVE`; *fiscal discipline*)	✗	✗	✗	✗	✗	✗	✗	✗
(*fiscal discipline*; `NEGATIVE`; *_ performance*)	✗	✗	✗	✗	✗	✗	✗	✗
(*_ performance*; `NEGATIVE`; *fiscal discipline*)	✗	✗	✗	✗	✗	✗	✗	✗

Table C.3: Conditions for Prediction to be Marked Correct

Prediction	ExactSpan	ExactSpanND	FuzzySpan	FuzzySpanND	RougeSpan	RougeSpanND	JaccardSpan	JaccardSpanND
(*fiscal discipline*; POSITIVE; *economic performance*)	✓	✓	✓	✓	✓	✓	✓	✓
(*economic performance*; POSITIVE; *fiscal discipline*)	✗	✓	✗	✓	✗	✓	✗	✓
(*fiscal discipline*; POSITIVE; *sustaining economic performance*)	✗	✗	✓	✓	✓	✓	✓	✓
(*sustaining economic performance*; POSITIVE; *fiscal discipline*)	✗	✗	✗	✓	✗	✓	✗	✓
(*fiscal discipline*; POSITIVE; _ *performance*)	✗	✗	✗	✗	✓	✓	✓	✓
(_ *performance*; POSITIVE; *fiscal discipline*)	✗	✗	✗	✗	✗	✓	✗	✓
(*fiscal discipline*; NEGATIVE; *economic performance*)	✓	✓	✓	✓	✓	✓	✓	✓
(*economic performance*; NEGATIVE; *fiscal discipline*)	✗	✓	✗	✓	✗	✓	✗	✓
(*fiscal discipline*; NEGATIVE; *sustaining economic performance*)	✗	✗	✓	✓	✓	✓	✓	✓
(*sustaining economic performance*; NEGATIVE; *fiscal discipline*)	✗	✗	✗	✓	✗	✓	✗	✓
(*fiscal discipline*; NEGATIVE; _ *performance*)	✗	✗	✗	✗	✓	✓	✓	✓
(_ *performance*; NEGATIVE; *fiscal discipline*)	✗	✗	✗	✗	✗	✓	✗	✓

Appendix D

DyFinIE Experiment Results

D.1 BERT vs FinBERT Embedding

Tables D.1 - D.20 compare the results of the DyFinIE model using BERT and FinBERT models to generate context embeddings. We find that FinBERT achieves a higher precision, recall and F1 score across all metric types and datasets (both FinMechanic and External). Our results strongly suggest the benefit of using a specially trained BERT model for financial text (FinBERT), over the default BERT model, when training models for the financial domain.

D.1.1 FinMechanic (Coarse Relations)

Table D.1: Performance on FinMechanic Dataset (Coarse Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	26.9	17.3	21.0	29.9	19.2	23.4	31.3	20.1	24.4	34.2	22.0	26.8
FinBERT	34.9	18.7	24.3	37.5	20.1	26.1	39.4	21.1	27.5	42.0	22.5	29.3

Table D.2: Performance on FinMechanic Dataset (Coarse Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	38.9	25.0	30.4	43.5	27.9	34.0	31.3	20.1	24.4	34.5	22.2	27.0
FinBERT	48.2	25.8	33.6	53.7	28.8	37.5	42.0	22.5	29.3	45.9	24.6	32.0

Table D.3: Performance on FinMechanic Dataset (Coarse Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	29.6	19.0	23.2	33.7	21.6	26.4	35.1	22.5	27.4	39.9	25.7	31.2
FinBERT	36.5	19.5	25.5	39.7	21.3	27.7	41.0	22.0	28.6	44.6	23.9	31.1

Table D.4: Performance on FinMechanic Dataset (Coarse Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	43.5	27.9	34.0	51.1	32.8	40.0	35.1	22.5	27.4	39.9	25.7	31.2
FinBERT	50.8	27.2	35.5	58.6	31.4	40.9	44.0	23.6	30.7	49.8	26.7	34.8

Table D.5: Performance on FinMechanic Dataset (Coarse Relations) (5)

	Token		
Encoder	P	R	F1
BERT	82.2	38.5	52.4
FinBERT	86.4	34.3	49.1

D.1.2 FinMechanic (Granular Relations)

Table D.6: Performance on FinMechanic Dataset (Granular Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	29.3	11.3	16.4	32.0	12.4	17.9	37.8	14.7	21.1	40.5	15.7	22.6
FinBERT	29.3	13.6	18.6	31.2	14.5	19.8	33.1	15.4	21.0	35.3	16.4	22.4

Table D.7: Performance on FinMechanic Dataset (Granular Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	41.0	15.9	22.9	45.9	17.8	25.7	35.6	13.8	19.9	40.1	15.5	22.4
FinBERT	40.2	18.7	25.5	45.1	20.9	28.6	35.0	16.2	22.2	38.7	18.0	24.6

Table D.8: Performance on FinMechanic Dataset (Granular Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	39.6	15.4	22.1	44.6	17.3	24.9	49.1	19.0	27.4	54.1	20.9	30.2
FinBERT	38.0	17.6	24.1	41.7	19.4	26.5	42.1	19.5	26.7	46.2	21.5	29.3

Table D.9: Performance on FinMechanic Dataset (Granular Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	53.6	20.8	29.9	62.6	24.3	35.0	47.3	18.3	26.4	55.0	21.3	30.7
FinBERT	51.5	23.9	32.7	60.2	27.9	38.1	44.0	20.4	27.9	50.4	23.4	31.9

Table D.10: Performance on FinMechanic Dataset (Granular Relations) (5)

	Token		
Encoder	P	R	F1
BERT	85.0	27.6	41.7
FinBERT	89.4	31.0	46.0

D.1.3 External (Coarse Relations)

Table D.11: Performance on External Dataset (Coarse Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	3.0	4.8	3.7	3.0	4.8	3.7	3.0	4.8	3.7	3.8	6.0	4.7
FinBERT	3.8	7.2	5.0	3.8	7.2	5.0	3.8	7.2	5.0	3.8	7.2	5.0

Table D.12: Performance on External Dataset (Coarse Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	9.8	15.7	12.1	12.1	19.3	14.9	8.3	13.3	10.2	10.6	16.9	13.0
FinBERT	12.6	24.1	16.5	13.8	26.5	18.2	8.2	15.7	10.7	9.4	18.1	12.4

Table D.13: Performance on External Dataset (Coarse Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	3.0	4.8	3.7	3.0	4.8	3.7	3.0	4.8	3.7	3.8	6.0	4.7
FinBERT	3.8	7.2	5.0	4.4	8.4	5.8	3.8	7.2	5.0	4.4	8.4	5.8

Table D.14: Performance on External Dataset (Coarse Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	9.8	15.7	12.1	13.6	21.7	16.7	8.3	13.3	10.2	10.6	16.9	13.0
FinBERT	12.6	24.1	16.5	15.1	28.9	19.8	8.2	15.7	10.7	10.7	20.5	14.0

Table D.15: Performance on External Dataset (Coarse Relations) (5)

	Token		
Encoder	P	R	F1
BERT	36.5	24.0	29.0
FinBERT	36.2	27.7	31.4

D.1.4 External (Granular Relations)

Table D.16: Performance on External Dataset (Granular Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	1.8	2.4	2.1	1.9	2.4	2.1	1.9	2.4	2.1	1.9	2.4	2.1
FinBERT	2.1	3.6	2.6	2.1	3.6	2.6	2.1	3.6	2.6	2.1	3.6	2.6

Table D.17: Performance on External Dataset (Granular Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	9.4	12.0	10.6	9.4	12.0	10.6	6.6	8.4	7.4	6.6	8.4	7.4
FinBERT	8.9	15.7	11.4	8.9	15.7	11.4	4.1	7.2	5.2	4.1	7.2	5.2

Table D.18: Performance on External Dataset (Granular Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	4.7	6.0	5.3	4.7	6.0	5.3	4.7	6.0	5.3	4.7	6.0	5.3
FinBERT	2.7	4.8	3.5	3.4	6.0	4.4	2.7	4.8	3.5	3.4	6.0	4.4

Table D.19: Performance on External Dataset (Granular Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Encoder	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	13.2	16.9	14.8	14.2	18.1	15.9	10.4	13.3	11.6	11.3	14.5	12.7
FinBERT	13.0	22.9	16.6	15.1	26.5	19.2	8.2	14.5	10.5	10.3	18.1	13.1

Table D.20: Performance on External Dataset (Granular Relations) (5)

	Token		
Encoder	P	R	F1
BERT	38.8	22.8	28.7
FinBERT	37.4	25.8	30.6

D.2 DyFinIE vs Off-the-Shelf NER Models

In this section, we compare the performance of DyFinIE in extracting entities with several off-the-shelf NER models.

D.2.1 Baselines

We compare DyFinIE with the following off-the-shelf NER models:

- **SpaCy.** The spaCy NER recogniser is able to identify a total of 18 types of entities, including person names, locations, organizations, time expressions, quantities, and monetary values. The NER model feeds subword features and "Bloom" embeddings into a deep convolutional neural network.
- **NER-BERT.** The NER-BERT model proposed by Z. Liu et al. (2021) is based on the BERT (Devlin et al., 2018) and trained using a large corpus of 475.6 million tokens. The model is able to retrieve 315 different categories, including organizations, locations, jobs, and language.
- **NLTK.** The NLTK model identifies nouns in text by first performing part-of-speech tagging on sentences, and subsequently using rule and pattern-based logic to chunk the tokens (words) into phrases, and to identify the specific phrases corresponding to nouns. For the purpose of this evaluation, we take all nouns as candidate entities retrieved by NLTK.
- **Stanford NER.** The Stanford NER model recognises and annotates named (persons, locations, organisations), ordinal (money, percentages), and temporal (time, duration) entities. The model combines specialised rule-based components and sequence learning models to label entities and interpret ordinal and temporal values.

D.2.2 Results

We utilise the same metrics for determining a correct entity as that mentioned in Section 5.2, and compute the number of True Positives (TP), False Positives (FP), and False Negatives (FN) for the predicted entities retrieved from each model. As our objective is for our model to retrieve as many correct entities and as little wrong entities as possible, the ideal model will have a high TP score, and low FP and FN scores. Table D.21 shows the performance of each NER model on our FinMechanic test set. The bolded values indicate the best model for the metric.

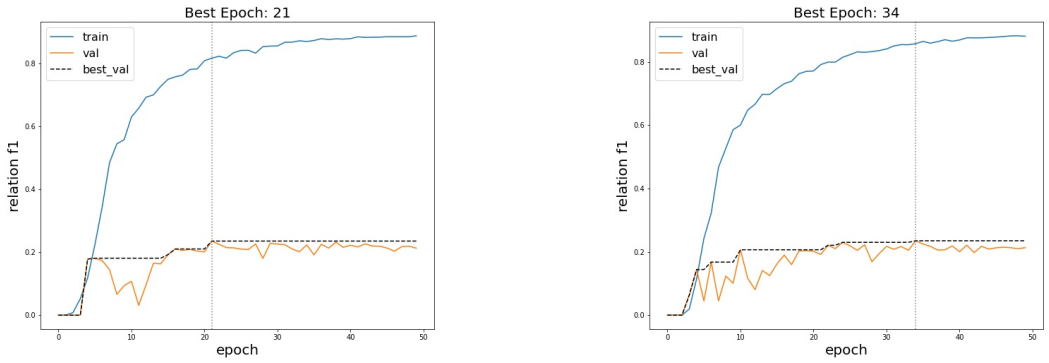
We find that DyFinIE consistently has the highest TP and lowest FN values across all four correct entity computation methods, but has the worst FP. These findings imply that DyFinIE successfully flags out more correct entities at the cost of retrieving a greater number of wrong entities. These results are expected as the DyFinIE model does not have a set classification of entity types that it retrieves, unlike the baseline models. However, as our pipeline involves further filtering for relevant relations and abstracts at the FinSearch service-level, we believe that selecting a model with higher recall is more important than a model with higher accuracy, and are therefore willing to accept the trade-off between a higher TP and lower FP score.

Table D.21: NER Model Performance on FinMechanic Dataset

NER Model	Exact			Fuzzy			Jaccard			Rouge		
	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
SpaCy	0	285	16	9	276	13	0	285	16	3	282	16
NER-BERT	1	211	16	14	198	14	0	212	16	3	282	16
NLTK	1	216	16	18	199	14	0	217	16	1	211	16
Stanford	0	**118**	16	11	**107**	14	0	**118**	16	0	**118**	16
DyFinIE	**26**	780	**7**	**48**	758	**3**	**39**	767	**6**	46	760	**3**

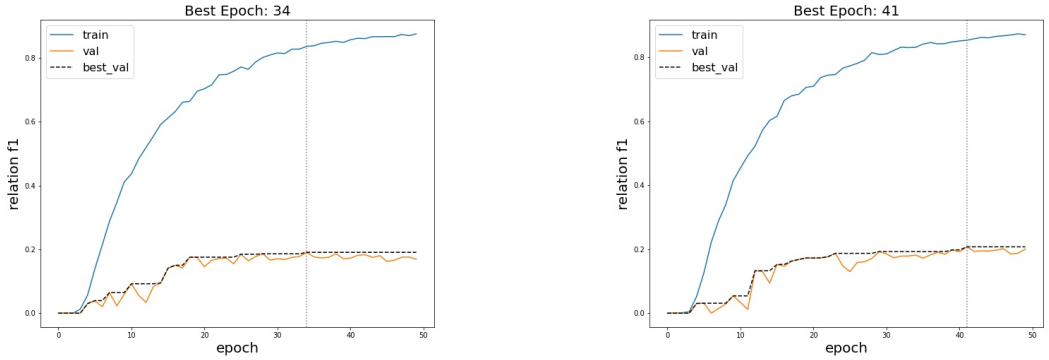
D.3 Validation Curves

Figures D.1 and D.2 show the train and validation F_1 scores for our DyFinIE models. The final models selected were the models as of the best epoch step, indicated by the vertical dotted grey lines in each graph. We note that there is a significant fall in F_1 score across all models from the train to validation sets, likely due to our small train set size.



(a) Coarse Bert (b) Coarse Finbert

Figure D.1: FinMechanic Coarse Validation Curves



(a) Granular Bert (b) Granular Finbert

Figure D.2: FinMechanic Granular Validation Curves

D.4 DyFinIE Prediction Examples

In this section, we provide some examples of correct and incorrect predictions. Similar to the Prodigy annotations, each colored box refers to a contiguous entity span and arrows denote the relations between the entities. Entities that were identified correctly but are not linked to a relation tag are colored in grey, while relations that are incorrectly identified are highlighted in red.

D.4.1 Correct

Figures D.3 and D.4 illustrate examples where our DyFinIE model correctly identifies the entire relational triplet (in accordance with ExactRel criteria).

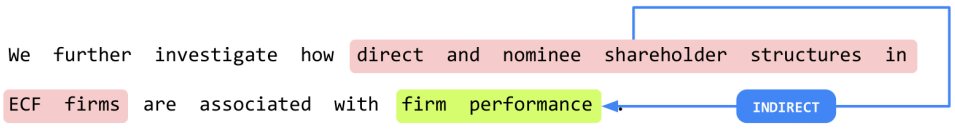


Figure D.3: Correct Prediction (1)

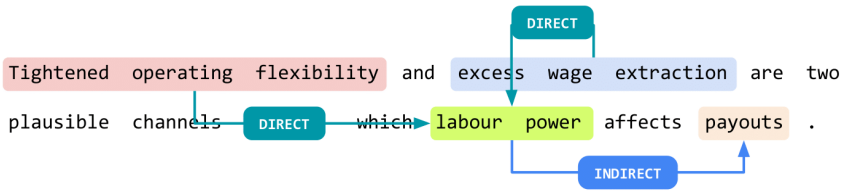
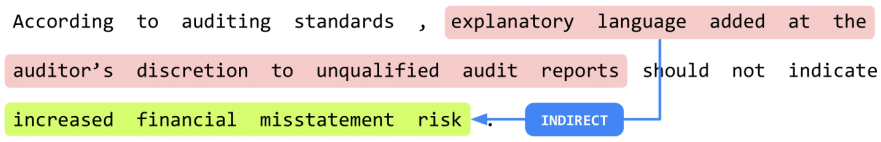


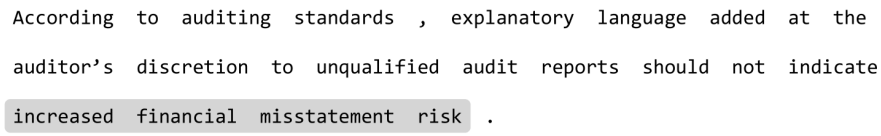
Figure D.4: Correct Prediction (2)

D.4.2 Incorrect

Figure D.5 and D.6 are some examples where DyFinIE fails to identify the entire relational triplet. In Figure D.6, our model both fails to identify the correct entity spans ('*positive skewness in their return distributions*') and wrongly identifies relations between two unrelated entities ('*glamour stocks*' and '*return distributions of value stocks*'). We observe that compared to the correct examples provided in Section D.4.1, the incorrect cases tend to be longer and contain multiple clauses, adding complexity to the relation-extraction problem.

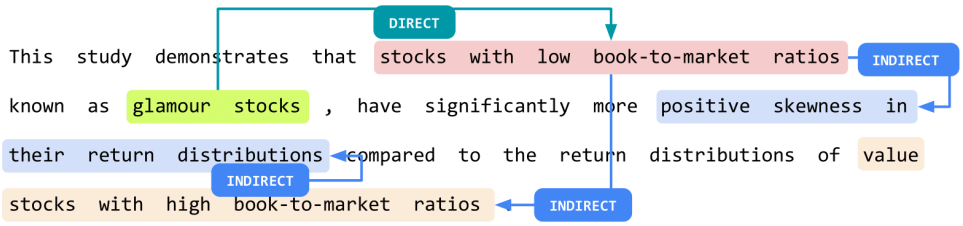


(a) Gold Standard

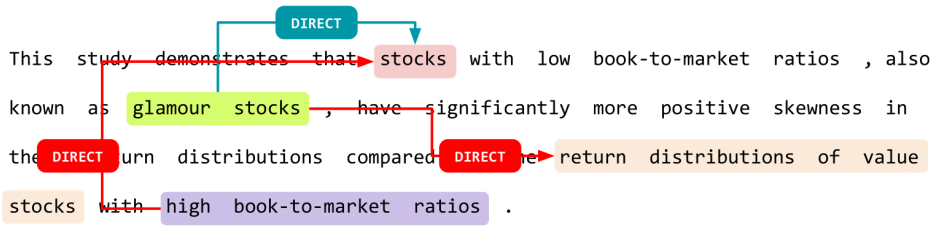


(b) Prediction

Figure D.5: Incorrect Prediction (1)



(a) Gold Standard

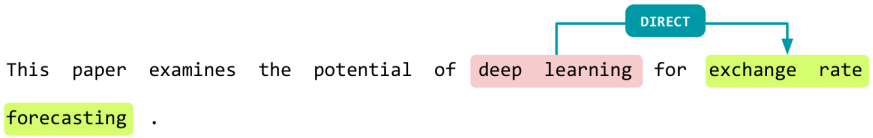


(b) Prediction

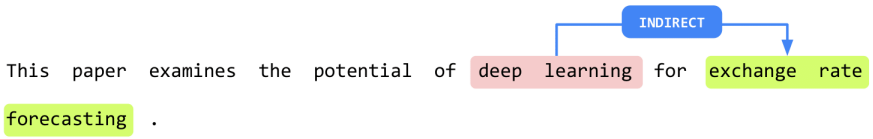
Figure D.6: Incorrect Prediction (2)

D.4.3 Partially Correct - Incorrect Relation Label

We find that our model often generates partially correct relational triplets from each input sentence. In the following sections, we provide examples of common mistakes made by DyFinIE. First, we observe many instances where entities are correctly identified but the relation label derived is wrong, explaining the significant difference in performance between the **Rel** metrics that checks for relation labels, and the **Span** metrics that only consider if entities are correctly identified. Figure D.7 is one example of an incorrect relation label.



(a) Gold Standard

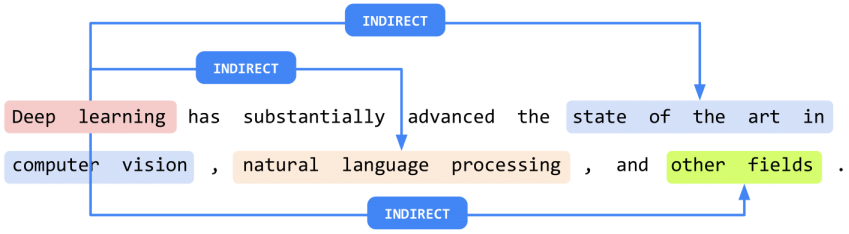


(b) Prediction

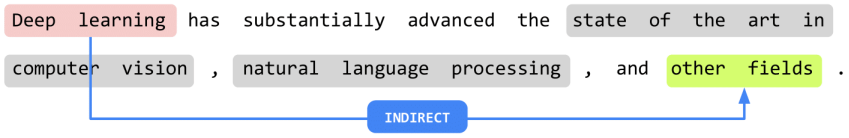
Figure D.7: Incorrect Relation Label

D.4.4 Partially Correct - Missing Relation Label

We also find that our DyFinIE model often correctly identifies entity spans, but fails to label the relationship between the entity spans.

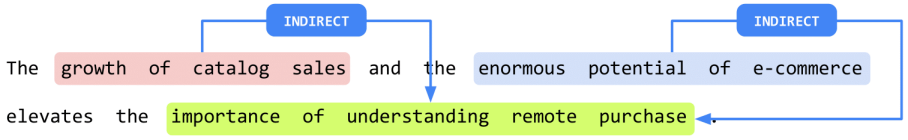


(a) Gold Standard

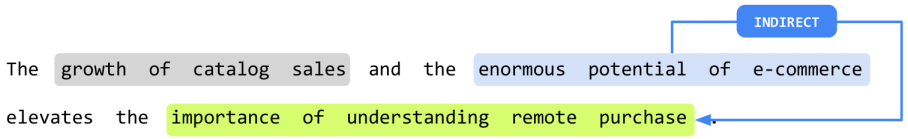


(b) Prediction

Figure D.8: Missing Relation Label (1)

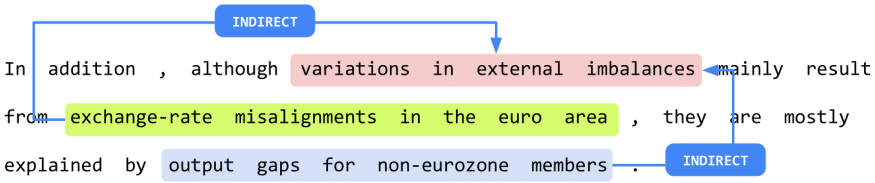


(a) Gold Standard

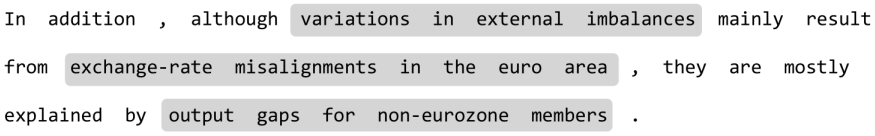


(b) Prediction

Figure D.9: Missing Relation Label (2)



(a) Gold Standard

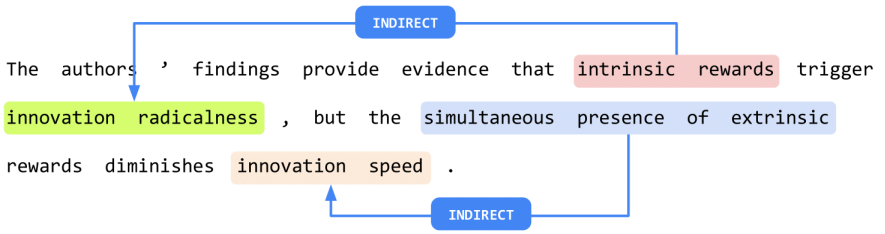


(b) Prediction

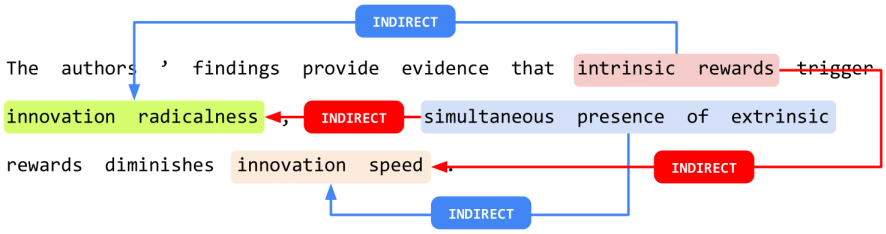
Figure D.10: Missing Relation Label (3)

D.4.5 Partially Correct - Extra Relations

Alternatively, there are also cases where our model correctly identifies all existing relational triplets, but flags out relations between two entities that should be uncorrelated.

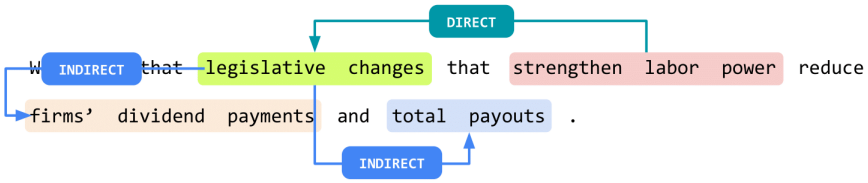


(a) Gold Standard

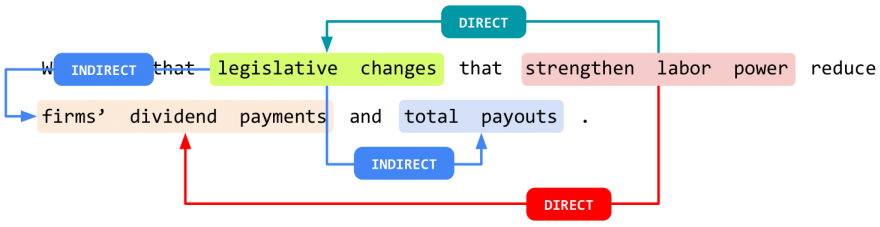


(b) Prediction

Figure D.11: Extra Relation (1)



(a) Gold Standard

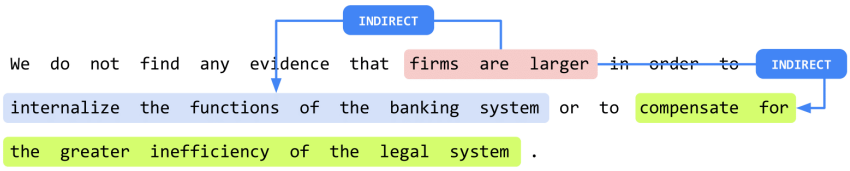


(b) Prediction

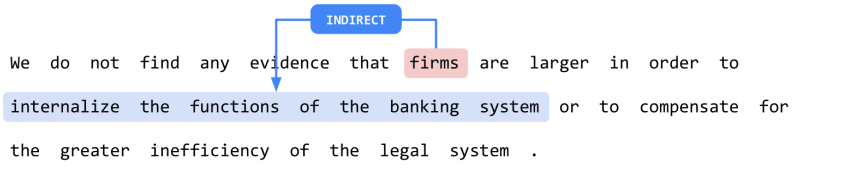
Figure D.12: Extra Relation (2)

D.4.6 Partially Correct - Missing Entities

We also observe that for sentences where there is a many-to-one relation between entity spans (multiple entities are listed consecutively and compared to the same one entity), that our DyFinIE model tends to omit a few entities. For example, in Figure D.14, 5 universities were listed in the text but only 3 were successfully identified and 1 was partially identified.

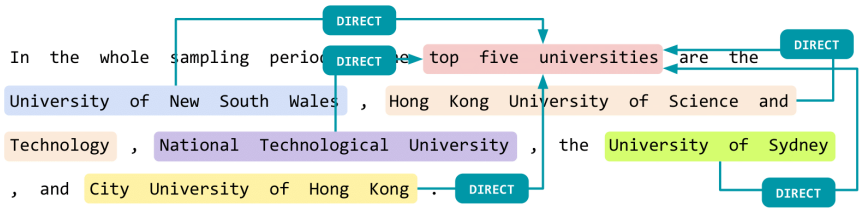


(a) Gold Standard

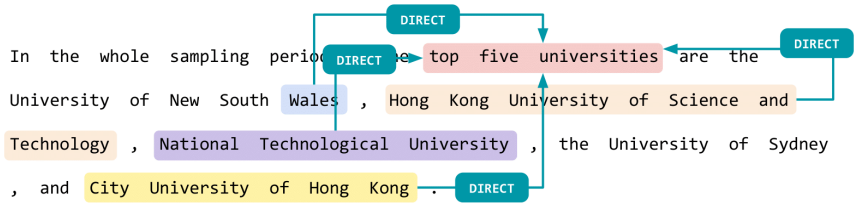


(b) Prediction

Figure D.13: Missing Entity (1)



(a) Gold Standard



(b) Prediction

Figure D.14: Missing Entity (2)

Appendix E

DyFinIE External Dataset Results

This dataset consists of 37 abstracts from various journals (Accounting Review, Contemporary Accounting Research, Financial Analysts Journal, Financial Management, International Journal of Finance & Economics, Journal of Accounting & Economics, Journal of Banking & Finance, Review of Financial Studies, and Journal of Behavioural Finance). These abstracts were hand-labelled by a volunteer who is an expert in finance. We loaded the dataset into Prodigy and replicated the hand-labelled annotations to fit our relation schema, generating External Coarse and External Granular datasets. We test both datasets on our DyFinIE model, and compare them against benchmarks OpenIE and SRL.

E.1 Results

Tables E.1 - E.10 reflect the performance of the baselines and our model on the external dataset. Further discussion of the results can be found in Section E.2.

E.1.1 External (Coarse Relations)

Table E.1: Performance on External Dataset (Coarse Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	3.8	7.2	5.0	3.8	7.2	5.0	3.8	7.2	5.0	3.8	7.2	5.0

Table E.2: Performance on External Dataset (Coarse Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	12.6	24.1	16.5	13.8	26.5	18.2	8.2	15.7	10.7	9.4	18.1	12.4

Table E.3: Performance on External Dataset (Coarse Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	3.7	14.5	5.9	4.0	15.7	6.4	6.8	26.5	10.8	8.3	32.5	13.2
SRL	3.4	13.3	5.4	3.7	14.5	5.9	6.5	25.3	10.3	8.0	31.3	12.8
DyFinIE	3.8	7.2	5.0	4.4	8.4	5.8	3.8	7.2	5.0	4.4	8.4	5.8

Table E.4: Performance on External Dataset (Coarse Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	8.6	33.7	13.7	11.1	43.4	17.6	7.3	28.9	11.8	9.5	37.3	15.2
SRL	8.0	31.3	12.8	9.9	38.6	15.8	6.5	25.3	10.3	8.0	31.3	12.8
DyFinIE	12.6	24.1	16.5	15.1	28.9	19.8	8.2	15.7	10.7	10.7	20.5	14.0

Table E.5: Performance on External Dataset (Coarse Relations) (5)

	Token		
Model	P	R	F1
OpenIE	22.9	73.5	34.9
SRL	22.1	70.4	33.7
DyFinIE	36.2	27.7	31.4

E.1.2 External (Granular Relations)

Table E.6: Performance on External Dataset (Granular Relations) (1)

	ExactRel			ExactRelND			FuzzyRel			FuzzyRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	2.1	3.6	2.6	2.1	3.6	2.6	2.1	3.6	2.6	2.1	3.6	2.6

Table E.7: Performance on External Dataset (Granular Relations) (2)

	RougeRel			RougeRelND			JaccardRel			JaccardRelND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DyFinIE	8.9	15.7	11.4	8.9	15.7	11.4	4.1	7.2	5.2	4.1	7.2	5.2

Table E.8: Performance on External Dataset (Granular Relations) (3)

	ExactSpan			ExactSpanND			FuzzySpan			FuzzySpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	3.7	14.5	5.9	4.0	15.7	6.4	6.8	26.5	10.8	8.3	32.5	13.2
SRL	3.4	13.3	5.4	3.7	14.5	5.9	6.5	25.3	10.3	8.0	31.3	12.8
DyFinIE	2.7	4.8	3.5	3.4	6.0	4.4	2.7	4.8	3.5	3.4	6.0	4.4

Table E.9: Performance on External Dataset (Granular Relations) (4)

	RougeSpan			RougeSpanND			JaccardSpan			JaccardSpanND		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OpenIE	8.6	33.7	13.7	11.1	43.4	17.6	7.3	28.9	11.8	9.5	37.3	15.2
SRL	8.0	31.3	12.8	9.9	38.6	15.8	6.5	25.3	10.3	8.0	31.3	12.8
DyFinIE	13.0	22.9	16.6	15.1	26.5	19.2	8.2	14.5	10.5	10.3	18.1	13.1

Table E.10: Performance on External Dataset (Granular Relations) (5)

	Token		
Model	P	R	F1
OpenIE	22.9	73.5	34.9
SRL	22.1	70.4	33.7
DyFinIE	37.4	25.8	30.6

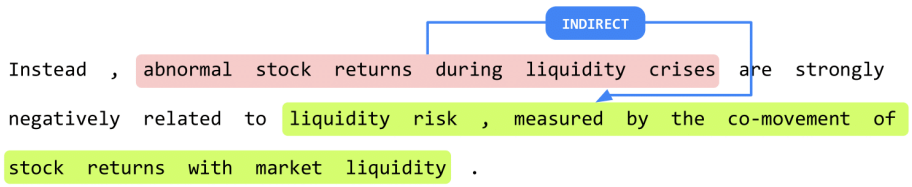
E.2 Analysis of Results

We find that in both the EXTERNAL Coarse and Granular datasets, our model DYFINIE achieves comparable or higher precision than the benchmarks OPENIE and SRL, but have a significantly lower recall and hence lower F_1 score. In this section, we provide some possible explanations for the poor recall of our model.

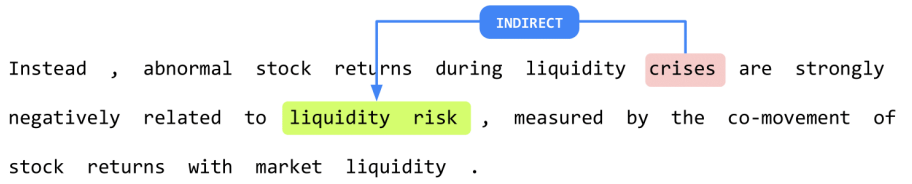
E.2.1 Shorter Entities

We find that our DYFINIE tends to tag shorter spans of text than the OPENIE and SRL models. Figures E.1 - E.3 show the differences between the gold standard (a) and predicted relations from our model (b). We observe that in all 3 cases, our model was able to identify a segment of the gold entity span, but failed to identify the entire entity span. This can likely be attributed to differences in the nature of annotations in FINMECHANIC that our model was trained on, and the EXTERNAL dataset.

For instance, the example in E.1 has the entity span '*liquidity risk, measured by the co-movement of stock returns with market liquidity*'. However, we would have annotated this entity span to be '*liquidity risk*', cutting off the entity span before the comma (,) in FINMECHANIC, which is identical to our DYFINIE model prediction. Likewise, the entity spans in Figure E.2 ('*degree of informational asymmetry and the ownership structure of the firm*') and Figure E.3 ('*abnormal returns with apparent sector-by-sector differences*') are all entity spans that contain either conjunctions or prepositions, that would have been identified as separate entities or omitted in FINMECHANIC.

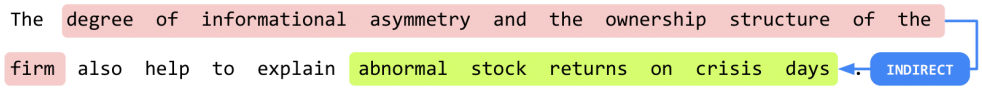


(a) Gold Standard

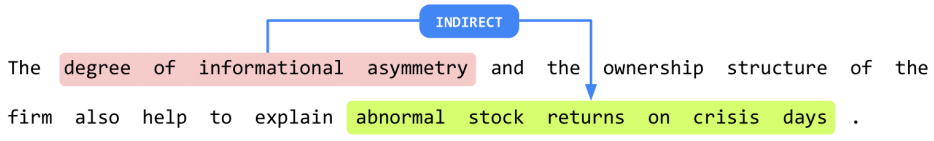


(b) Prediction

Figure E.1: Shorter Entity (1)

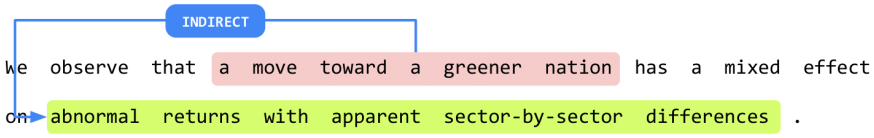


(a) Gold Standard

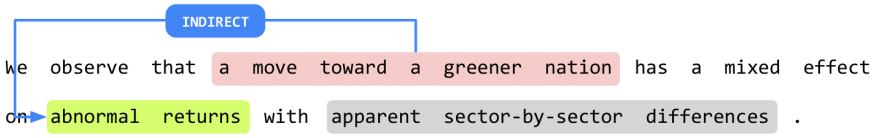


(b) Prediction

Figure E.2: Shorter Entity (2)



(a) Gold Standard

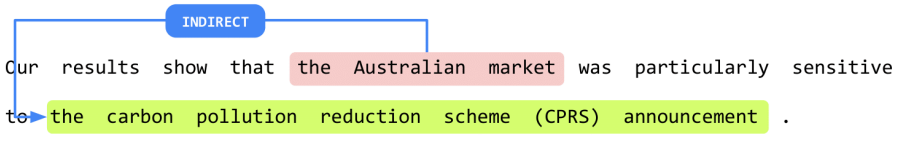


(b) Prediction

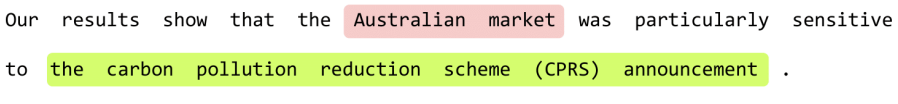
Figure E.3: Shorter Entity (3)

E.2.2 Missing Relations

We also found that our model would correctly identify the entity spans, but failed to identify a relation linking the two entity spans. An example is provided below (Figure E.4).



(a) Gold Standard



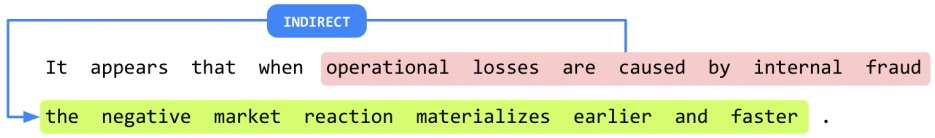
(b) Prediction

Figure E.4: Missing Relation

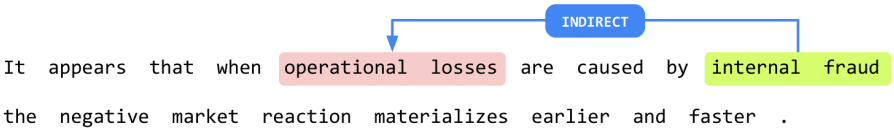
E.2.3 Alternative Relations

We also note that our DyFinIE model identified several plausible relations that were unlabelled in the External dataset due to different relation schemas. Figure E.5 demonstrates a case where DyFinIE accurately identifies a relation, but the relation is different from the one labelled in External. This is a result of a limitation in the labelling process of both FinMechanic and External where each token is only allowed to belong to one entity span, such that annotators will have to label the sentence with the relation identified in the gold standard or the prediction, but not both.

In Figure E.6, the relation between '*underwriting risk*' and '*investment risk*' was identified as `INDIRECT` by our model but not labelled in the External dataset. This is because External only labels relational triplets that pertain to the findings of the authors of the paper, and not findings cited from other sources, as is the case in this example (the relation was discovered by 'Scrand and Unal, 1998'). However, our FinMechanic dataset does not differentiate between cited and discovered relations, predisposing our model to wrongly identify this relation. Other types of relational triplets that would be labelled in FinMechanic but not in External include relations as part of the `NONE` and `UNCERTAIN` categories.

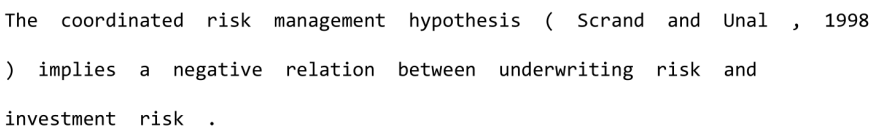


(a) Gold Standard

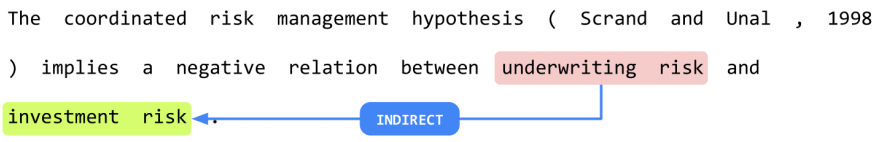


(b) Prediction

Figure E.5: Alternative Relation (1)



(a) Gold Standard



(b) Prediction

Figure E.6: Alternative Relation (2)

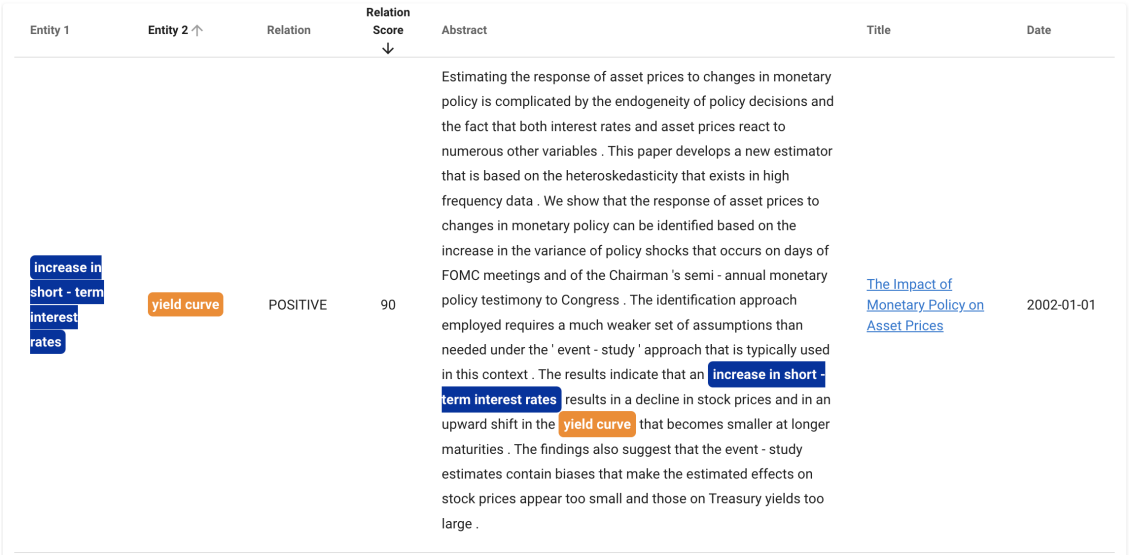
Appendix F

FinSearch Query Examples

In this section, we provide some examples of correct and incorrect search query results. The figures below are screenshots of the top abstract by relation score for each search query from the FinSearch service. For each figure, (E1: *Entity 1 String*; E2: *Entity 2 String*) in the figure caption denote the search queries that were used.

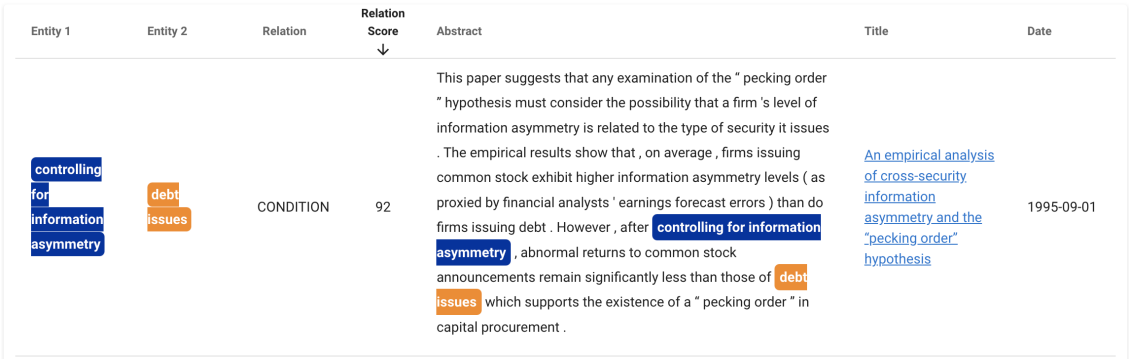
F.1 Correct

Figures F.1 and F.2 illustrate query examples where FinSearch successfully shortlists the most relevant relational triplets. We observe that both search queries returned an abstract with entities that were semantically similar to the search query, and that the relation in the abstract accurately reflects the relationship between the two entities, and is correctly identified.



Entity 1	Entity 2	Relation	Relation Score	Abstract	Title	Date
increase in short - term interest rates	yield curve	POSITIVE	90	Estimating the response of asset prices to changes in monetary policy is complicated by the endogeneity of policy decisions and the fact that both interest rates and asset prices react to numerous other variables . This paper develops a new estimator that is based on the heteroskedasticity that exists in high frequency data . We show that the response of asset prices to changes in monetary policy can be identified based on the increase in the variance of policy shocks that occurs on days of FOMC meetings and of the Chairman 's semi - annual monetary policy testimony to Congress . The identification approach employed requires a much weaker set of assumptions than needed under the ' event - study ' approach that is typically used in this context . The results indicate that an increase in short - term interest rates results in a decline in stock prices and in an upward shift in the yield curve that becomes smaller at longer maturities . The findings also suggest that the event - study estimates contain biases that make the estimated effects on stock prices appear too small and those on Treasury yields too large .	The Impact of Monetary Policy on Asset Prices	2002-01-01

Figure F.1: Correct (E1: *interest rates*; E2: *yield curve*)



Entity 1	Entity 2	Relation	Relation Score	Abstract	Title	Date
controlling for information asymmetry	debt issues	CONDITION	92	This paper suggests that any examination of the " pecking order " hypothesis must consider the possibility that a firm 's level of information asymmetry is related to the type of security it issues . The empirical results show that , on average , firms issuing common stock exhibit higher information asymmetry levels (as proxied by financial analysts ' earnings forecast errors) than do firms issuing debt . However , after controlling for information asymmetry , abnormal returns to common stock announcements remain significantly less than those of debt issues which supports the existence of a " pecking order " in capital procurement .	An empirical analysis of cross-security information asymmetry and the "pecking order" hypothesis	1995-09-01

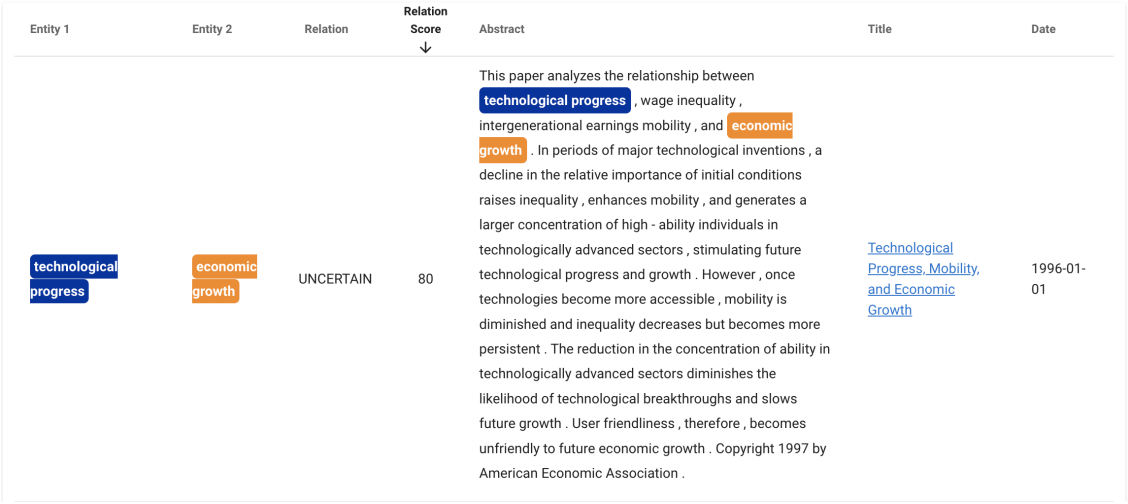
Figure F.2: Correct (E1: *information asymmetry*; E2: *debt*)

F.2 Incorrect Search Terms

We find that the FinSearch semantic search algorithm is mostly able to retrieve entities that are relevant to the search queries. In this section, we provide two types of search terms we identified that the FinSearch service performs badly on.

F.2.1 Niche Terms

We observe that the FinSearch service tends to be unable to retrieve relevant abstracts when the search terms are more niche or specific. Figure F.3 provides one such example, where we wish to search specifically for firms in the technology sector (*'technology firm'*). In this case, FinSearch is unable to find entities semantically related to '*technology firm*', and instead returns '*technological progress*' - which incorporates the concept of technology, but is not related to the firm.

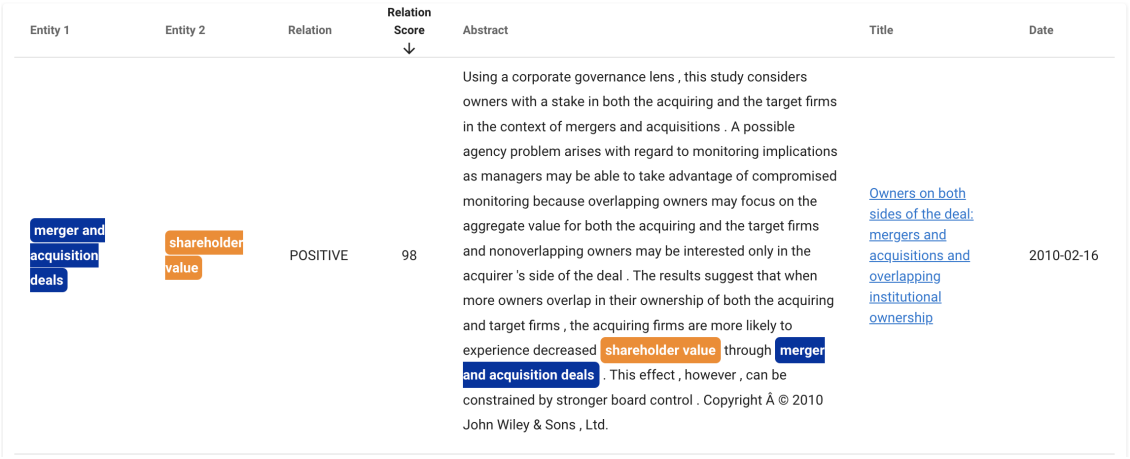


Entity 1	Entity 2	Relation	Relation Score ↓	Abstract	Title	Date
technological progress	economic growth	UNCERTAIN	80	This paper analyzes the relationship between technological progress , wage inequality , intergenerational earnings mobility , and economic growth . In periods of major technological inventions , a decline in the relative importance of initial conditions raises inequality , enhances mobility , and generates a larger concentration of high - ability individuals in technologically advanced sectors , stimulating future technological progress and growth . However , once technologies become more accessible , mobility is diminished and inequality decreases but becomes more persistent . The reduction in the concentration of ability in technologically advanced sectors diminishes the likelihood of technological breakthroughs and slows future growth . User friendliness , therefore , becomes unfriendly to future economic growth . Copyright 1997 by American Economic Association .	Technological Progress, Mobility, and Economic Growth	1996-01-01

Figure F.3: Niche Terms (E1: *technology firm*; E2: *economic growth*)

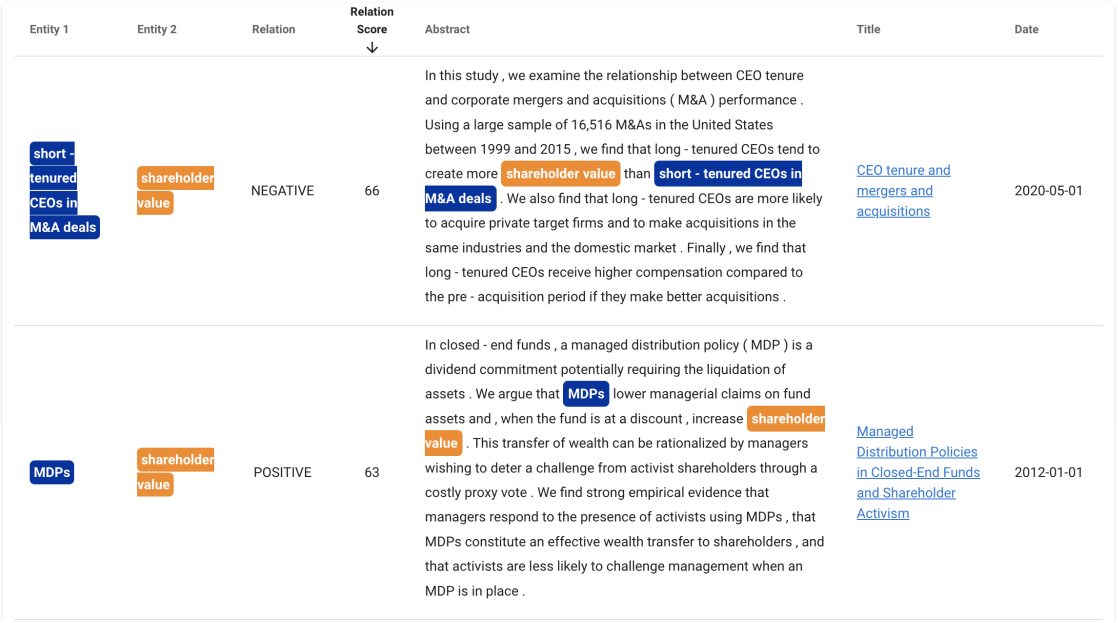
F.2.2 Abbreviated Terms

We also observe that FinSearch performs poorly to retrieve information when the search terms used are abbreviated. Figure F.4 provides an example of searching for '*merger and acquisition*', while Figure F.5 provides an example of searching for the abbreviation, '*M&A*' (merger and acquisition). We find that while there are abstracts with relational triplets that are semantically similar to the search queries, these abstracts are only flagged out when '*merger and acquisition*' is used, and FinSearch is unable to identify that '*M&A*' is equivalent to '*merger and acquisition*'. More data will likely be required to train the embedding model, FinMultiQA, to understand abbreviations.



Entity 1	Entity 2	Relation	Relation Score ↓	Abstract	Title	Date
merger and acquisition deals	shareholder value	POSITIVE	98	Using a corporate governance lens , this study considers owners with a stake in both the acquiring and the target firms in the context of mergers and acquisitions . A possible agency problem arises with regard to monitoring implications as managers may be able to take advantage of compromised monitoring because overlapping owners may focus on the aggregate value for both the acquiring and the target firms and nonoverlapping owners may be interested only in the acquirer 's side of the deal . The results suggest that when more owners overlap in their ownership of both the acquiring and target firms , the acquiring firms are more likely to experience decreased shareholder value through merger and acquisition deals . This effect , however , can be constrained by stronger board control . Copyright Â © 2010 John Wiley & Sons , Ltd.	Owners on both sides of the deal: mergers and acquisitions and overlapping institutional ownership	2010-02-16

Figure F.4: Abbreviated Terms (E1: *merger and acquisition*; E2: *shareholder value*)



Entity 1	Entity 2	Relation	Relation Score ↓	Abstract	Title	Date
short - tenured CEOs in M&A deals	shareholder value	NEGATIVE	66	In this study , we examine the relationship between CEO tenure and corporate mergers and acquisitions (M&A) performance . Using a large sample of 16,516 M&As in the United States between 1999 and 2015 , we find that long - tenured CEOs tend to create more shareholder value than short - tenured CEOs in M&A deals . We also find that long - tenured CEOs are more likely to acquire private target firms and to make acquisitions in the same industries and the domestic market . Finally , we find that long - tenured CEOs receive higher compensation compared to the pre - acquisition period if they make better acquisitions .	CEO tenure and mergers and acquisitions	2020-05-01
MDPs	shareholder value	POSITIVE	63	In closed - end funds , a managed distribution policy (MDP) is a dividend commitment potentially requiring the liquidation of assets . We argue that MDPs lower managerial claims on fund assets and , when the fund is at a discount , increase shareholder value . This transfer of wealth can be rationalized by managers wishing to deter a challenge from activist shareholders through a costly proxy vote . We find strong empirical evidence that managers respond to the presence of activists using MDPs , that MDPs constitute an effective wealth transfer to shareholders , and that activists are less likely to challenge management when an MDP is in place .	Managed Distribution Policies in Closed-End Funds and Shareholder Activism	2012-01-01

Figure F.5: Abbreviated Terms (E1: *M&A*; E2: *shareholder value*)

F.3 Incorrect Relation Labels

We also observe some instances where FinSearch is able to retrieve an abstract that contains relevant entities, but that the relations between the two entities are not correctly predicted. For instance, in Figure F.6, we observe that FinSearch is successfully able to retrieve entities (E1: *negative news*; E2: *bank stock returns*) from the search query (E1: *bad news*; E2: *bank stock returns*), but has incorrectly identified the relation between the two entities as `NEGATIVE`, likely due to Entity 1 containing the word '*negative*', when the actual relation should be `NEUTRAL`. Figure F.7 shows another example of an incorrect relation label.

Entity 1	Entity 2	Relation	Relation Score ↓	Abstract	Title	Date
negative news	bank stock returns	NEGATIVE	92	This paper investigates the effect of media talk on bank stock returns in response to corporate governance news . Using Loughran and McDonald 's (2011) dictionary , we create four categories of word lists that define the positive / negative tone and degree of certainty / uncertainty of news . We document three relevant findings . First , negative news significantly affects bank stock returns . Second , media coverage and the degree of certainty of the news are associated with more severe stock market losses . Third , bank capital and risk - adjusted performance mitigate the effect of negative news on stock prices . Overall , our study suggests that media talk on bank corporate governance events is an important determinant of abnormal stock returns .	Don't talk too bad! stock market reactions to bank corporate governance news	2020-12-01

Figure F.6: Incorrect Granular Label (E1: *bad news*; E2: *bank stock returns*)

Entity 1	Entity 2	Relation	Relation Score ↓	Abstract ↑	Title	Date
stock merger activity	market valuation	UNCERTAIN	88	Does valuation affect mergers ? Data suggest that periods of stock merger activity are correlated with high market valuations . The naive explanation that overvalued bidders wish to use stock is incomplete because targets should not be eager to accept stock . However , we show that potential market value deviations from fundamental values on both sides of the transaction can rationally lead to a correlation between stock merger activity and market valuation . Merger waves and waves of cash and stock purchases can be rationally driven by periods of over- and undervaluation of the stock market . Thus , valuation fundamentally impacts mergers . Copyright 2004 by The American Finance Association .	Market Valuation and Merger Waves	2002-01-01

Figure F.7: Incorrect Granular Label (E1: *stock merger activity*; E2: *market valuation*)