Knowledge graph extraction in a practical context

Roos Bakker [a,b,*], Daan Di Scala [a] and Maaike de Boer [a]
[a] *Department Data Science, TNO, The Netherlands*
E-mails: roos.bakker@tno.nl, daan.discala@tno.nl, maaike.deboer@tno.nl
[b] *Leiden University Centre for Linguistics, Leiden University, The Netherlands*

Abstract. Knowledge graphs provide structure and semantic context to unstructured data. Creating them is labour intensive: it requires a close collaboration of graph developers and domain experts. Information extraction techniques can automate this process. This paper presents a comparative analysis of relation extraction methods for knowledge graph extraction. The methods are assessed within a real-life scenario, aiming for graph quality comparable to manually developed graphs. Previous methodologies often relied on automatically extracted datasets and a limited range of relation types, consequently constraining graph expressivity. Moreover, these datasets typically feature short or simplified sentences, failing to capture the complexity of real-world texts like news messages or research papers. The results show that GPT models outperform other relation extraction methods in quantitative metrics. However, qualitative analysis reveals that alternative approaches like REBEL and KnowGL excel in leveraging external world knowledge to enrich the graph beyond textual content alone. This highlights the importance of considering methods that not only extract relations directly from text but also incorporate supplementary knowledge sources to enhance the richness and depth of resulting knowledge graphs.

Keywords: Knowledge Graph Extraction, Relation Extraction, Ontology Learning, Knowledge Graphs, Large Language Models

1. Introduction

In the digital age, the abundance of textual data presents a chance and a challenge. Knowledge graphs can aid in structuring and utilising this knowledge, leading to an increasing demand for methodologies to extract knowledge from textual sources. Knowledge graph extraction has become a popular technique to answer this demand because of their ability to organise and connect information. By representing knowledge and data in a structured graph format, relationships between entities become explicit, allowing for reasoning, interoperability, efficient retrieval, and other downstream applications.

Creating knowledge graphs from scratch is a labour intensive task. Domain specialised models require time from domain experts and graph developers to ensure its quality. Information extraction techniques can support this process by extracting entities and relations from text. Existing fields of study that are often used in the process of knowledge graph extraction include, but are not limited to, relation extraction, Named Entity Recognition (NER), keyword extraction, and link prediction [30]. Additionally, end-to-end approaches have been suggested, which classify relations in texts utilising language models [29]. The multi-step approach suffers from its dependence on individual parts. As Jaradeh et al. [30] discuss, their text triple extractor is a weakness in their architecture due to low quality of its output. End-to-end relation extraction models are often task specific, with set relation types and do not have the

[*]Corresponding author. E-mail: roos.bakker@tno.nl.

flexibility to work on a range of texts [29]. Additionally, many datasets for this task are created distantly supervised, which impacts the quality.

In this paper[1], different knowledge graph extraction techniques are compared with the goal to create a knowledge graph that represents the text from which it is extracted in a way a graph developer would manually create it for an applied use case. For this purpose, a news message and a simple baseline text are annotated, identifying subjects, objects, and their relation. A collection of relation extraction methods is run on this dataset, and evaluate their performance. The contributions of this paper are as follows: 1) A small annotated dataset containing different versions of a news message, and 2) A comprehensive comparison of relation extraction techniques within the context of this dataset. With the dataset, containing simple and complex versions of a news message, this work demonstrates how complexity affects performance. Additionally to using standard evaluation metrics (precision, recall, F1), a clustering coefficient is included to show the density of the knowledge graph. To illustrate the strengths and weaknesses of the methods that are not reflected in the metrics, a qualitative analysis is performed. Additional results and materials are included in the open source repository from this paper.[2]

The next section sketches an overview of knowledge graph extraction, including information extraction techniques. In Section 3, the data and annotation method is introduced and the various methods are discussed. The results are presented in Section 4 and further analysed in Section 5. Finally, this work is concluded with a summary and suggestions for future research directions in Section 6.

2. Related Work

Several fields are involved or related to the extraction of knowledge graphs, such as information extraction and ontology learning. In this section, the definition of knowledge graphs is discussed, followed by a short overview of knowledge graph extraction and related fields. Afterwards, the field of information extraction and techniques related to knowledge graph extraction are introduced.

2.1. Knowledge Graphs

Knowledge graphs and similar concepts such as ontologies have been around since the originating of the field of philosophy. The computer science interpretation of knowledge modelling, with terms such as knowledge graphs, appears in literature as early as the 1970s [21], with early research on extraction from text and other sources starting a little later [7]. Knowledge graphs as a term and technique have become more popular in both research and industry since Google announced their implementation [52].

The knowledge graph as it is known today is a powerful representation framework that organises information in the form of interconnected nodes and edges [33]. In this graph-based structure, nodes typically represent entities (such as people, places, or concepts), while edges denote relationships between these entities. The combination of the two nodes and its relation is called a triple. This structure allows for the creation of a rich network that captures the context and connections within a dataset. Knowledge graphs enable a more nuanced and context-aware understanding of information, facilitating effective data exploration and retrieval [61].

Another feature of knowledge graphs is their ability to integrate diverse sources of information, aggregating data from various domains and presenting it in a unified format. This makes it possible to discover new relationships in complex datasets, such as textual databases where no structure is present. Knowledge graphs have applications in a range of fields, including semantic search, recommendation systems, and artificial intelligence, where the structured representation of knowledge is an essential counterpart to the large unstructured machine learning models [31]. They are also increasingly more used in applications where understanding and transparency of the data is essential, such as the safety or the medical domain [18].

[1]This manuscript is an extension of the paper “From Text to Knowledge Graph: Comparing Relation Extraction Methods in a Practical Context” by Roos M. Bakker and Daan L. Di Scala, in proceedings of the workshop GeNeSy24 co-located with ESWC 2024 [9]

[2]https://gitlab.com/genesysubmission/text2kg

2.2. Knowledge Graph Extraction

Knowledge Graph Extraction is a task that aims to extract knowledge graphs from different sources, using a variety of techniques. It is closely related to ontology learning and information extraction. Ontology learning, an established field, focuses on automatically or semi-automatically learning ontologies, including complex elements like rules and hierarchies. Information Extraction techniques are often used for extracting knowledge graphs. Common techniques are discussed Section 2.3. Throughout this work, we adopt the term *knowledge graph extraction* [9]. The objective in this work is to extract structured information from unstructured texts, with the goal to create a knowledge graph. This term underscores our focus on this specific task, distinct from related tasks like relation extraction or specialised areas such as ontology learning.

2.2.1. Ontology Learning

Ontologies have been used as knowledge bases, reasoning tools, and schematic tools in the context of information science. Consensus is that they are stricter than knowledge graphs; in ontology terms they could be considered a subclass of knowledge graph [1]. An ontology is a formal specification of concepts in the world [53]. In other words, an ontology can represent knowledge about part of our world. For instance, an ontology about pizza can include different types of pizza, pizza toppings, types of dough, etc. [49].

Creating ontologies is an extensive task, which involves the time of a domain expert and a modeller and requires maintenance. Therefore, automatically creating ontologies or part of them has been a fruitful field of research. Multiple overviews have been published, for instance earlier overviews based on rule-based approaches from Buitelaar et al. [15] to recent surveys including machine learning approaches from Khadir et al. [34]. Buitelaar et al. [15] give an extensive overview of the field as it was until 2005. They divide the task into complexity levels: starting with the learning of just terms and ending at the top with hierarchies, relations, and finally rules. State-of-the-art techniques were rule-based and focused on lexico-syntactical patterns. Such patterns could not consistently be identified, and the recall was low [15]. First attempts at relation extraction were done using statistical analysis combined with linguistic patterns such as dependencies [23]. For all approaches on all levels, manual work was necessary to produce a coherent ontology.

More recently, Wong et al. [60] describe the field from the start of the millennium until 2012, also including the work described above. They point out the relevant distinction in types of ontologies; ranging from lightweight ontologies without axioms [24], to heavyweight ontologies that have extensive axioms and relations [22]. Successful techniques and tools were at best able to extract a lightweight ontology, as Wong et al. [60] state. These lightweight ontologies are in many aspects similar to knowledge graphs.

As statistical approaches gained in popularity due to the increase of computing power and successful machine learning applications, the field of ontology learning changed. Asim et al. [5] include more recent statistical approaches such as co-occurrences, hierarchical clustering, and shortly touch upon transforming ontological concepts and relations into vectors. They make the distinction between linguistic and statistical approaches, and recognise the difference between term extraction and relation extraction, with the second being the more complex task [5].

Recent advancements in ontology learning include the use of natural language processing techniques for more efficient and scalable ontology learning [27, 34]. The term ontology learning is used less, and the focus seems to have shifted to knowledge graphs. Khadir et al. [34] make a distinction between two approaches: linguistic, and statistical and machine learning. The first approach utilises linguistic features to collect concepts and taxonomic relations, for instance, the syntactic patterns can be used to extract noun phrases (NP), and taxonomic relations can be extracted using Hearst patterns [28]. Machine learning approaches utilises clustering or classification algorithms for adding new concepts to an existing graph. As of today, linguistic and machine learning approaches often go together. For instance, Tian et al. [54] train graph convolutional networks using dependency trees.

Another example of a combination of techniques is the multi-tool Plumber [30], that tries to optimise the combination of different approaches. For different texts, different approaches are suggested. The distinction is made between co-reference solution, triple extraction, entity linking, and relation linking [30]. Its dynamic pipelines outperform static pipelines, but scores for the knowledge graph completion task remain low. This has to do with performance of the individual components, where results can still be improved. The work of Jaradeh et al. [30] focuses mainly on Knowledge Graph Completion, and state that current approaches are not viable for complete knowledge graph construction from unstructured text, because tasks such as keyword extraction are not enough by themselves to produce a knowledge graph. However, within the field of information extraction, the relation extraction task has been approached as an end-to-end task, which might produce a knowledge graph with its combined relations.

An essential aspect of ontology learning besides the creation of ontologies is the evaluation of them [60]. As soon as ontologies increase in complexity, for instance by adding a taxonomical structure, evaluation techniques come short. McDaniel and Storey [38] sketch an overview of this problem and introduce a metric suite that is suitable for different types of lightweight ontologies or knowledge graphs. Recent work by Bakker and de Boer [8] extend previous metrics and test them in an experimental setting. They show that such metrics can indicate the quality of changes, but evaluation of a first version of a knowledge graph or an ontology still requires manual steps.

2.3. *Information Extraction Techniques*

The field of Information Extraction is closely associated with the extraction of knowledge graphs. A specific area within information extraction dedicated to knowledge graphs is Open Information Extraction (OpenIE) [41]. This technique involves generating triples, comprising a subject, relation, and object, from textual data. Multiple techniques are often combined, in a similar approach to Jaradeh et al. [30] as described above. Information Extraction as a field underwent a surge with the implementation of word embeddings [39]. These first models lead to more complex architectures such as long short-term memory models (LSTMs) and the current state-of-the-art, transformers [55], of which BERT [19] is widely used for a variety of tasks.

Currently, decoder-only models such as GPT [14] have gained prominence. They are known as generative Large Language Models (LLMs), due to the vast amounts of texts and parameters with which they are trained. They excel at absorbing and producing factual information in natural text [14], with their main purpose being language generation. Research on emergent properties of LLMs, including semantic entailment and reasoning, is in its early stages [59]. OpenAI's neuron activity analysis tooling [13] provides insights into how GPT models respond to specific inputs, contributing to the understanding of knowledge representation within LLMs.

Despite their powerful memories, LLMs, trained in just a few epochs, exhibit limitations in factual consistency due to hallucination issues [32]. Memory augmentation techniques, such as infusing external data sources into LLM architectures [48], aim to address this misalignment with factual data.

Research efforts focus on extracting stored information directly from LLMs. Prompting schemes, like *in-context learning*[40], reveal the potential to reconstruct verbatim training data[16]. Other approaches involve manual prompting for completing subject-predicate-object triples [3], leading to a redefined role for LLMs in the inference or completion of knowledge graphs rather than their direct production[46].

Knowledge graph extraction can be done end-to-end, as algorithms such as co-occurrences aim for and experiments with LLMs illustrate [11, 17]. However, to achieve a high quality graph, without much noise and that is future-proof, often multiple approaches are combined. Both node and relation extraction techniques can be useful, depending on the goal of the graph and the pipeline around it.

Node Extraction For Knowledge Graph Extraction, multiple techniques can be combined in steps to produce a graph. A first step is the extraction of nodes. Named Entity Recognition (NER), where entities such as persons and locations are identified in texts, can serve as an initial step in knowledge graph extraction, although its application has been limited thus far [2]. NER is often combined with Named Entity Linking (NEL), where entities are linked in an existing graph or database. Similarly to NER, keyword extraction can provide subjects and objects to the graph. Common techniques include frequency analysis where words appearing frequently within the document are considered as potential keywords [43]. A more advanced version of this, TF-IDF, was introduced in 1973, where the uncommonness of the word is also taken into consideration by including the inverse document frequency [51]. Later approaches involve linguistic aspects of the text such as part-of-speech tagging and syntactic analysis [37, 42]. This can be used to identify important terms based on their grammatical roles and relationships within the sentence [12]. Recent approaches are often based on language models which are trained on this task, such as the BERT-based method KeyBERT [19, 25]. This approach demonstrates high performances in producing accurate keywords for respective texts.

Relation Extraction Beyond extracting terms or concepts from text, determining the relationships between these concepts is crucial for creating graphs. Over the past decade, machine learning approaches have been employed to view relation extraction as a binary classification problem [6]. Similarly to early approaches of keyword extraction, early approaches of relation extraction were based on frequencies. For knowledge graph extraction, such approaches have been demonstrated by [17], with the side note that manual filtering and other steps are necessary to produce a high quality graph. Recently, deep learning methods, specifically neural relation extraction, are more often applied, but these methods usually lead to noisy patterns [31]. Nowadays, statistical approaches are prevalent, with architectures such as graph LSTMs [47] and transformers.

Relation extraction can be done on sentences, or on paragraphs or documents. On a document level, this is still a challenging task, but the results might be more representative to the domain and comparable to manual development because relations outside sentence boundaries are also included [47]. Wang et al. [57] propose a method of training a language model such that it becomes better at recognising the structure in text, which also benefits relation extraction. They introduce structure pretraining, where their model is pretrained to recognise triples using relation datasets. They tested their model on a variety of tasks, including relation extraction. It achieved state-of-the-art performance on many of them, indicating that this approach can be successful in improving the structural understanding of language models. However, with state-of-the-art F1 scores going up till 0.67 for models such as [36], extracting relations on a document level is still a challenging task.

Extracting relations on a sentence level has been approached traditionally as a multi-step problem, similarly to ontology learning approaches described in Section 2.2.1. Recently, more end-to-end solutions have been proposed [29, 58]. Huguet and Navigli [29] introduce the REBEL model and dataset, where an encoder-decoder transformer is trained on their dataset for relation extraction. The distantly supervised dataset is created by expanding on T-REx [20]. 220 types of relations are extracted from Wikipedia abstracts, which are combined with extracted relations from Wikipedia texts using a Natural Language Inference model. The REBEL model has shown state-of-the-art performance on relation classification benchmarks. With the KnowGL model, Rosiello et al. [50] extend the REBEL dataset by adding entity labels and types, with the goal to generate a set of facts relevant for generating a knowledge graph.

Early work on generative large language models for relation extraction shows potential. Bakker et al. [11] perform a qualitative comparison of methods among which GPT-3.5 Turbo and propose a multi-step approach. However, this works lacks a quantitative analysis. Wan et al. [56] use a prompt engineering approach in a multi-step architecture for relation extraction and demonstrate that such an approach shows promise for relation classification. Allen et al. [4] outline the diverse roles of LLMs in knowledge engineering, and propose research questions, among which are questions regarding how LLMs can support the engineering of knowledge systems. This work implements and tests one possible answer to this question: the automatic extraction of a knowledge graph from text.

3. Method

In this paper, different relation extraction techniques are compared with the goal to create a knowledge graph that represents the text in a way a knowledge graph developer would manually create it. For this purpose, a small annotated dataset was created, which is described in Section 3.1.A collection of relation extraction methods is run on the dataset, they are described in Section 3.2. Finally, their performance is evaluated with the metrics described in Section 3.3.

3.1. Dataset

In this work, the focus lies on creating a knowledge graph from a real-life use case and compare the performance of different relation extraction methods. An example of a domain where knowledge graph extraction is valuable is the Safety domain. Keeping track of multiple sources of information is critical for effective decision-making and response to emerging threats. News messages are an important source of information. By extracting structured knowledge from news messaged, safety organisations can swiftly identify and analyse relevant entities and relations. Therefore, a news message that is relevant for this domain was annotated: the first news message about the Nord

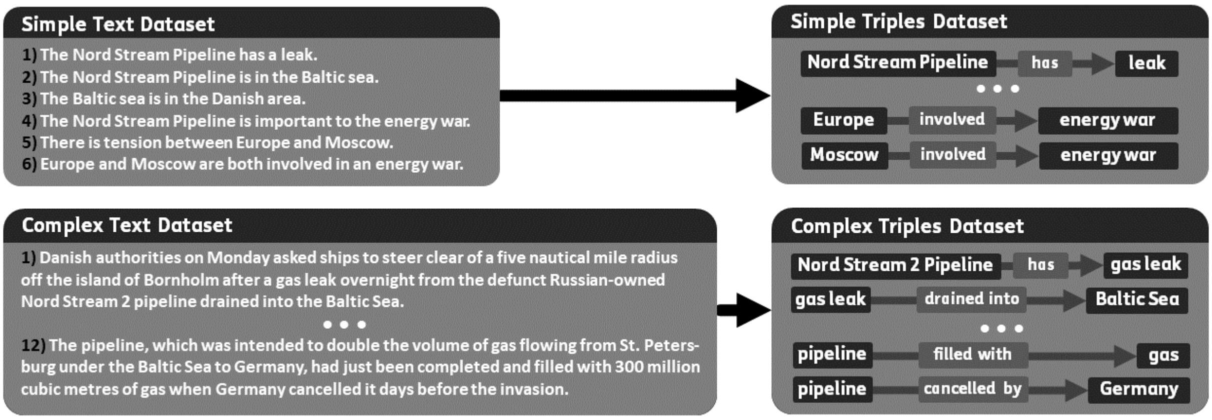

Fig. 1. Overview of the dataset generation process. From the news message, a simple and complex text dataset is built, from which a simple and complex triples dataset is extracted for evaluation purposes

Stream Pipeline incident by Reuters[3]. On September 26, 2022, a series of underwater explosions and subsequent gas leaks struck the two Nord Stream natural gas pipelines. Both pipelines were inactive due to the Russian invasion of Ukraine. The leaks occurred in international waters prompting separate investigations by Denmark, Germany, and Sweden. As of January 2024, investigations continue, with the explosions characterised as sabotage and the perpetrators yet to be officially identified. The news message does not include details on the cause or involved parties.

This news message consists of two parts: 1) a report of the incident and involved parties, and 2) a collection of statements on the incident that the writer has gathered from involved parties. Knowledge graphs are better suited to factual information, therefore the first part of the news message was included as the baseline complex text. Additionally to this complex news message text, a simplified version of the text was created, containing only the key points of the news message in simple sentences. This simplified text was used to test the hypothesis that relation extraction should be easier on a simpler text than on a complex text such as the news message.

Both texts were annotated with full triples, consisting of two nodes and the relation between them, as shown in Figure 1. The following conditions were given for annotation: 1) Each triple must be fully stated in the text, 2) each triple must indicate factual information, and 3) each triple must adhere to the (subject, relation, object) format. The first condition ensures that no common-sense world knowledge is included, or other information that is known to the annotator but cannot be found in the text. The second condition excludes opinions or non-factual statements such as `Denmark's energy agency gave a statement`. The final condition excludes statements about statements, so no additional time or location information, which means the triple (operator, disclosed, pressures drop) is extracted instead of (operator, (disclosed, pressure drop, on Monday)). The constructed baseline knowledge graphs can be seen in Figures 2 and 3. The full news message, the complex and simple texts, the complex and simple triples and the graph visualisations can all be found in the repository of this paper[2].

3.2. Methods

In this section, the different methods are described that are tested on the news message and its simplified version described in Section 3.1. All methods are run on on sentence level and on document level. These methods each have their strengths and weaknesses, which can be vary depending on the applied use case. An overview of the methods and their properties is given in Table 1. Each method is compared based on the type of information it can extract (Extracts), whether it can generate additional type information (Type info), whether weights are included in its output (Weights), whether its output follows a formal standard (Standard output), whether external knowledge

[3]https://www.reuters.com/business/energy/pressure-defunct-nord-stream-2-pipeline-plunged-overnight-operator-2022-09-26

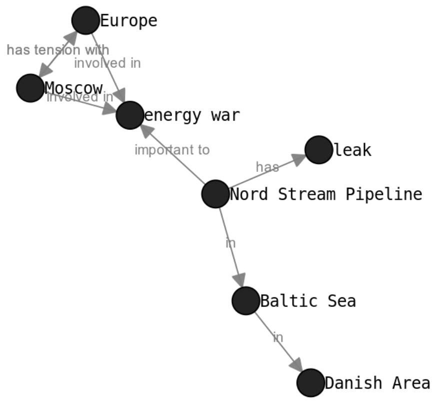

Fig. 2. Simple text knowledge graph

Fig. 3. Complex text knowledge graph

outside of document input is provided in the output or just information strictly from the given text (External info), and on which model it is based on (Base model). Each method and its parameters is described in the next sections.

KeyBERT In creating a knowledge graph, selecting informative and representative nodes is crucial. Therefore, KeyBERT [25], which is based on BERT [19], is implemented.Parameters are set to a top-n of 15 and a diversity of 0.5, along with Maximal Marginal Relevance (MMR) to ensure a balance between similarity and diversity in the extracted keywords. MMR minimises redundancy and maximises diversity by selecting keywords similar to the document iteratively, enhancing the quality of keywords for graph construction.

Co-occurrences (COOC) While useful in a pipeline because of its high quality concepts, keyword extraction does not provide us with triples; there are no relations between the nodes. For extracting triples, several approaches are implemented. Firstly, a co-occurrence algorithm as a baseline, similarly to previous work by de Boer et al. [17] and Bakker et al. [10]. The algorithm works by analysing the frequency with which words appear together within a sentence or document. It is suitable as a baseline method because of its simplicity and interpretability. The maximum distance for words that occur together is set to 5. Further, a threshold of 0.9 is used to define a minimum amount of times a word pair must occur.

REBEL and KnowGL Another implementation is the REBEL model [29]. The REBEL model is specifically designed for extracting relation triplets from raw text and is built upon the BART Transformer model [35]. The REBEL model is implemented using the number of beams and the number of return sequences settings of 5 on sentence level, and both on 30 on document level. KnowGL [50] is an extension to REBEL with entity types. The KnowGL parameters are set to the same values as REBEL.

Table 1

A comparison between relation extraction methods

	Extracts	**Type info**	**Weights**	**Standard output**	**External info**	**Base model**
COOC	Linked entities	✗	✓	✓	text only	-
KeyBERT	Entities	✗	✓	✓	text only	BERT
REBEL	Triples	∼	✗	✓	external	BART
KnowGL	Triples	✓	✗	✓	external	BART
GPT-3.5 Turbo	Triples	∼	✗	✗	external	GPT
GPT-4	Triples	∼	✗	✗	external	GPT
GPT-4o	Triples	∼	✗	✗	external	GPT

GPT methods Alternatively, three generative LLMs were implemented for relation extraction. Previous approaches using these models for relation extraction showed promise [11, 56], but no extensive analysis has been performed yet. Three models from the GPT family were tested: GPT-3.5 Turbo [14], GPT-4 [44] and GPT-4o [45]. These models were selected for their superior performance on most tasks and their availability. Both GPT-3.5 Turbo and GPT-4 were queried with the following prompt:

```
Take the following document: [text], Extract all relations in this text to a
graph. The graph format must be in JSON, with nodes and edges. Make sure to
include the three parts of the 'subject', 'object' and 'relation' triple
for each relation you find. Think carefully before you answer.
```

For GPT-4o, the prompt is extended with:

```
Only include the JSON in your answer, do not describe or explain the process.
```

This is added to ensure the results are kept to just the JSON-formatted Knowledge Graph without any additional information included in the answer. For all GPT models, the temperature was kept at the low setting of 0.3 to prevent hallucinations of relations that can not be found in the text.

3.3. Evaluation

To evaluate the performance of all methods on extracting nodes and triples from the dataset,precision, recall, and the F1 score is used; commonly used metrics for information retrieval tasks. Correctness of nodes and triples are compared manually to the dataset, and nodes/triples that are semantically identical to the baseline are counted as correct (e.g., `the pipeline` instead of `pipeline` is approved, yet `Danish` instead of `Danish Area` is disapproved). The triple is only evaluated as correct when both the nodes and the relation is correct.

Additionally, the average Clustering Coefficient CC_{avg} [26] is measured of each of the graphs. The CC_{avg} is based on the density of the neighbourhood surrounding each node of the graph, and is calculated by counting for each node the amount of triples it is either a subject or object of, divided by the total amount of possible triples ($n \times n - 1$, with n=total amount of nodes). With this metric, the completeness of the graph is measured. As discussed by Guéret et al. [26], the aim of knowledge graphs should not be to be fully complete ($CC_{avg} = 1$), as most links would be meaningless, but a high clustering coefficient indicates a well-cohesive graph. Finally, a qualitative analysis is performed, highlighting distinctive qualities of various methods from the perspective of a graph developer.

4. Results

In this section, the results are discussed from the methods described in Section 3.2 on the dataset as discussed in Section 3.1, based on the metrics described in Section 3.3. First, the performance on node level is shown, where only the extracted entities are evaluated (Section 4.1). Second, the results evaluated on the triple level are presented, considering the full extracted triples (Section 4.2). Third, the extracted graphs are compared on their density scores (Section 4.3). Finally, a qualitative evaluation on observations made during evaluation (Section 4.4) is described.

4.1. Nodes

The node extraction performance of the methods is shown in Tables 2 and 3. As shown in Table 2, performance on the simple text is overall high, with GPT-3.5 Turbo on sentence level scoring an F1 of 1, matching perfectly to the ground truth. While REBEL on document level scores lowest, on sentence level REBEL scores a perfect recall and high precision. As seen in Table 3, on the complex text, the highest precision is scored by KeyBERT on document level. Highest recall and F1-measure are by GPT-4o on sentence level. Furthermore, on the complex text, the F1 and recall scores are lower on a document level in all cases except for co-occurrences. No such pattern can be observed for precision.

Table 2
Scores on Node level for Simple text

Method	Level	Precision	Recall	F1
KeyBERT	doc	0.875	**1**	0.933
KeyBERT	sen	0.875	**1**	0.933
COOC	doc	0.438	**1**	0.609
COOC	sen	0.438	**1**	0.609
REBEL	doc	0.429	0.429	0.429
REBEL	sen	0.875	**1**	0.933
KnowGL	doc	0.667	0.571	0.615
KnowGL	sen	0.636	**1**	0.778
GPT-3.5 Turbo	doc	0.875	**1**	0.933
GPT-3.5 Turbo	sen	**1**	**1**	**1**
GPT-4	doc	0.875	**1**	0.933
GPT-4	sen	0.875	**1**	0.933
GPT-4o	doc	0.875	**1**	0.933
GPT-4o	sen	0.778	**1**	0.875

Table 3
Scores on Node level for Complex text

Method	Level	Precision	Recall	F1
KeyBERT	doc	**0.8**	0.261	0.393
KeyBERT	sen	0.537	0.783	0.637
COOC	doc	0.232	0.696	0.348
COOC	sen	0.232	0.696	0.348
REBEL	doc	0.667	0.174	0.276
REBEL	sen	0.628	0.587	0.607
KnowGL	doc	0.571	0.087	0.151
KnowGL	sen	0.559	0.413	0.475
GPT-3.5 Turbo	doc	0.604	0.63	0.617
GPT-3.5 Turbo	sen	0.633	0.826	0.717
GPT-4	doc	0.657	0.5	0.568
GPT-4	sen	0.529	0.804	0.638
GPT-4o	doc	0.778	0.761	0.769
GPT-4o	sen	0.676	**1**	**0.807**

4.2. Triples

The triple extraction performance of the methods is shown in Tables 4 and 5. Note that due to KeyBERT only providing entities and COOC only providing linked entities (see Table 1), both methods score 0 on all metrics and are left out. Table 4 shows that the GPT methods score a perfect precision on the simple text, with GPT-3.5 Turbo scoring a perfect recall and F1-measure as well. KnowGL scores the lowest F1-measure on the simple text. On the complex text, as seen in Table 5, GPT-4o scores the highest F1-measure and recall on sentence level, and on document level it scores the highest precision. KnowGL on document level notably extracted no correct triples. Recall on document level was again lower for all methods than on sentence level.

4.3. Graph Density

The density of the graphs extracted by the methods is shown in Figures 4 and 5, based on the average clustering coefficient CC_{avg} metric (see Section 3.3). KeyBERT does not produce relations and therefore the density cannot be calculated. The density of the baseline knowledge graphs is shown as a line, which is CC_{avg}=0.054 for Simple text and CC_{avg}=0.00085 for Complex text. As seen in Figure 4, on the Simple text, KnowGL on document level and REBEL both score highest and COOC on sentences has the lowest density score. The density of GPT-3.5 Turbo's

Table 4
Results for Simple text on Triple level

Method	Level	Precision	Recall	F1
REBEL	doc	0.429	0.375	0.4
REBEL	sen	0.227	0.625	0.333
KnowGL	doc	0.222	0.25	0.235
KnowGL	sen	0.167	0.375	0.231
GPT-3.5 Turbo	doc	0.875	0.875	0.875
GPT-3.5 Turbo	sen	**1**	**1**	**1**
GPT-4	doc	**1**	0.875	0.933
GPT-4	sen	0.75	0.75	0.75
GPT-4o	doc	**1**	0.875	0.933
GPT-4o	sen	**1**	0.875	0.933

Table 5
Results for Complex text on Triple level

Method	Level	Precision	Recall	F1
REBEL	doc	0.143	0.073	0.097
REBEL	sen	0.079	0.122	0.096
KnowGL	doc	0	0	-
KnowGL	sen	0.122	0.122	0.122
GPT-3.5 Turbo	doc	0.333	0.195	0.246
GPT-3.5 Turbo	sen	0.434	0.561	0.489
GPT-4	doc	0.625	0.244	0.351
GPT-4	sen	0.364	0.585	0.449
GPT-4o	doc	**0.719**	0.561	0.630
GPT-4o	sen	0.610	**0.878**	**0.720**

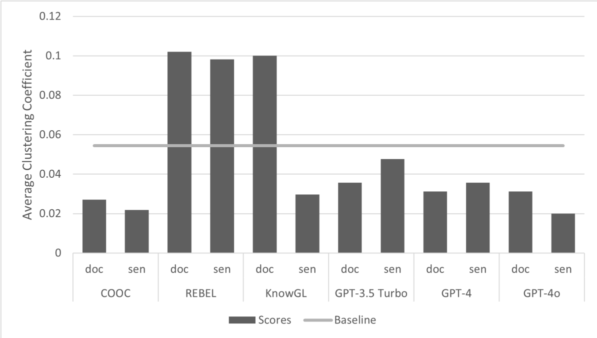

Fig. 4. CC_{avg} density scores for Simple text

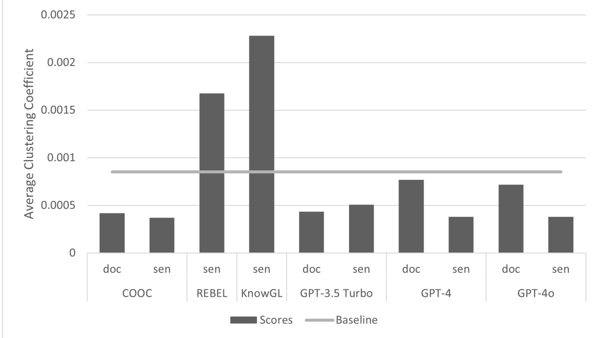

Fig. 5. CC_{avg} density scores for Complex text

graph is closest to the baseline's density. Results of the complex text can be seen in Figure 5. Because on document level REBEL and KnowGL produce high outlier density scores (CC_{avg}=0.061 for KnowGL and CC_{avg}=0.027 for REBEL), both are left out of the Figure. On complex text, REBEL and KnowGL score higher than the baseline, while GPT-4's density on document level is closest to the baseline density.

4.4. Qualitative Analysis

Co-occurrences (COOC) COOC generates odd results from a knowledge graph developer's standpoint. It only outputs unigrams, so many full entities are missed. For example, `Danish`, `maritime` and `authorities` are extracted instead of `Danish maritime authorities`. Furthermore, many verbs are incorrectly found as entities (e.g., `declined`, `ran`). Also, due to only providing linked relations, no triples are incorrect as no semantic value is given when a connection between subject and object does exist. For example, `st, cooc, petersburg` while St. Petersburg exists as a city, or `pressure, cooc, pipeline` while pipeline and pressure are connected because it is the pipeline's pressure.

KeyBERT Because KeyBERT provides no relations altogether, sometimes entities are picked up what ideally would be considered the relation of a triple. For example, the simple text baseline includes `Europe, tension, Moscow` as (subject, relation, object)-triple, yet KeyBERT provides `tension` as entity.

REBEL and KnowGL Where the results of the COOC method entail too little semantic value, REBEL and KnowGL often provide too much additional information. For example, KnowGL includes world knowledge, providing triples such as `Germany, shares border with, Denmark` or `St. Petersburg, located in or next to body of water, Baltic Sea`. While including this world knowledge can be useful to create well-connected knowledge graphs, it is rather a knowledge graph extension step with additional information outside of the text. This is less effective when generated triples are slightly incorrect or non-informative, such as `Danish, located in or next to body of water, Danish area`.

However, opposed to other methods, KnowGL does include symmetric relations (e.g., providing both `Nord Stream AG, owner of, website` and `website, owned by, Nord Stream AG`), which from the perspective of knowledge graph development is an informative feature, although a proper schema might be able to infer them. Due to the REBEL classifying the text to relations it is trained on, often the relation part of the triples are semantically slightly incorrect, such as `Nord Stream 2, product or material produced, gas` instead of `Nord Stream 2 pipeline, is filled with, gas`.

GPT The GPT models both often provided nodes made out of multiple entities, sometimes even providing entire subclauses as nodes, such as `currently not known what had caused the pressure drop` or `leak today occurred on the nord stream 2 pipeline in the danish area`. Keeping the goal of knowledge graph development in mind, smaller connected entities are preferable. However, this is also a strength of this approach, as the baseline knowledge graph include occasional long entities (e.g., `German network regulator president` or `defunct russian-owned nord stream`

2 pipeline), which are only picked up by GPT. Especially in GPT4o the harder triples, such as Danish Maritime authorities, established, zone around the Nord Stream 2 pipeline are perfectly retrieved, as well as time determination such as February. On the other hand, the GPT model fails to include symmetry in the triples (Moscow, tension, Europe vs. Europe, tension, Moscow) in which object and subject are switched. It also labels some nouns as concept, such as tension, which in the golden standard is denoted as predicate. It is furthermore evident that the GPT4o model is more narrative and returns multiple output forms, which requires the prompt engineering and post-processing to be more elaborate compared to the other models.

5. Discussion

The set goal of the annotation process of both the news message and its simplified version, was to adhere to the text as much as possible. However, this approach has its limitations, as it precludes the inclusion of details that a graph developer might typically incorporate into the model. For example, by adhering to the annotation restrictions (see Section 3.1), world knowledge facts and meta-triples which abstract over what is in the text are not included. During annotation, some abstractions were made but they were left out of the dataset, due to the difficulty in objectively determining which additional information should be incorporated. This is always a limitation of manually creating knowledge graphs, as graph developers are influenced by their world and domain knowledge on deciding what is relevant to include. For extracted knowledge graphs, flattened results without abstractions are often acceptable, but for an ontology, abstractions and external world knowledge are essential. Some methods do have such additional knowledge. During evaluation, it was observed that KnowGL, and to lesser extent REBEL, include additional triples such as St. Petersburg is located in Russia. According to the annotated data, this is not counted as a correct triple. However, such additional triples containing external world knowledge might be valuable and desirable, depending on the use case.

Additionally, one thing considered during the annotation of the dataset, is that triples might not always be the best medium to describe the facts of complex natural language. For example, a sentence such as pressure in the pipeline dropped from 105 bars to 7 bars is annotated as two triples pressure in the pipeline, dropped from, 105 bars and pressure in the pipeline, dropped to, 7 bars. This might not be the optimal solution, as it might be better described as a quad or ternary predicate. The same holds for the annotation of DMA, established, zone around NS2P, which could also be annotated as DMA, established, zone and zone, around, NS2P. The direct link between the first zone and the second zone can be lost, which also occurs with specific operators (if both triples with for example an NSP1 and NSP2 operator exist). The same holds for triples with the predicate between for example. These predicates are not a unary relation, while this is necessary in a triple. Triples also lack the strength of meta-triples, such as the construct said that.

Furthermore, manually creating a knowledge graph is a step-by-step process. The considerations of what is considered a triple influence subsequent decisions. This affects the density of the knowledge graph, as a knowledge graph developer has an incentive to keep nodes consistent (e.g., considering Nord stream 2 pipeline to be the same node as The Nord stream 2 pipeline). Recreating the sequential nature of this process could be further researched, by tasking the models to generate the knowledge graph based on earlier considerations. While not all methods are suited for this, it is possible to prompt the GPT models with "based on the following nodes found in earlier sentences, extract the triples of the current sentence".

In the results, all methods on the complex text demonstrated higher recall scores on sentence-level extraction. This makes sense, as more triples are extracted when methods are ran on sentence level. Triple counts are included in the repository from this paper[2]. The complex text yielded much lower performance scores that the simple text. While to be expected, this is an indication that the effectiveness of relation extraction is influenced by the complexity of text. For node extraction, a similar pattern could be observed. Scores were also influenced by the fact that most methods are suitable for relation extraction, not necessarily node extraction, whereas KeyBERT scores high on the simple text, and has a high precision on the complex text. However, separate entities are not solely interesting, they have to be relevant for forming a knowledge graph. This is reflected in the qualitative results, where KeyBERT identifies `tension` as an entity, where ideally this is a relation. The GPT models outputs the nodes separately from the relations, resulting in some nodes that are not part of triples. Similarly to KeyBERT, from a graph developer's perspective this is not desirable. An advantage of KeyBERT and all relation extraction methods except GPT, is that depending on the method, parameters can be set — such as the amount of results to be extracted — giving more control over the results and making such methods more suitable for pipeline usage. Overall, the results from the GPT models were of decent quality, as can be seen from their high scores. However, the output content and its format differed per run and required manual post-processing to evaluate the results. This makes this method less suitable for use in a pipeline. The output consistency can be influenced to a certain extent by utilising few-shot learning and/or function calling. Further research into these consistency methods would make a valuable addition to this work.

The Graph Density results indicate that REBEL and KnowGL are able to generate high density knowledge graphs, which might be desirable depending on the use case. However, GPT-3.5 Turbo performs closest to the baseline density on simpler texts, while GPT-4 performs closest on complex texts. Note that with the complex text, the graphs tend to have very low CC_{avg} scores overall, because the larger and more complicated the text, the more possible nodes, and thus the denominator of the formula growing exponentially. While having denser graphs as results might not always be better, it is an interesting metric as there might be use cases where you would want a highly connected knowledge graph, or have your graphs meet a threshold of connectivity, for which REBEL and KnowGL might be more useful than GPT.

6. Conclusion and Future Work

In this paper, an annotated dataset was created containing different versions of a news message annotated with triples, as well as a comprehensive comparison of relation extraction methods within the context of this dataset. The primary objective of this study was to assess relation extraction methods on a real-life use case scenario, where the resulting graph reflects a manually created graph. The results indicate that the generative Large Language Models GPT-3.5 Turbo, GPT-4 and GPT-4o outperform the other tested relation extraction methods. Each with their own merits, as GPT-4o produces the best results in terms of F1-measure on complex texts, GPT-3.5 Turbo yields the closest graph density. However, in the qualitative analysis performed additionally to the evaluation metrics, it was observed that alternative approaches like REBEL and KnowGL exhibit strengths in leveraging external world knowledge to enrich the graph beyond the textual content alone. The findings and analysis indicate that while current methods cannot entirely replicate a manually created graph in a real-life scenario, GPT-4o's relation extraction capabilities demonstrate promising potential for future advancements.

For future work, several directions can be imagined: a comparison of a broader range of generative LLMs, exploring alternative methods for evaluating the quality and performance of extracted knowledge graphs, using different type of texts and larger and more diverse real-world use cases and adding more pre- and post-processing steps. Looking forward, such avenues offer significant potential to further advance the field of knowledge graph extraction.

Acknowledgements

We extend our gratitude to the NATO Science & Technology Organization Centre for Maritime Research & Experimentation (CMRE) for providing us with a practical use case, and we extend special thanks to Dr. Pawel Kowalski for his guidance in the maritime domain and previous work. Additionally, we would like to thank the *DEFRAUDify*, *TrustLLM*, and *RVO Causal Relations in Behavioural Modelling* projects for their support.

References

[1] F.N. AL-Aswadi, H.Y. Chan and K.H. Gan, From Ontology to Knowledge Graph Trend: Ontology as Foundation Layer for Knowledge Graph, in: *Iberoamerican Knowledge Graphs and Semantic Web Conference*, Springer, 2022, pp. 330–340.

[2] T. Al-Moslmi, M.G. Ocaña, A.L. Opdahl and C. Veres, Named entity extraction for knowledge graphs: A literature overview, *IEEE Access* **8** (2020), 32862–32881.
[3] D. Alivanistos, S.B. Santamaría, M. Cochez, J.-C. Kalo, E. van Krieken and T. Thanapalasingam, Prompting as Probing: Using Language Models for Knowledge Base Construction, 2023.
[4] B.P. Allen, L. Stork and P. Groth, Knowledge Engineering using Large Language Models (2023).
[5] M.N. Asim, M. Wasim, M.U.G. Khan, W. Mahmood and H.M. Abbasi, A survey of ontology learning techniques and applications, *Database, The Journal of Biological Databases and Curation* **2018** (2018), 1–24. doi:10.1093/database/bay101.
[6] N. Bach and S. Badaskar, A review of relation extraction, *Literature review for Language and Statistics II* **2** (2007), 1–15.
[7] R.R. Bakker, Knowledge Graphs: representation and structuring of scientific knowledge, PhD thesis, University of Twente, 1987.
[8] R.M. Bakker and M.H.T. de Boer, Dynamic Knowledge Graph Evaluation, *TechRxiv preprint* (2024), Under review at IEEE Transactions on Knowledge and Data Engineering. https://www.techrxiv.org/users/791451/articles/1070833-dynamic-knowledge-graph-evaluation.
[9] R.M. Bakker and D.L. Di Scala, From Text to Knowledge Graph: Comparing Relation Extraction Methods in a Practical Context, in: *First International Workshop on Generative Neuro-Symbolic AI, co-located with ESWC 2024*, Hersonissos, Crete, Greece, 2024.
[10] R.M. Bakker, M.H. de Boer, A.P. Meyer-Vitali, B.J. Bakker and S.A. Raaijmakers, A Hybrid Approach for Creating Knowledge Graphs: Recognizing Emerging Technologies in Dutch Companies, *HHAI2022: Augmenting Human Intellect* (2022), 307–309.
[11] R.M. Bakker, G.J. Kalkman, I. Tolios, D. Blok, G.A. Veldhuis, S. Raaijmakers and M.H.T. de Boer, Exploring Knowledge Extraction Techniques for System Dynamics Modelling: Comparative Analysis and Considerations, in: *Proceedings of the Benelux Conference on Artificial Intelligence (BNAIC)*, 2023.
[12] K. Barker and N. Cornacchia, Using noun phrase heads to extract document keyphrases, in: *Advances in Artificial Intelligence: 13th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2000 Montéal, Quebec, Canada, May 14–17, 2000 Proceedings 13*, Springer, 2000, pp. 40–52.
[13] S. Bills, N. Cammarata, D. Mossing, H. Tillman, L. Gao, G. Goh, I. Sutskever, J. Leike, J. Wu and W. Saunders, Language models can explain neurons in language models — openaipublic.blob.core.windows.net, 2023, [Accessed 24-08-2023].
[14] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter and D. Amodei, Language models are few-shot learners, *Advances in neural information processing systems* **33** (2020), 1877–1901.
[15] P. Buitelaar, P. Cimiano and B. Magnini, *Ontology learning from text: methods, evaluation and applications*, Vol. 123, IOS press, 2005.
[16] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, A. Oprea and C. Raffel, Extracting Training Data from Large Language Models, 2021.
[17] M. De Boer and J. Verhoosel, Creating and evaluating data-driven ontologies, *International Journal on Advances in Software* **12**(3–4) (2019).
[18] A. De Nicola, R. Zgheib and F. Taglino, Toward a knowledge graph for medical diagnosis: issues and usage scenarios, in: *Semantic Models in IoT and Ehealth Applications*, Elsevier, 2022, pp. 129–142.
[19] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805* (2018).
[20] H. Elsahar, P. Vougiouklis, A. Remaci, C. Gravier, J. Hare, F. Laforest and E. Simperl, T-rex: A large scale alignment of natural language with knowledge base triples, in: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.
[21] E.A. Feigenbaum, The art of artificial intelligence: themes and case studies of knowledge engineering, in: *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'77, Morgan Kaufmann Publishers Inc., 1977, pp. 10141029–.
[22] F. Fürst and F. Trichet, Heavyweight ontology engineering, in: *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops: OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, COMINF, IS, KSinBIT, MIOS-CIAO, MONET, OnToContent, ORM, PerSys, OTM Academy Doctoral Consortium, RDDS, SWWS, and SeBGIS 2006, Montpellier, France, October 29-November 3, 2006. Proceedings, Part I*, Springer, 2006, pp. 38–39.
[23] P. Gamallo, M. Gonzalez, A. Agustini, G. Lopes and V.S. De Lima, Mapping syntactic dependencies onto semantic relations, in: *Proceedings of the ECAI workshop on machine learning and natural language processing for ontology engineering*, 2002, pp. 15–22.
[24] F. Giunchiglia and I. Zaihrayeu, Lightweight ontologies (2007).
[25] M. Grootendorst, Keyword Extraction with BERT, 2020, Accessed on: 07-03-2024. https://towardsdatascience.com/keyword-extraction-with-bert-724efca412ea.
[26] C. Guéret, P. Groth, C. Stadler and J. Lehmann, Assessing linked data mappings using network measures, in: *The Semantic Web: Research and Applications: 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings 9*, Springer, 2012, pp. 87–102.
[27] A. Hari and P. Kumar, WSD Based Ontology Learning from Unstructured Text Using Transformer, *Procedia Computer Science* **218** (2023), 367–374.
[28] M.A. Hearst, Automatic acquisition of hyponyms from large text corpora, in: *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*, 1992.
[29] P.-L. Huguet Cabot and R. Navigli, REBEL: Relation Extraction By End-to-end Language generation, in: *Findings of the Association for Computational Linguistics: EMNLP 2021*, Association for Computational Linguistics, 2021, pp. 2370–2381.
[30] M.Y. Jaradeh, K. Singh, M. Stocker, A. Both and S. Auer, Information extraction pipelines for knowledge graphs, *Knowledge and Information Systems* **65**(5) (2023), 1989–2016.

[31] S. Ji, S. Pan, E. Cambria, P. Marttinen and S.Y. Philip, A survey on knowledge graphs: Representation, acquisition, and applications, *IEEE transactions on neural networks and learning systems* **33**(2) (2021), 494–514.
[32] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y.J. Bang, A. Madotto and P. Fung, Survey of Hallucination in Natural Language Generation, *ACM Computing Surveys* **55**(12) (2023). doi:10.1145/3571730.
[33] A.C. Khadir, H. Aliane and A. Guessoum, Ontology learning: Grand tour and challenges, *Computer Science Review* **39** (2021), 100339.
[34] A.C. Khadir, H. Aliane and A. Guessoum, Ontology learning: Grand tour and challenges, *Computer Science Review* **39** (2021), 100339.
[35] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov and L. Zettlemoyer, BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, *arXiv preprint arXiv:1910.13461* (2019).
[36] Y. Ma, A. Wang and N. Okazaki, DREEAM: Guiding Attention with Evidence for Improving Document-Level Relation Extraction, *arXiv preprint arXiv:2302.08675* (2023).
[37] C.D. Manning, Part-of-speech tagging from 97% to 100%: is it time for some linguistics?, in: *International conference on intelligent text processing and computational linguistics*, Springer, 2011, pp. 171–189.
[38] M. McDaniel and V.C. Storey, Evaluating domain ontologies: clarification, classification, and challenges, *ACM Computing Surveys (CSUR)* **52**(4) (2019), 1–44.
[39] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado and J. Dean, Distributed representations of words and phrases and their compositionality, *Advances in neural information processing systems* **26** (2013).
[40] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi and L. Zettlemoyer, Rethinking the Role of Demonstrations: What makes In-context Learning Work?, in: *EMNLP*, 2022.
[41] C. Niklaus, M. Cetto, A. Freitas and S. Handschuh, A Survey on Open Information Extraction, *arXiv preprint arXiv:1806.05599* (2018).
[42] J. Nivre, M.-C. De Marneffe, F. Ginter, Y. Goldberg, J. Hajic, C.D. Manning, R. McDonald, S. Petrov, S. Pyysalo, N. Silveira et al., Universal dependencies v1: A multilingual treebank collection, in: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 1659–1666.
[43] T. Nomoto, Keyword extraction: a modern perspective, *SN Computer Science* **4**(1) (2022), 92.
[44] OpenAI, GPT-4 Technical Report, 2023.
[45] OpenAI, Hello GPT-4o, 2024, Were announcing GPT-4o, our new flagship model that can reason across audio, vision, and text in real time. https://openai.com/index/hello-gpt-4o/.
[46] J.Z. Pan, S. Razniewski, J.-C. Kalo, S. Singhania, J. Chen, S. Dietze, H. Jabeen, J. Omeliyanenko, W. Zhang, M. Lissandrini, R. Biswas, G. de Melo, A. Bonifati, E. Vakaj, M. Dragoni and D. Graux, Large Language Models and Knowledge Graphs: Opportunities and Challenges, 2023.
[47] N. Peng, H. Poon, C. Quirk, K. Toutanova and W.-t. Yih, Cross-sentence n-ary relation extraction with graph lstms, *Transactions of the Association for Computational Linguistics* **5** (2017), 101–115.
[48] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown and Y. Shoham, In-Context Retrieval-Augmented Language Models, 2023.
[49] A. Rector, N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang and C. Wroe, OWL Pizzas: Common errors & common patterns from practical experience of teaching OWL-DL, in: *Proceedings of the European Knowledge Acquisition Workshop (EKAW-2004)*, Springer Verlag, Northampton, England, 2004.
[50] G. Rossiello, M.F.M. Chowdhury, N. Mihindukulasooriya, O. Cornec and A.M. Gliozzo, Knowgl: Knowledge generation and linking from text, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37, 2023, pp. 16476–16478.
[51] G. Salton, C.-S. Yang and C.T. Yu, Contribution to the theory of indexing, Technical Report, Cornell University, 1973.
[52] A. Singhal, Introducing the Knowledge Graph: things, not strings, 2012. https://blog.google/products/search/introducing-knowledge-graph-things-not/.
[53] R. Studer, V.R. Benjamins and D. Fensel, Knowledge engineering: Principles and methods, *Data & knowledge engineering* **25**(1–2) (1998), 161–197.
[54] Y. Tian, G. Chen, Y. Song and X. Wan, Dependency-driven relation extraction with attentive graph convolutional networks, in: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, pp. 4458–4471.
[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser and I. Polosukhin, Attention is All you Need, in: *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 5998–6008.
[56] Z. Wan, F. Cheng, Z. Mao, Q. Liu, H. Song, J. Li and S. Kurohashi, GPT-RE: In-context learning for relation extraction using large language models, *arXiv preprint arXiv:2305.02105* (2023).
[57] C. Wang, X. Liu, Z. Chen, H. Hong, J. Tang and D. Song, DeepStruct: Pretraining of language models for structure prediction, *arXiv preprint arXiv:2205.10475* (2022).
[58] J. Wang and W. Lu, Two are better than one: Joint entity and relation extraction with table-sequence encoders, *arXiv preprint arXiv:2010.03851* (2020).
[59] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E.H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean and W. Fedus, Emergent Abilities of Large Language Models, 2022.
[60] W. Wong, W. Liu and M. Bennamoun, Ontology learning from text: A look back and into the future, *ACM computing surveys (CSUR)* **44**(4) (2012), 1–36.
[61] X. Zou, A survey on application of knowledge graph, in: *Journal of Physics: Conference Series*, Vol. 1487, IOP Publishing, 2020, p. 012016.