Prompting Large Language Models to create Knowledge Graphs from unstructured text

David Ika

This report is submitted as partial fulfilment
of the requirements for the Master of Data Science at
The University of Western Australia,
2023

Abstract

Knowledge graph construction is the process of extracting structured information, usually from unstructured text, and visualising the information in a graph format. Our study focuses on raw news articles as the dataset. Our goal is to extract relevant information about organisations, including their name and other related information implied in the article, such as risks they face, items or services provided, or other business relationships.

Our methodology involved prompting ChatGPT [2], a large language model (LLM), to extract structured information from the articles. The responses from the LLM were then converted into a querying language which was used to automatically populate a knowledge graph, allowing visual representation of the information.

The approach was compared to existing annotated datasets and studies, where they had similar intentions – to extract information from unstructured text. This study provides a modern approach on automating knowledge graph construction without relying on training datasets. We show that this methodology can be applied across domains with relative ease, compared to existing methods that may require extensive fine-tuning to achieve comparable results. We present a qualitative 5-pronged evaluation method to assess this, and our results show that prompting ChatGPT with well-engineered prompts can produce quality knowledge graphs from scratch - even from large, raw, messy and unseen datasets.[1]

[1]Code available at: `https://github.com/davidika/KG_construction`

Acknowledgements

We acknowledge Aylien News, from which the core dataset of raw news articles was extracted. We also acknowledge Neo4j, for their graph database software that was used to visualise the knowledge graphs.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Large langauge models such as OpenAI's GPT-3.5 [2] have brought remarkable advancements in the field of AI for Natural Language Processing. These models have demonstrated unprecedented capabilities in generating human-like text, understanding context, and even engaging in creative storytelling. This paper focuses on harnessing their potential to extract structured information from unstructured text.

As a preface to Knowledge Graphs (KGs), a graph database stores information via nodes and edges as opposed to, say, 2-d tables or n-d matrices: a node stores data on an entity, such as an object, notion or person, and an edge captures the relations between nodes. Graph databases provide faster synthesis of large and widely dispersed information, often utilised for ever-changing data that requires real-time analysis. Social media companies are an example of where these data requirements exist - a 'social network' graph would include *Person A* (node) *is friends with* (edge) *Person B* (node). Attributes of persons may also be tagged to these nodes, forming a property graph.

A knowledge graph (KG) is still a collection of information in a graph structure, but the focus is on information exchange as opposed to data storage. It then follows that the focus of knowledge graph construction is information extraction. KGs have information represented as triples, formed by 2 nodes and an edge, typically in the form of a subject-predicate-object statement. Whilst a property graph is an implementation model that focuses on enhancing storage and querying capabilities, a knowledge graph is the concept of presenting information in a way that improves synthesis. They are a means of representing structured knowledge and observing relations between entities. Traditionally, creating knowledge graphs has relied on structured data sources or manual curation, limiting their coverage and scalability. Or, when noisy datasets were investigated, heavy pre-processing and complex, domain-specific models have been required to extract useful information, increasing cost and reducing time-to-value.

The aim of this paper is to explore the capability of ChatGPT to automatically generate knowledge graphs from a new and messy dataset: raw, unprocessed news articles published in the past 6 months. As such, these articles will have not been seen by ChatGPT before, as its training data cut-off was late 2021. The flexibility of this methodology means users can set up an architecture to extract information from unseen text, without specifying specific entities or relations to look for - and without training complex, multi-staged models.

We have centred our research on assessing the ChatGPT's gpt-3.5-turbo model. This is a closed model which means effort was required to determine optimal prompts (prompt engineering) and query parameters, before assessing the model's strengths and weaknesses.

1.1 Use cases

Extracting information from freshly-released news articles can service many types of organisations in more ways that one. Articles that mention competitors' product launches, partnerships, or mergers and acquisitions, means organisations can have a live view of their competitive landscape. Or, information related to new regulations, legal issues, or geopolitical events can allow proactive risk and compliance management. Further examples occur in supply chain management, whereby fresh information related to companies in a supply chain can be crucial in identifying potential bottlenecks, managing inventory, and ensuring timely delivery of products.

1.2 Related Work

Entity extraction and relation extraction are fundamental tasks in Knowledge Graph (KG) construction, playing a crucial role in aggregating structured information from unstructured text. Entity extraction involves identifying and classifying named entities, such as people, organisations or locations. This process is essential for populating the KG with relevant entities, forming the building blocks of the graph. Relation extraction aims to discover and classify the relationships between those mentioned entities. By extracting relations, the KG can capture the connections and associations between entities, enabling a more comprehensive representation of knowledge. Together, entity extraction and relation extraction facilitate efficient data integration, knowledge discovery, and reasoning.

Early methods for KG construction were often pipeline models that consisted of several stages (entity extraction, relation extraction, and linking them together in the graph). Pipeline models, however, have certain limitations. Since each component is developed independently, errors from one stage can propagate to subsequent stages. This led to the development of joint models that perform entity and relation extraction simultaneously. For instance, Miwa and Sasaki [6] proposed a table-based composition model for joint entity and relation extraction. Bekoulis et al. [1] introduced an approach using deep neural networks to jointly learn named entity recognition and relation extraction. These end-to-end models have the advantage of dealing with inter-dependencies among different stages of KG construction, and tend to have a more streamlined training process.

The introduction of the Transformer architecture in 2017, by Vaswani et. al. [9] then marked a fundamental shift in natural language processing (NLP). Preceded by recurrent neural networks (RNNs) like LSTM and GRU, Transformers leveraged attention mechanisms, effectively sidestepping recurrence and yielding improvements in scalability and efficiency. Devlin et al. [3] then introduced their transformer-based model, BERT (Bidirectional Encoder Representations from Transformers), also known as a Pre-trained Language Model (PLM), which provided contextualised embeddings through conditioning on both left and right context.

These transformer-based models have given rise to the evolution of Large Language Models (LLMs), characterised by a substantial increase in parameter count compared to PLMs. Recently, there has been a surge in leveraging Large Language Models (LLMs) like ChatGPT for enhancing knowledge graph construction and information extraction. Trajanoska et al.[8] utilised ChatGPT in building pipelines for automated knowledge graph construction, significantly enhancing accuracy and delving into automatic ontology creation. Han et al. [4] undertook a comprehensive examination of ChatGPT's performance across multiple datasets and information extraction sub-tasks. They emphasized the need for soft-matching evaluation strategies, given ChatGPT's propensity for identifying longer spans. Zhu et al. [10] evaluated various LLMs, including ChatGPT and GPT-4, for KG construction and reasoning. These studies highlight the importance of re-thinking how KG construction is evaluated, particularly when using a model to jointly extract entities and relations (or, triples): whilst an F1-score may work for entity and relation extraction separately, issues become apparent when working with more complex datasets, especially when they are not annotated.

CHAPTER 2

Methodology

Figure 2.1 outlines our overall process, which first involves ‘prompt engineering’, an iterative process of investigating and determining a suitable prompt for our dataset and goals.

2.1 Prompt engineering

Prompt engineering refers to the deliberate crafting and refinement of prompts to elicit specific responses from chat-based AI models such as ChatGPT. It is an iterative process that involves defining the desired output, creating initial prompts, assessing the responses, and subsequently adjusting the prompts for optimal results. There are several types of prompt designs that may be used.

Chain-of-Thought (COT) prompts involve constructing a series of connected messages to maintain context throughout the conversation. COT prompts are well-suited for tasks requiring contextual awareness over multiple turns, where the subsequent step may change based on the answer of a prior step. This approach, however, consumes more tokens and can cause propagation of contextual errors through the conversation. We investigated this option, but noticed increased error rates compared to a single prompt. Since our methodology can be handled in a single step, chaining our prompts was also not necessary.

Masked prompts involve using placeholders, such as "[BLANK]", within a text and asking the model to fill in the missing information. They are best suited for tasks like filling in missing words, sentence completion, or solving equations. This approach is direct and simple to implement but is not ideal for complex or multi-step tasks. Since our project is interested in the complex and concise extraction of entities and their relations, using a masked prompt added unnecessary token counts.

In-Context Learning (ICL) prompts are structured to teach the model new information within the conversation or query. Similar to COT, ICL is effective with conversation-based models like gpt-3.5-turbo. It is beneficial for scenarios where the

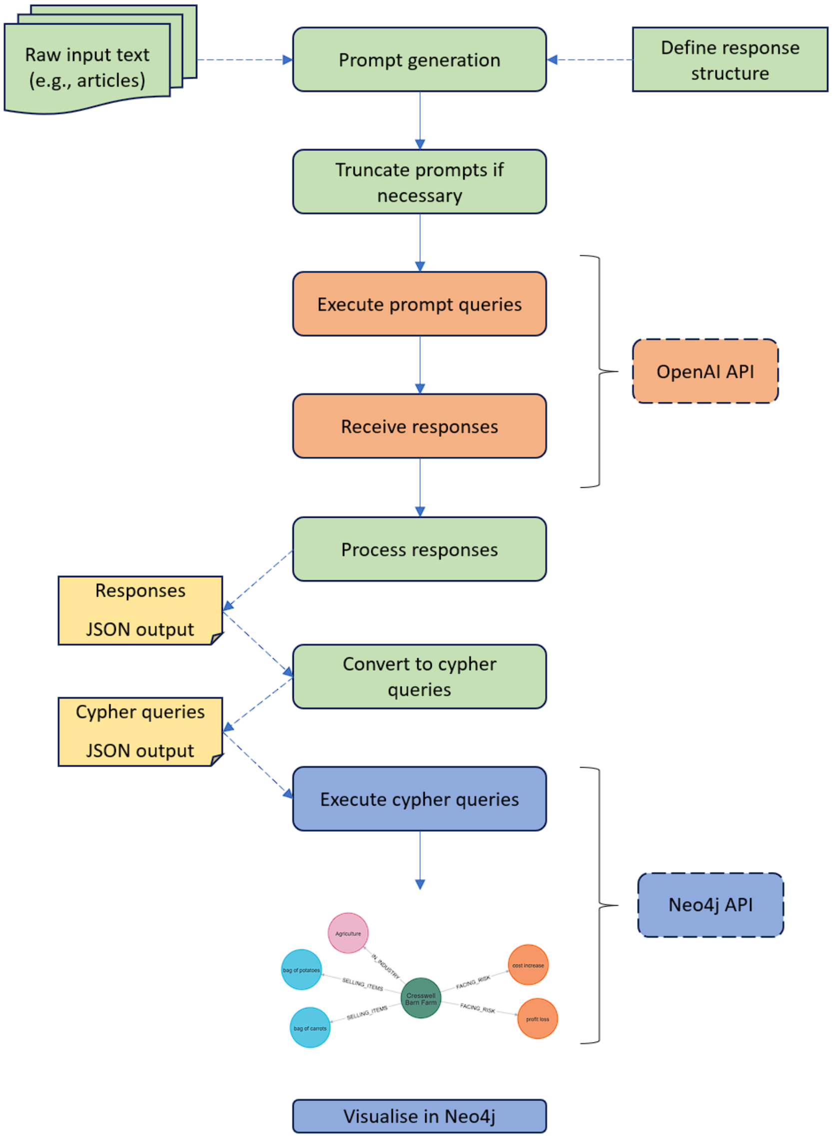

Figure 2.1: Architecture - Prompting LLMs to produce knowledge graphs.

model must apply newly acquired information, such as understanding a novel scientific concept and answering questions about it. However, it may require a precise structure to be effective and consumes tokens for teaching, which might limit querying capacity. ICL prompts worked well with our project, as we were feeding in articles which would have not been seen by ChatGPT during its training.

The above prompts can then be combined with zero-shot, one-shot or few-shot learning. In the context of prompting LLMs, this means that zero, one or multiple examples are provided within the prompt. We explored all 3 options, comparing responses received with the same ICL logic, but within each prompt, having zero, 1 and then 2 example articles with corresponding example responses. We noticed that providing one versus two examples produced minimal differences in responses, which could just be due to variations in ChatGPT's generative AI. When we looked at the zero-shot responses, however, we noticed repeated poor formatting errors and longer strings of text in responses. For example, in a zero-shot response, we saw *"service_provided: Multi-channel marketing strategy including automotive retail, wholesale and distribution, and franchise operations"*. The one-shot response on the same article produced *"service_provided: marketing; distribution; wholesale services"*, which was pursuant to our desired formatting (delimited by a semi-colon), and more concise. The downstream effect of this was that in the knowledge graph on the zero-shot response, that organisation had one *service* node called *"Multi-channel marketing strategy including automotive retail, wholesale and distribution, and franchise operations"* as opposed to the one-shot response producing three *service* nodes ('marketing', 'distribution' and 'wholesale services'); the latter being better preferred.

Some other key methods that were used when investigating prompts included providing explicit instruction with simple language on our main desires for the response; format specification, where the desired response format was indicated within the prompt; and context provision, which involved supplying pertinent background information to direct the model's comprehension and response.

There are also parameters that can be tweaked in the OpenAI API call, which can affect the responses received. The most important is the *temperature* search parameter, which influences the randomness of the model's output. A higher temperature leads to more diverse outputs, while a lower temperature makes the output more focused and deterministic. We investigated responses at 0, 0.2, 0.4, 0.6, 0.8 and 1, but noticed responses straying from the desired format when a temperature of 0 was not used. This makes sense as this is not a task requiring creativity - the aim was to extract information only from the provided article (included in the prompt).

For all datasets (including for benchmarking), our prompts incorporated the general structure outlined in Figure 2.2 - the bracketed items would be replaced with suitable text for the dataset and objective, whilst the curly-braced items represents dynamic input into the prompt; i.e. feeding in the input text for the required task. Figure 2.3 shows an example of a popualted prompt that was used for the Aylien News articles dataset (with an actual article, as opposed to the body placeholder). The verbatim response received to that prompt is then shown in Figure 2.4.

```
"Give me (desired information) as follows and nothing else. If there
is no (desired information), respond with "NA".
###
Desired response format:
###
(show the desired response)
###
Example of (input text):
###
(give an example of input text)
Corresponding example response:
###
(give an example response in the desired format)
###
The provided (input text) is:
###
{actual input text}
###"
```

Figure 2.2: A high-level example of a prompt structure that may be applied for different domains.

Our prompt structures pertained to the following.

1. Starting the prompt with the key request (the desired information), in plain language.
2. Specifying what to return if there is no desired information ("NA").
3. Specifying the desired response format.
4. Providing an example of input text (an example of an article).
5. Providing the corresponding example response.
6. Utilising ‘breaks’ (###) to logically split up the prompt.

Once a suitable prompt was determined, we created a prompt generation function to iterate over the dataset and produce prompts that differed only by the article body (the provided article). These prompts were of varying lengths and the key measurement was the tokenisation. Consideration was required in relation to the overall token limits per prompt, the budgeting constraints of running the model, and the request limitations attached to the API ID running the model (restricted at an organisation level). Of course, these may change over time at the discretion of OpenAI.

2.2 Token considerations

The OpenAI API uses tokenisation to determine the usage of their API. This impacts the model limitations of individual prompts, the overall rate limitations and the cost of using the model. GPT-3.5-turbo counts tokens using *cl100k_base* encoding, which produces a higher token count compared to the *NLTK* library, for example. We used the *tiktoken* library to iterate over the prompts, which produced the following token summary on the Aylien News dataset (5,000 records).

Item	**Token count**
Smallest prompt	436
Largest prompt	14,544
Total (all prompts)	5,607,221
Average per prompts	1,121

The large article bodies of text produced large prompts, some of which exceeded the total number of allowed tokens, which at the time, was 4,096 for the GPT-3.5-turbo model. Importantly, this limit includes the prompt (per Figure 2.3), the response received, and the message settings used when executing the prompt. We first determined the token count of the context setting applied to the system (being GPT-3.5-turbo). In our case, this was static for all prompts: 'You will be extracting information from the provided article in the specified format.' equated to 16 tokens. Then, we estimated the largest potential response that may have been received (similar to the response shown in Figure 2.4), being around 300 tokens. To be safe, when calling the API, we set *max_tokens=400*, which refers to the limit on response tokens received by the model.

Therefore, to ensure we remained within the total token limit for 3.5-turbo, we truncated our prompts to be no more than 3,500 tokens. Naturally, we truncated from the end backwards, which trimmed the latter part of some articles. Only 52 out of 5,000 prompts were truncated. After exploring the truncated prompts, some article bodies contained large amounts of random characters, or tables of contents, and some were just large articles. This truncation produced the following updated token count statistics (again, using the tiktoken library with cl100k_base encoding).

Item	**Token count**
Smallest prompt	436
Largest prompt	3,500
Total (all prompts)	5,499,101
Average per prompts	1,100

2.3 Pricing considerations

The above truncated queries' total token counts, the max_token parameter and the system context setting parameter were then combined to determine the maximum possible token usage, and thus the maximum price of prompting gpt-3.5-turbo for a dataset. The cost varies greatly depending on the model being used. At the time of writing, gpt-3.5-turbo was 1500% cheaper than running GPT-4 at 8k context, and 3000% cheaper than running GPT-4 at 32k context (32k context meaning up to 32,768 tokens could be processed on that model variation). Note that currently, the GPT-4 model usage via API is only available via special request, along with a wait-list.

The gpt-3.5-turbo cost-per-1,000 tokens was $0.002 USD, and with 5,000 records used in the Aylien News articles dataset, the following was calculated to determine maximum possible cost of iterating over the dataset. The same logic can be applied to any use-case, and the offical OpenAI API documentation should be checked beforehand for any updates.

Estimating maximum possible cost	
Average query tokens per prompt	1,102
System context tokens per prompt	16
Max response tokens per prompt	400
Total max token estimate per prompt	1,518
Multipled by 5,000 records	7,590,000
7,590,000 tokens at $0.002/1k tokens	**$15**

So to run the prompting over all 5,000 records, the max cost estimate was only $15. The actual cost ended up being around $13, as we would expect, because some articles were not detected to have an organisation, and so responded only with 'NA'. Other articles may have had some information against the requested slots, but 'NA' for other slots, hence also reducing the response token count.

2.4 Querying the API

Once the prompts were generated and truncated, they were fed into OpenAI's API. The set-up to query the API is relatively simple. Most of the iterative testing went into the prompt engineering, as there were only a few search parameters for the actual request body, as shown in Figure 2.5.

```
model = "gpt-3.5-turbo",
messages = [
"role": "system", "content": "You will be extracting information from the
provided article in the specified format.",
"role": "user", "content": query
], temperature = 0,
max_tokens = 400
```

Figure 2.5: Parameters for calling the OpenAI API

The model parameter refers to the ID of the model being used. The messages parameter allows high-level context setting on the system, and can be used for iterative chain-of-thought prompts by repeating the user and system role messages (not applied with our prompts). The content fed into the ‘user’ role are the generated prompts (only stated once with our prompt style). The temperature parameter is a float, where a higher value creates more randomness in responses. For information extraction and knowledge graph construction, however, having a temperature of 0 assists the model to be more deterministic. Finally, the max_tokens parameter restricts the response output from ChatGPT to be 400 tokens at most, as mentioned.

2.5 Post-processing of responses

At this point, we have the responses received from the model appended to our dataframe, making it easy to see the flow from the raw article (and other attributes), to the generated prompt, to the response received.

When testing a zero-shot method, and with temperature of 0.5, we noticed some responses not pertaining to the specified format; e.g. random trailing spaces and newlines, or characters like < or > appearing unintentionally, or the breaks (###) would leak into the response. Figure 2.6 shows an example of a poorly-formatted response. Considering this, we introduced some post-processing to remove unwanted characters (<, >, #) and then splitting on newlines, thus forming clean pseudo-dictionaries before exporting to a JSON file. If the populated slot was ‘NA’ then it was cleared, resulting in an empty value on that key-value pair. The clean example corresponding to Figure 2.6 is shown in Figure 2.7. After introducing one-shot prompts and changing the temperature to 0, we did not see any unintentional < or # characters, but we did notice some unintentional spacing in the response; hence, we left in this post-processing step. Not shown in the figure, two new slots were added to link information from the raw dataset to the responses: the *article_id* slot and the *source* slot, which were datapoints obtained when pulling data from the Aylien News API. This way, we could link the information collected to the article that it was detected in. There was no need to feed this into the OpenAI API, and only the *article_id* slot progressed to the Cypher query creation step.

```
###org: Dollar General Corp.
    country: U.S.
state: <Tennessee
  city: Goodlettsville
industry: Retail
risks: workplace safety; <public health >
items_sold:   NA
service_provided:     Retail
business_relations: Goodwill Easterseals Miami Valley ######
```

Figure 2.6: An example of raw response output with zero-shot and temperature = 0.5.

```
{
  "org": "Dollar General Corp.",
  "country": "U.S.",
  "state": "Tennessee",
  "city": "Goodlettsville",
  "industry": "Retail",
  "risks": "workplace safety; public health",
  "items_sold": "",
  "service_provided": "Retail",
  "business_relations": "Goodwill Easterseals Miami Valley"
},
```

Figure 2.7: Output from Figure 2.6 after post-processing.

2.6 Conversion to Cypher queries and visualising in Neo4j

The processed responses were then fed into a function that converted the cleaned JSON output into a series of Cypher queries. Figure 2.8 shows the creation of the Cresswell Barn Farm *organiastion* node, and the *item* node, named 'bag of potatoes'. Similar queries were created for every slot The implied relation was then *SELLING_ITEM*.

Similar Cypher queries were created for each slot of every response received from all the articles, forming around 18,000 queries. We see how the relations are implied because of how the prompt was engineered - the events and entities all link up to the main organisation discussed. The benefit of designing the prompt in this way meant that less tokens were used, and the output was cleaner, compared to requesting triples in the response. The relation definitions between the response slots were as follows *(subject-predicate-object)*.

1. org - *AT_COUNTRY* -> country
2. org - *AT_STATE* -> state
3. org - *AT_CITY* -> city
4. org - *IN_INDUSTRY* -> industry
5. org - *FACING_RISK* -> risk
6. org - *SELLING_ITEM* -> item 1
7. org - *SELLING_ITEM* -> item 2, etc.
   where new nodes and relations were created based on a list of items being received in the response (delimited by a semi-colon, as specified in the prompt).
8. org - *PROVIDING_SERVICE* -> service 1
9. org - *PROVIDING_SERVICE* -> service 2, etc.
   where new nodes and relations were created based on a list of services being received in the response (delimited by a semi-colon, as specified in the prompt).
10. org - *IN_BUSINESS_WITH* -> org
    where the second org referred to other ‘secondary’ organisations mentioned in the article, that were deemed to have a business relations with the main organisation discussed.

In the full graph, if other records (articles) had any of the same above values mentioned at the given slot, then they would be linked to each other, forming a holistic view of the entire dataset. The article ids containing the detected entities/events (nodes) were then attached as properties to all of those nodes (not shown in the graph produced). These queries were then run via Neo4j’s Python API to directly populate the graph database. Continuing from the output in Figure 2.4, the graph in Figure 2.9 was produced.

```
[
    "
    MERGE (o:Organisation {name: $org})
    ON CREATE SET o.article_ids = [$article_id]
    ON MATCH SET o.article_ids = coalesce(o.article_ids, [])
        + [$article_id]
    ",
    {
        "org": "Cresswell Barn Farm",
        "article_id": 5707800144
    }
],
[
    "
    MATCH (o:Organisation {name: $org})
    MERGE (i:Item {name: $item})
    ON CREATE SET i.article_ids = [$article_id]
    ON MATCH SET i.article_ids = coalesce(i.article_ids, [])
        + [$article_id]
    MERGE (o)-[:SELLING_ITEM]->(i)
    ",
    {
        "org": "Cresswell Barn Farm",
        "item": "bag of potatoes",
        "article_id": 5707800144
    }
],

[
... simillar code for other slots ..
]
```

Figure 2.8: Cypher code that creates the organisation node, the items node, and the implied relation: (org) - selling_item -> (item); i.e. (Cresswell Barn Farm) - selling_-item -> (bag of potatoes).

```
For the main organisation discussed in this article (if any), give me information as
follows and nothing else. If there is no organisation discussed, respond with "NA".
###
Desired response format:
###
org: <main organisation discussed>
country: <country of main organisation>
state: <state/province of main organisation>
city: <city/town of main organisation>
industry: <industry of the organisation>
risks: <semi-colon-separated list of risks faced by main organisation> Summarise each
risk in a phrase of up to 3 words.
items_sold: <semicolon-separated list of items sold by main organisation>
service_provided: <semi-colon-separated list of services provided by organisation>
business_relations: <semi-colon-separated list of organisations with a business relation
to the main organisation>
###
Example of an article:
###
"The collapse of Silicon Valley Bank, located in Santa Clara, California, U.S.,
continues to reverberate, hitting bank stocks, revealing hidden stresses with partner
bank, Credit Suisse, and setting off a political blame-game. Why the $212bn tech-lender
abruptly collapsed, triggering the most significant financial crisis since 2008, has no
single answer. Was it, as some argue, the result of Trump-era regulation rollbacks, risk
mismanagement at the bank, sharp interest rate rises after a decade of ultra-low
borrowing costs, or perhaps a combination of all three? Federal investigations have
begun and lawsuits have been filed and no doubt new issues at the bank will emerge. But
for now, here are the main reasons experts believed SVB failed."
###
Corresponding example response:
###
org: Silicon Valley Bank
country: U.S.
state: California
city: Santa Clara
industry: Financial
risks: political; financial; mismanagement; regulatory
items_sold: NA
service_provided: tech-lending;
business_relations: Credit Suisse
###
The provided article is:
###
"I know people are paying more for their bag of potatoes or bag of carrots, but it’s not
because we want more money," said Tessa.
"Our costs have gone up drastically and we are still trying to understand where we can
even make a profit."
"For fertiliser we were paying £290-odd and it shot up to £900 for that same bag," said
Tessa, whose family has run Cresswell Barn Farm for. . .  This story appeared on
bbc.co.uk , ." ###
```

Figure 2.3: Prompt structure used for the Aylien news dataset, populated with an article.

```
org: Cresswell Barn Farm
country: NA
state: NA
city: NA
industry: Agriculture
risks: cost increase; profit loss
items_sold: bag of potatoes; bag of carrots
service_provided: NA
business_relations: NA
```

Figure 2.4: Verbatim response received from ChatGPT (3.5-turbo API) for the Aylien prompt.

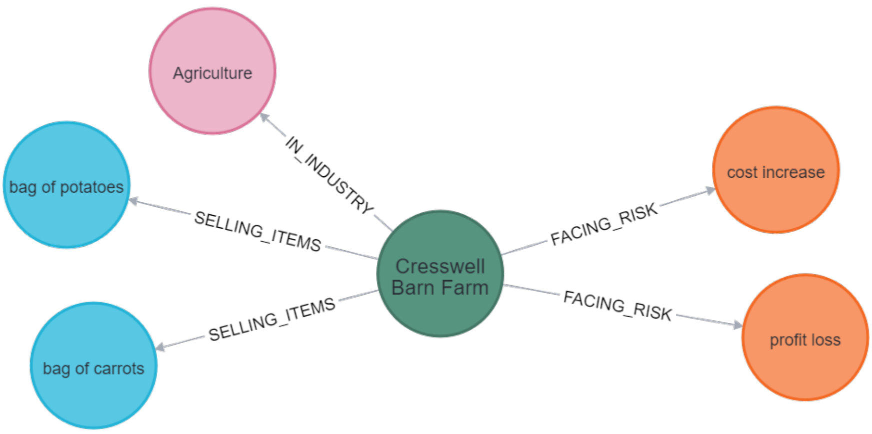

Figure 2.9: Graph produced from response received in Figure 2.4

CHAPTER 3

Data collection

3.1 Aylien News articles dataset

For the Aylien News articles dataset, we used their Python SDK API to retrieve bulk news articles. We modified some search parameters to suit our goal of extracting timely, business-related articles. Aylien uses two main categories for classifying their news content: IAB-QAG and IPTC. For our study, we utilised the IPTC classification system as they advise the higher suitability for market and business intelligence. Within the IPTC taxonomy, we selected three classifiers: *business*, *company information*, and *business enterprise*, represented by category IDs 04018000, 04016000, and 04008034, respectively.

To further refine our search, we used additional search parameters: Firstly, source ranking, represented by the parameter 'source_rankings_alex_rank_min.' We set a minimum source ranking of 10,000, ensuring that the news articles originated from online sources with an Alexa ranking of 10,000 or higher. This criterion aimed to include sources with significant online presence and influence. Secondly, we restricted the publication date range to the last six months, prioritising timely and up-to-date information. Lastly, we set the *sentiment_titly_polarity* as 'negative', with an aim to detect articles with a negative sentiment. The logic was that this would assist in detecting risks associated to companies.

We extracted 10,000 articles but focused on the first 5,000 for the main analysis. The dataset had 5 attributes: id, title, published_at (date of news article publication), source (source of the news article), and body (the raw unstructured text, which is the key input text being fed into the prompts).

3.2 Benchmarking datasets

For our news Aylien News article dataset, there were no manually verified annotations of the entities nor relations in the articles. To navigate this, we utilised two other datasets that contained manual annotations - meaning they were annotated by a human as to the entities, relations or triples that they deem should have been detected.

1. CoNLL-2003: The CoNLL-2003 dataset [7] is a widely used benchmark dataset for Named Entity Recognition (NER) tasks. It consists of sentences from news articles from the Reuters Corpus, annotated with named entity tags following the IOB (Inside, Outside, Beginning) tagging scheme. We processed the tokenised sentence and NER tags to produce the raw sentence and the list of the annotated entities. We used the test set, which contains 3453 records, and isolated to those containing at least one ORG tag, signifying the presence of an organisation. This reduced the number of records to 1229. We then worked with the raw input text (articles) and a list of the organisation(s) detected (target output). The smallest amount of words in a sentence was 1, and the largest was 63.

2. SciERC: The SciERC dataset [5] is a collection of scientific abstracts that were manually annotated with entities and relations (and triple annotations, when combined). The dataset was designed to facilitate information extraction and understanding in the scientific domain. For our comparison, the annotation and text files were imported, processed and combined to form 500 records of abstracts (input text) and the corresponding annotated triples in word-form (target output). The SciERC annotation guidelines specified 6 types of entities (e.g. task, method, material, . . .) and 8 types of relations (e.g. Used-for, feature-of, compare, . . .). For example, the sentence *"The TISPER system has been designed to enable many text applications."* would be annotated as *(TISPER system) – USED-FOR -> (text applications)*. The smallest amount of words in an abstract was 23, and the largest was 316.

CHAPTER 4

Results

4.1 Benchmarking

Benchmarking in a quantitative manner (e.g. F1 scores) is a difficult task in relation to knowledge graph construction, as there are many variants of entities and relations that may be extracted from a corpus of unstructured text. This is similarly noted by [8] and [4]. Since our research combined large, noisy news articles with complex, generative AI (via ChatGPT), this difficulty was further heightened. We used the CoNLL-2003 [7] dataset to benchmark entity detection, and the SciERC dataset [5] to benchmark relation detection. We designed prompts in a way that would attempt to achieve the human annotations against these datasets; firstly to assess entity detection by specifying entity type (organisations only), and secondly to assess relation detection by specifying relation type.

4.1.1 CoNLL-2003

We worked with 1,229 records, being the subset of sentences from the test set that contained at least one organisations, per the annotations. We ran the prompt in figure Figure 4.1 with the aim of detecting the annotated organisations in the same format. We firstly observed hard matches, defined as when the response from ChatGPT was the exact same as the annotations; i.e. same organisations with the same spelling detected in the same order. This amounted to 507 (exact true positives). Records that were annotated with an organisation but which ChatGPT did not detect anything, amounted to 421 (false negatives). Records for which ChatGPT detected an organisation (or several), that did not exactly match the annotations, amounted to 301 (false positives).

Looking into the apparent false negatives, the quality of the annotated CoNLL-2003 dataset came into question. Roughly 370 of the 421 records were what appeared to simply be sports scores of a team, with very little context provided. Some examples are shown in Table 4.1, with the annotated IOB tags and organisations (the response from ChatGPT for these examples was ‘no_orgs’, as intended by the prompt). It may

```
I need you to show me organisations in the provided sentence. Do not
show me any other entity types.
###
Desired response format: ###
org1, org2, org3
###
Desired response if no organisations found: ###
no_orgs
###
For example, if I gave you: ###
"James, living in Japan, missed his club ’s last two games after FIFA
slapped a worldwide ban on him for
appearing to sign contracts for both Wednesday and Udinese while he
was playing for Feyenoord .",
you would only give me:
FIFA, Wednesday, Udinese, Feyenoord
###
The provided sentence is: ###
Blinker was fined 75,000 Swiss francs ( $ 57,600 ) for failing to
inform the Engllsh club of his previous commitment to Udinese .
###
```

Figure 4.1: Prompt to benchmark against CoNLL-2003 (entity detection of organisations).

be reasoned that whilst the human annotator recognised that these were sports 'organisations', ChatGPT did not have enough context nor reason to make that assumption from seeing just a name next to a sequence of numbers, rightly so.

Table 4.1: CoNLL-2003 entity detection - false negative examples

Input text	**Annotated organisations**	**ChatGPT response**
Birmingham 0 Grimsby 0	Grimsby,Birmingham	no_orgs
Cambridge 22 13 3 6 33 27 42	Cambridge	no_orgs
Reading 22 7 5 10 25 33 26	Reading	no_orgs
Zafririm Holon 12 2 4 6 8 14 10	Zafririm Holon	no_orgs
VANCOUVER 14 11 1 84 83 29	VANCOUVER	no_orgs

4.1.2 SciERC

The prompt used on the ScieERC dataset gave a detailed example of each of the 8 types of relations that were desired to be detected, which were pursuant to the annotations guideline used by the annotators in the SciERC paper. An example of a populated prompt (containing an abstract) is shown in figure Figure 4.2. We see how the input text for SciERC (the abstracts), having multiple sentences, was more complex than the CoNLL-2003 dataset. This complexity made the benchmarking comparison even more difficult.

The number of annotated relations in the SciERC dataset was 6,394, whilst the number of relations detected by ChatGPT was 3,947. These numbers exclude 1 abstract (C88-2132) that contained a blank annotation file. Whilst the prompt specified the restricted list of relations to be detected, ChatGPT did sometimes stray from this, responding with other types outside of the list of 8. The distribution of relations detected is shown in Table 4.2.

```
 "Give me a comma-separated list of relations and coreferences detected in the provided
abstract, as long as they are in this list of restricted relation types:
###
USED-FOR, FEATURE-OF, HYPONYM-OF, PART-OF, EVALUATE-FOR, COMPARE, CONJUNCTION, COREF
###
Here are some examples of each type of relation that may be detected:
USED-FOR: "The TISPER system has been designed to enable many text applications, Our
method models user proficiency" would become "TISPER system USED-FOR text applications".
FEATURE-OF: "prior knowledge of the model" would become "prior knowledge FEATURE-OF
model".
HYPONYM-OF: "TUIT is a software library." would become "TUIT HYPONYM-OF software
library".
PART-OF: "We incorporate NLU module to the system." would become "NLU module PART-OF
system".
EVALUATE-FOR: "our work improves both the quality and the efficiency of entity
summarization" would become "efficiency EVALUATE-FOR entity summarization".
COMPARE: "Unlike the quantitative prior, the qualitative prior is often ignored" would
become "quantitative prior COMPARE qualitative prior".
CONJUNCTION: "obtained from human expert or knowledge base" would become "human expert
CONJUNCTION knowledge base".
COREF: "We introduce a machine reading system... The system..." would become "machine
reading system COREF system".
###
Desired response format: ###
rel1, rel2, rel3
###
Example response: ###
machine reading system COREF system", "prior knowledge FEATURE-OF model", "TISPER system
USED-FOR text applications"
###
The provided abstract is: ###
This paper introduces a system for categorizing unknown words . The system is based on a
multi-component architecture where each component is responsible for identifying one
class of unknown words . The focus of this paper is the components that identify names
and spelling errors . Each component uses a decision tree architecture to combine
multiple types of evidence about the unknown word . The system is evaluated using data
from live closed captions - a genre replete with a wide variety of unknown words .
###"
```

Figure 4.2: Prompt to benchmark against the SciERC dataset (relation detection).

Table 4.2: SciERC - Distribution of relations detected by ChatGPT

Relation type	**Annotated count**	**Detected count**
USED-FOR	2,437	136
COREF	1,675	902
CONJUNCTION	583	158
HYPONYM-OF	477	404
EVALUATE-FOR	454	15
PART-OF	270	574
FEATURE-OF	264	712
COMPARE	234	583
Other (non-annotated)	0	463
Total	6,394	3,947

We see that 3,482 of the relations detected were within the given list, whilst 463 relation types were given labels outside of that list, which was not intended per the prompt. Some examples of unintentional relation types being detected are shown in Table 4.3.

Table 4.3: SciERC - Some examples of relation types detected outside of the restricted list

Abstract ID	**Abstract (relevant text only)**	**New relation type detected**
C86-1081	"... In this paper we propose a logical formalism, which, among other things, is suitable for representing determiners without forcing a particular interpretation ..."	"logical formalism SUITABLE-FOR representing determiners"
ECCV_2012_37_abs	"... desired patches need to satisfy two requirements: 1) to be representative, they need to occur frequently enough in the visual world; 2) to be discriminative, they need to be different enough from the rest of the visual world."	"patches DISCRIMINATIVE-FOR rest of visual world ..."
CVPR_2015_300_abs	"...We then extend this approach to account for the availability of heterogeneous data modalities such as geo-tags and videos pertaining to different locations, and also study a relatively under-addressed problem of transferring knowledge available from certain locations ..."	"approach EXTENDED-TO account for heterogeneous data modalities", "approach STUDIED-PROBLEM of transferring knowledge"
INTERSPEECH_2007_-31_abs	"... We conclude that the HMMs are able to produce highly intelligible neutral German speech, with a stable quality, ..."	"HMMs ABLE-TO produce highly intelligible neutral German speech"

4.2 Aylien News articles

Due to the complex nature of the articles, the lack of annotations on this dataset, and the complex nature of triple extraction, it is difficult to automate a quantitative evaluation over the entire dataset. Whilst summary statistics can be pulled, as shown in Table 4.4, manual evaluation of the responses was required to assess the validity of the information extracted from the articles.

Table 4.4: Quantities of information extracted from the Aylien News business articles dataset

Slot	**Unique count**
org	2,097
country	88
state	107
city	373
industry	879
risks	1,219
items_sold	1,040
service_provided	2,129
business_relations	1,430
Total items extracted	**9,362**
Total implied relations created	**7,265**

Manual evaluation was performed on 100 randomly selected responses (regardless of whether the responses had populated slots). We summarised the following categories of errors for each slot (each item in Table 4.4), on each of the 100 articles.

1. False positives (FP): being when the model populated slots with information that could not be linked back to the raw article. This could be due to hallucination, where factually incorrect data was produced, or perhaps ChatGPT was pulling information from outside the article, when it should not (regardless of whether that information was correct).

2. False negatives (FN): being where the model failed to detect obvious information that could be filled in the slots. That is, failing to detect an organisation, or any obvious information related to that organisation, that could populate the other slots.

3. False relation assignment (FRA): For populated slots, there were some cases where they were simply being detected because it existed in the article, but did

not necessarily link back to the main organisation being discussed. For example, the *business_relations* slot may have been populated because other organisations were mentioned in addition to the main organisation, without anything that implies an actual 'business relationship' with the main organisation. Another type of FRA existed where there was nothing populated in the organisation slot, but other slots were populated. In this case, the populated slots were classed as FRA.

4. Lacking concision (LC): If the response seemed to be pulling long strings of text directly from the article, instead of summarising, this was classed as an LC error. For example, on the *risks* slot - some responses failed to summarise to an overarching risk (overarching being 'cybersecurity', 'financial', etc.).

5. Poor formatting (PF): being where the response did not follow the specified format. Responses needed to follow the key-value pair structure, without trailing newlines or spaces, and without any further slots being created. For articles where no organisation was detected, we accepted the following as being valid: *'NA'*, *'org: NA'*, or, all slots as *'NA'*. Any other response such as a verbose *'... no organisations were detected in the article ...'* were classed as a PF error.

Table A.1 in Appendix A shows this evaluation for the 100 articles, and a snapshot of this is in Table 4.5. A blank cell indicates no error on that slot, for that article.

Table 4.5: Manual evaluation of 100 randomly selected articles - Snapshot of first 10

Article_id	**Org**	**Country**	**State**	**City**	**Industry**	**Risks**	**Items_sold**	**Services_-provided**	**Business_-relations**
5698601531									
5686615086									
5685875803	FN					FRA			
5701836855						FRA			FRA
5704400782									
5708571295						LC	FP		
5705340118									
5687481764	FN								
5688529282									
5697658328									

Based on the above, we can then produce an error rate. For example, in Table 4.5, the snapshot of 10 articles showed errors on 5 slots out of 90 possible slots (9×10 articles), producing an error rate of 5.6%. We apply equal weighting to the types of errors, however this might be changed depending on specific goals. The distribution of errors detected and corresponding error rate on the 100 articles is shown in Table 4.6.

To provide further context into the errors, we will observe some articles and responses from Table 4.5. For longer articles, some bulk text was excluded.

The article body in Figure 4.3 discusses cost of living and inflation issues faced by

Table 4.6: Error rate distribution on the 100 articles.

Error type	**Count**
FP	5
FN	13
FRA	5
LC	7
PF	2
Total	**32**
Error rate *(32/900)*	**3.56%**

multiple businesses. However it can be argued that the article is centred on ‘Enterprise Nation’ (highlighted), with the latter part of the article quoting its CEO. The *risks* slot is populated in the response, presumably from the text appearing after ‘Enterprise Nation’(also highlighted). The formatting of this response is perfect; however the org slot being NA is a false negative (FN), and the *risks* slot being populated when the *org* slot is not, equates to a false relation assignment (FRA) error. Notably, in this case, even if the *org* slot was populated, those risks refer to ‘other businesses’ in the article, and would not be necessarily referring to ‘Enterprise Nation’, and so it would still be an FRA error.

We see in Figure 4.4 another article that had some FRA errors - highlighting the difficulty of some of these articles. Again, relevant text has been highlighted in relation to the population of slots, or lack thereof. ChatGPT correctly identified the Financial Commission as being the main organisation of the article, and the *industry* slot of ‘Financial’ is also acceptable; though perhaps ‘Finance’ would have been more ideal. However, the model incorrectly assigned ‘compliance’ as a risk against Financial Commission, when it is actually discussed as a risk against the other organisations. Furthermore, the *business_relations* slot was only partially correct, as it mentioned organisations that were discussed as being expelled from the Financial Commission’s network (‘BalansFX’, ‘Bold Prime Fiber Markets’ and ‘LordFX’), and ‘TresorFX’ should also not be there, as it was just discussed as being assigned to Financial Commission’s warning list. The correct population of this slot should have only been ‘ThreeTrader TMGM; Pepperstone’.

In contrast to the above, two articles that were correctly processed are shown in Figure 4.5 and Figure 4.6. We see in Figure 4.5 one of the more simpler and cleaner examples of article bodies. Here, there was no organisation mentioned, and all the slots were populated with ‘NA’. Figure 4.6 also had no errors in its response. The response correctly assigned ‘Campbell Soup Company’ as the main organisation being discussed, it determined that its industry was ‘Food and Beverage’, that it sold ‘canned soup products’, and, it correctly identified the business relationship with ‘Flagstone

Article body:

```
"The number of small businesses planning to increase prices to their
customers is set to rise dramatically this quarter, further fuelling
inflationary pressures.
. . .
A new quarterly analysis of small business confidence conducted by
small business support platform Enterprise Nation has found the
number of small firms that say they must put up prices has gone up by
11 per cent since the last survey in 2022. Emma Jones, CBE, founder
and CEO of small business support platform and business support
provider Enterprise Nation , said: “Small businesses have been
holding back since energy costs started to bite last year. Now the
competing pressures of inflation , energy and staff costs have proved
too much, and they have had to make the difficult decision to increase
prices. “Many small businesses told us they felt costs would have
stopped rising by now and had hung on as long as they could. “Despite
all of that, businesses are still supporting their communities. Today
more entrepreneurs are being driven by purpose and are giving back
through profit share or social enterprises, which is so brilliant to
see, especially in these straightened times.” The Small Business
Barometer found more than a third (39%) of businesses said sales had
fallen due to the cost-of-living crisis . Again, businesses in the
North East were hardest hit, according to the analysis with 56 per
cent saying sales had dipped, the highest in the UK."
. . .
```

Response received when prompting on this article:

```
org: NA
country: NA
state: NA
city: NA
industry: NA
risks: inflation; cost-of-living
items_sold: NA
service_provided: NA
business_relations: NA
```

Figure 4.3: Article ID 5685875803 and response received.

Article body:

"Zephyrnet Republished By Plato May 31, 2023 Views: The Financial Commission (FinCom), the independent external dispute resolution (EDR) body for the forex industry, has expelled retail FX broker BalansFX from its organization. The mediator announced the development on Monday, attributing the decision to the brokerage's numerous failures to comply with its rules and guidelines.
. . .
BalansFX is only one of several members FinCom pushed out of its organization in recent months. Forex and contracts for difference brokers recently expelled include firms such as Bold Prime Fiber Markets and LordFX . On the other hand, FinCom also welcomed new members such as ThreeTrader TMGM and Pepperstone in recent months. FinCom this week added TresorFX to its warning list, urging traders and investors to avoid the company's website. TresorFX is a broker and an online investment management advisory platform targeted at forex traders. The EDR organization said it took the decision based on complaints from traders. However, it did not disclose the specific complaints laid.
. . .
Source: https://www.financemagnates.com//forex/fincom-discontinues-balansfx-membership-over-compliance-failure/Zephyrnet"
. . .

Response received when prompting on this article:

org: Financial Commission
country: NA
state: NA
city: NA
industry: Financial
risks: compliance
items_sold: NA
service_provided: dispute resolution
business_relations: BalansFX; Bold Prime Fiber Markets; LordFX; ThreeTrader TMGM; Pepperstone; TresorFX

Figure 4.4: Article ID 5701836855 and response received.

Article body:

```
"A DOZEN people linked to 35 Singapore-registered companies were
charged in court on Thursday (June 1) after their company bank
accounts, operated by foreign agents, were used to receive and launder
criminal proceeds of around US$36 million from scams."
```

Response received when prompting on this article:

```
org: NA
country: NA
state: NA
city: NA
industry: NA
risks: NA
items_sold: NA
service_provided: NA
business_relations: NA
```

Figure 4.5: Article ID 5704400782 and response received.

Foods'. ChatGPT did well to ignore other noise in the article that could have been false relation assignments; i.e. the exclusion of 'snacks maker' from any slots is correct, as that refers to Flagstone Foods and not Campbell Soup Company.

Article body:

```
" Campbell Soup Company ( CPB ) , a canned soup products company ,
said on Tuesday that it has sold Emerald nuts business to
Flagstone Foods, a snacks maker . The financial terms of the deal
have not been disclosed. ""The transaction is expected to be dilutive
to earnings per share by approximately $0.01 in fiscal 2024,
reflecting the timing of a transition service agreement and cost
actions,"" the company said in a statement. CPB added that it does not
expect the transaction to have a material impact on its fiscal 2023
financial results. For the nine-month period to April 30, the Emerald
business has posted net sales of and $46 million. For the fiscal 2022,
Emerald business had generated net sales of $66 million. For comments
and feedback contact: editorial@rttnews.com" ."
```

Response received when prompting on this article:

```
org: Campbell Soup Company
country: NA
state: NA
city: NA
industry: Food and Beverage
risks: NA
items_sold: canned soup products
service_provided: NA
business_relations: Flagstone Foods
```

Figure 4.6: Article ID 5697658328 and response received.

CHAPTER 5

Conclusion

One core limitation of using the OpenAI API is the reliance on a closed-source model. The internal workings and mechanisms of the model are not fully transparent or accessible to users. This lack of transparency can be problematic, as it hinders the ability to understand and address any potential biases, limitations, or errors within the model. Users have limited visibility into how the model makes decisions or generates responses, making it challenging to identify and rectify issues. An example of bias might occur for information extraction attached to political or government entities, e.g., omitting any information that could be seen as negative sentiment against those entities; or, overly-enhancing negative sentiment extraction for foreign government entities (foreign from the perspective of OpenAI), although we did not see this in evaluation of our results. Over-reliance on a single model is also not ideal from a business-sense, as this can pose operational risk.

Future work in this field would be to re-run the experiment using the GPT-4 model, when it becomes available via API. From a performance-assessment perspective, investigation into a more robust, scalable and fair evaluation method would be beneficial, especially for working with unseen datasets that lack annotations; or, using ChatGPT as a method of annotating datasets would also be an option, as a prepatory step of a multi-staged mode.

Overall, ChatGPT's GPT-3.5-turbo model was able to successfully extract quality information from raw text. When suitable prompt and query settings were used, the error rate was low, and it did well to ignore the mass amounts of noise present in many of the Aylien news articles. In cases such as these, where there is a lack of training data, utilising ChatGPT via API is a scalable alternative to a complex, supervised model, dramatically reducing time-to-value.

APPENDIX A

Manual evaluation of 100 randomly selected articles

Table A.1: Manual evaluation of 100 randomly selected articles

Article_id	Org	Country	State	City	Industry	Risks	Items_sold	Services_-provided	Business_-relations
5698601531									
5686615086									
5685875803	FN					FRA			
5701836855						FRA			FRA
5704400782									
5708571295						LC	FP		
5705340118									
5687481764	FN								
5688529282									
5697658328									
5687545313									
5707134140									
5702808668									
5703577393							FN		
5682847686									
5679035541									
5701697748									FRA
5680535454									
5696643047									
5701589860									
5702781336									
5682703381									
5701455085									
5704943144									LC
5670193959									
5705711456									
5675330635	FN								
5706715292									
5670968952						LC			
5705844608									
5694313094									
5706465273									
5669598185	PF								
5669336062									
5696526422									
5687136020									
5678798695								FN	

Continued on next page

Table A.1 – continued from previous page

Article_id	Org	Country	State	City	Industry	Risks	Items_sold	Services_-provided	Business_-relations
5673710094									
5668485277									
5693092322	FN								
5678718731									
5675081958									
5709082471									
5708768270								FP	
5703206756									
5676473793					FP				
5677566513									
5671314552									
5681943281									
5685869998									
5671426585									
5671356654									
5687728954									
5684161548									
5701280598									
5682497645						FRA			
5701526423	PF								
5707800144									
5697098987									
5686748770									
5689057334								FN	
5673167839									
5674110009								LC	FN
5676712656									
5709128508									
5686079952	FN								
5680632405	FN								
5691797729									
5678115490									
5670578732	FP								
5694870466									
5708642539	FP							LC	
5707131506						FN			
5706884459									
5702232704									
5692672714									
5704390080								LC	
5684586342									
5686993425									
5677657373									
5680808112									
5681867043									
5674575410									
5697568324									
5682827411	FN								
5699104293						LC			
5703910296									
5704855880									
5669955939									
5676852294									
5674807659									
5705923430									
5707881417									
5700082780									

Continued on next page

Table A.1 – continued from previous page

Article_id	Org	Country	State	City	Industry	Risks	Items_sold	Services_-provided	Business_-relations
5705745686									
5670446371									
5683292140									
5679562667									
5706519276	FN								
5708692067									

APPENDIX B

Knowledge graph

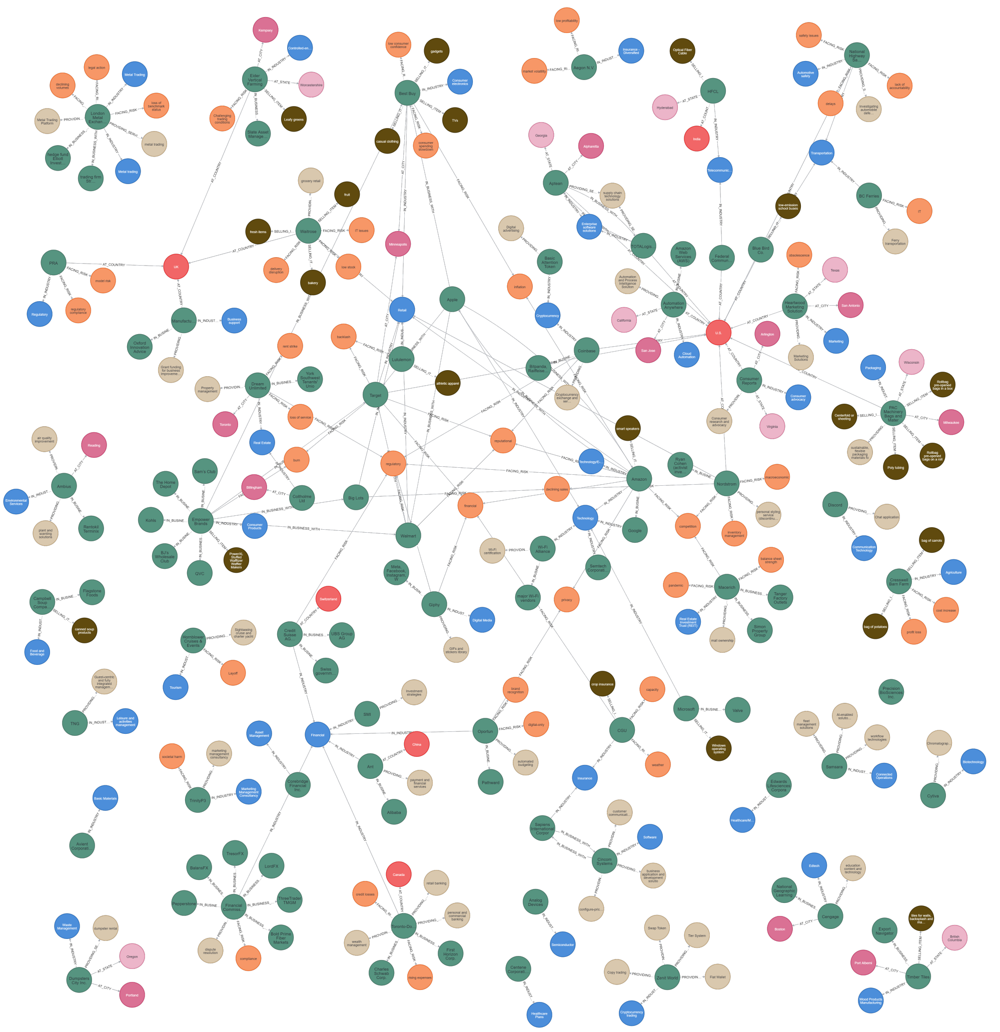

Figure B.1: Knowledge graph produced by the 100 randomly selected responses in Appendix A.

Bibliography

[1] BEKOULIS, G., DELEU, J., DEMEESTER, T., AND DEVELDER, C. Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (Brussels, Belgium, Oct.-Nov. 2018), Association for Computational Linguistics, pp. 2830–2836.

[2] BROWN, T., MANN, B., RYDER, N., SUBBIAH, M., KAPLAN, J. D., DHARIWAL, P., NEELAKANTAN, A., SHYAM, P., SASTRY, G., ASKELL, A., AGARWAL, S., HERBERT-VOSS, A., KRUEGER, G., HENIGHAN, T., CHILD, R., RAMESH, A., ZIEGLER, D., WU, J., WINTER, C., HESSE, C., CHEN, M., SIGLER, E., LITWIN, M., GRAY, S., CHESS, B., CLARK, J., BERNER, C., MCCANDLISH, S., RADFORD, A., SUTSKEVER, I., AND AMODEI, D. Language models are few-shot learners. In *Advances in Neural Information Processing Systems* (2020), H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., pp. 1877–1901.

[3] DEVLIN, J., CHANG, M.-W., LEE, K., AND TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[4] HAN, R., PENG, T., YANG, C., WANG, B., LIU, L., AND WAN, X. Is information extraction solved by chatgpt? an analysis of performance, evaluation criteria, robustness and errors, 2023.

[5] LUAN, Y., HE, L., OSTENDORF, M., AND HAJISHIRZI, H. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (Brussels, Belgium, Oct.-Nov. 2018), Association for Computational Linguistics, pp. 3219–3232.

[6] MIWA, M., AND SASAKI, Y. Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar, Oct. 2014), Association for Computational Linguistics, pp. 1858–1869.

[7] SANG, E. F. T. K., AND MEULDER, F. D. Introduction to the CoNLL-2003 shared task: Language-Independent Named Entity Recognition. In *Proceedings of CoNLL-2003 and the 7th Conference on Natural Language Learning* (2003), W. Daelemans and M. Osborne, Eds., pp. 142–147.

[8] Trajanoska, M., Stojanov, R., and Trajanov, D. Enhancing knowledge graph construction using large language models, 2023.

[9] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems* (2017), I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc.

[10] Zhu, Y., Wang, X., Chen, J., Qiao, S., Ou, Y., Yao, Y., Deng, S., Chen, H., and Zhang, N. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities, 2023.