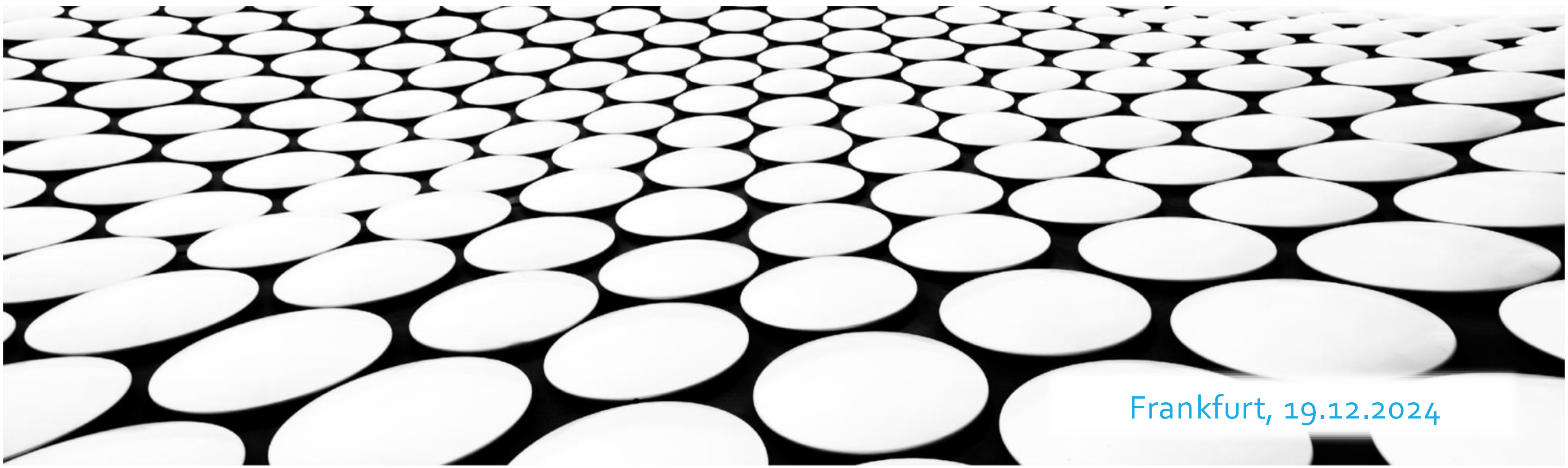
Master-Thesis:

Constructing a Knowledge Graph by extracting information from financial news articles

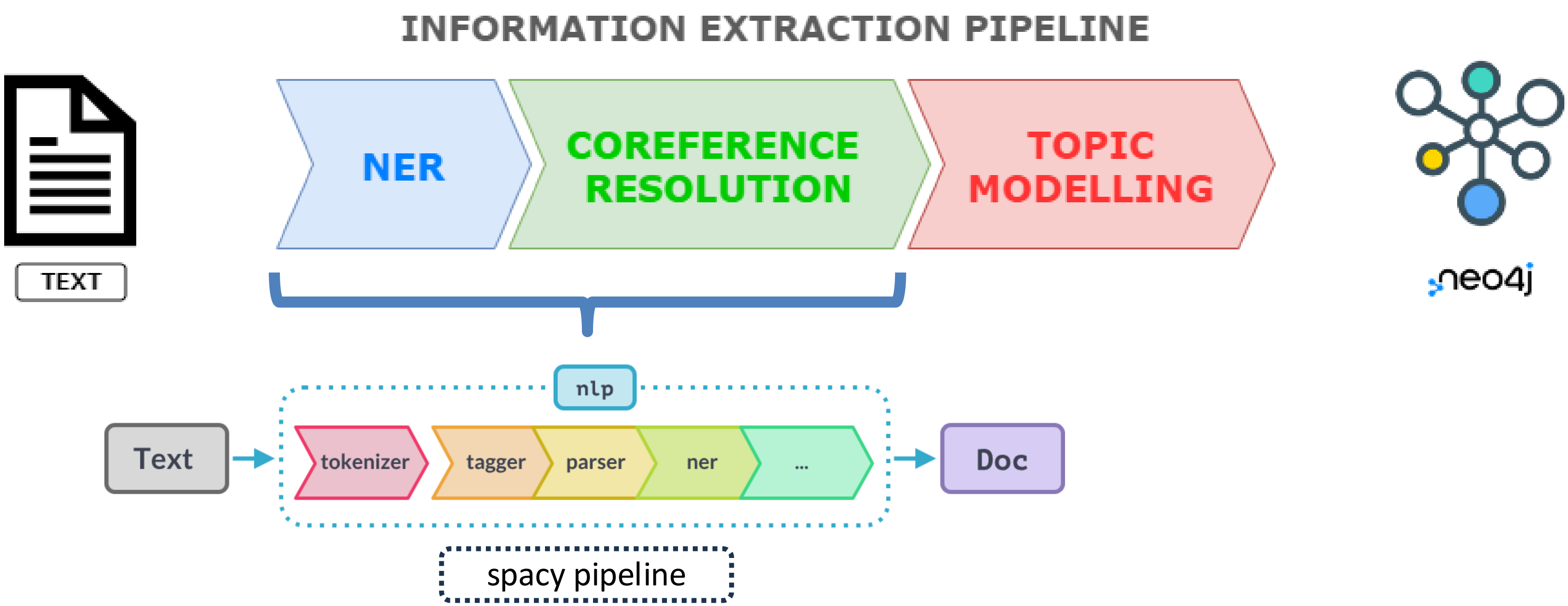
Rainer Gogel
Matr.Nr. 1272442



Frankfurt, 19.12.2024

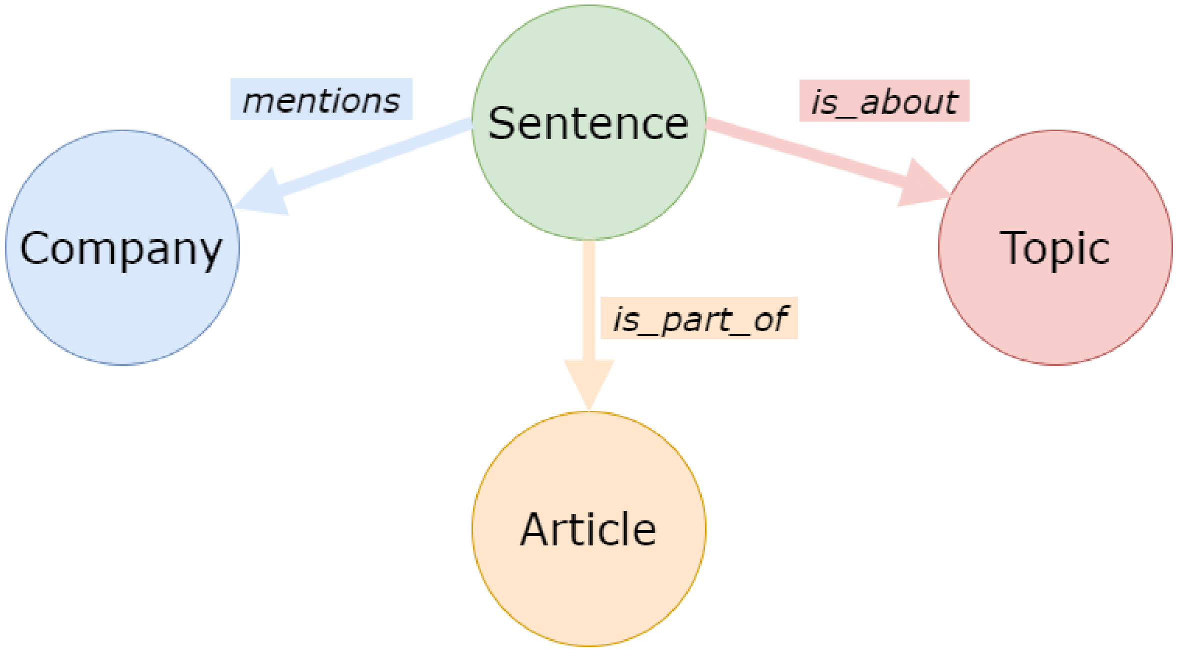
1. Overview

Information Extraction Pipeline and spacy pipeline



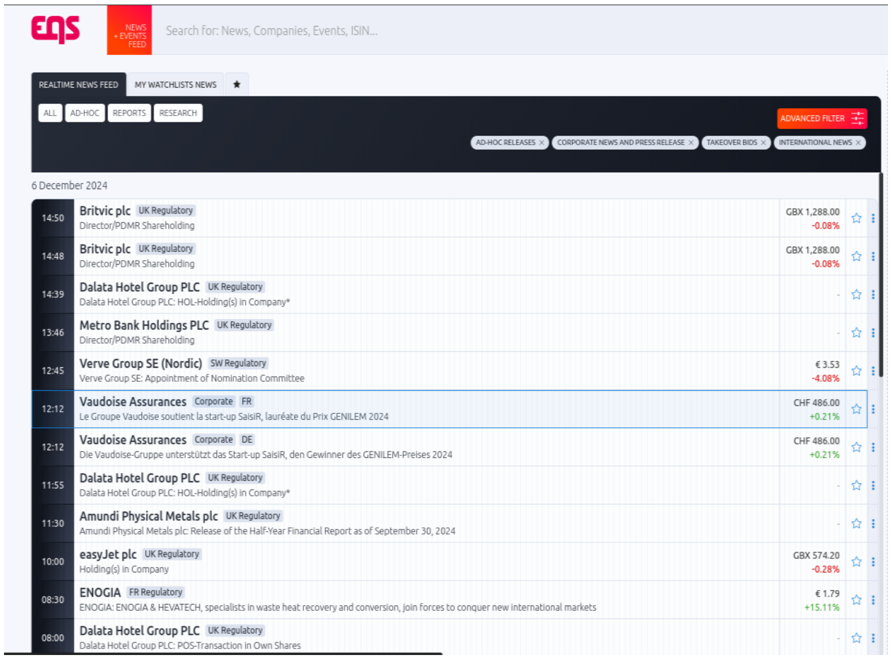
- From <u>unstructured text</u> to a <u>structured representation</u> in a Knowledge Graph
- Components: NER, Coreference Resolution, Topic Modelling
- NER, Coreference Resolution in spacy pipeline

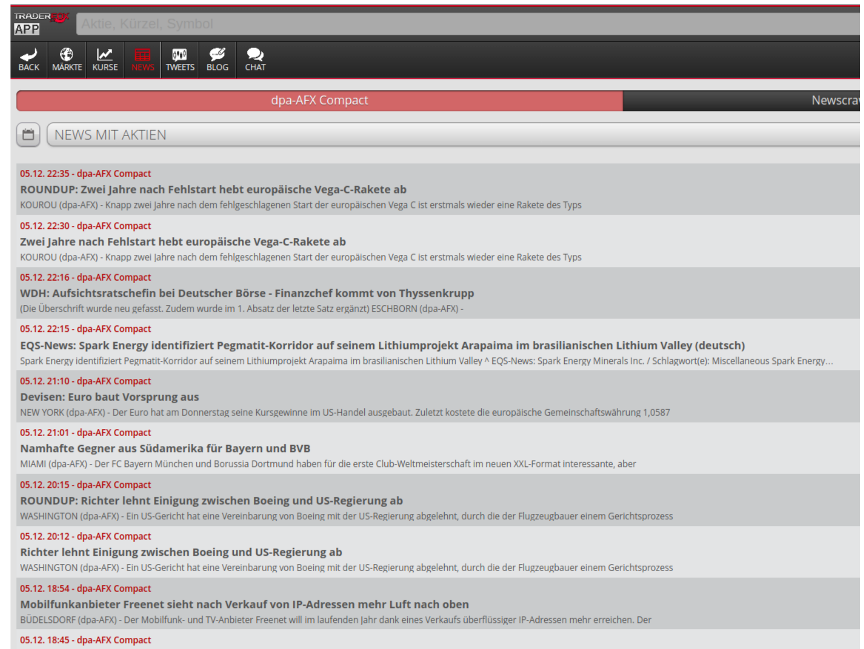
Knowledge Graph



- **Neo4j Graph Database**
- **Nodes**: Sentence, Company, Article, Topic
- **Relationships**: mentions, is_part_of, is_about

Financial News Articles





- **Language: Mostly German, sometimes English**
- **EQS**: https://www.eqs-news.com/
- **dpa compact**: https://mobile.traderfox.com/news/dpa-compact/

Companies

	symbol	name	market_cap	sector	industry
1	ENGQF	Engie SA	40486721464	Utilities	Diversified Utilities
2	AMUN.PA	Amundi	13606116250	Financial Services	Asset Management
3	GECFF	Gecina Société anonyme	8040674363	Real Estate	REIT - Office
4	GFC.PA	Gecina	7024326032	Real Estate	REIT - Office
5	GI6A.DU	Gecina Nom	7002167275	Real Estate	REIT - Industrial
6	NEOEN.PA	Neoen S.A.	5871329061	Utilities	Renewable Utilities
7	SPIE.PA	SPIE SA	5721606240	Industrials	Engineering & Construction
8	COV.PA	Covivio	5403592988	Real Estate	REIT - Diversified
9	ELIS.PA	Elis SA	5065802383	Industrials	Specialty Business Services
10	RF.PA	Eurazeo SE	5044525719	Financial Services	Asset Management
11	AYV.PA	Ayvens	4777173278	Industrials	Rental & Leasing Services
12	CBDG.PA	Compagnie du Cambodge	4002105250	Industrials	Railroads
13	BNJ.AS	BANIJAY GROUP N.V.	3915025648	Communication Services	Entertainment
14	TKO.PA	Tikehau Capital	3853375478	Financial Services	Asset Management
15	ATE.PA	Alten S.A.	3450427445	Technology	Information Technology Services
16	ITP.PA	Interparfums SA	3357486088	Consumer Defensive	Household & Personal Products
17	VRLA.PA	Verallia Société Anonyme	3146123341	Consumer Cyclical	Packaging & Containers
18	IDL.PA	ID Logistics Group SA	2786407983	Industrials	Specialty Business Services
19	FLY.PA	Société Foncière Lyonnaise	2757306733	Real Estate	REIT - Office
20	CRTO	Criteo S.A.	2650379575	Communication Services	Advertising Agencies

- **Source: OpenBB**: https://openbb.co/products/platform
- **> 2500 European Companies**

2. Information Extraction Pipeline

Test different approaches for each pipeline component

- **Traditional approaches**
  - Rule-based
  - Traditional Machine Learning: HMMs, CRFs, etc.
  - REGEX

- **Pre-Trained LLMs**
  - Use or fine-tune pre-trained LLMs: BERT, etc.

- **Generative LLMs**
  - Prompt LLMs --> Response

- **Best approach?**

A. NER


INFORMATION EXTRACTION PIPELINE
TEXT
NER
COREFERENCE RESOLUTION
TOPIC MODELLING
neo4j

Traditional approach: REGEX

4imprint Group plc.

[4imprint, Group, plc]

CHAR_TYPE_CASE_MIX | COMMON_WORD | LEGAL_TERM

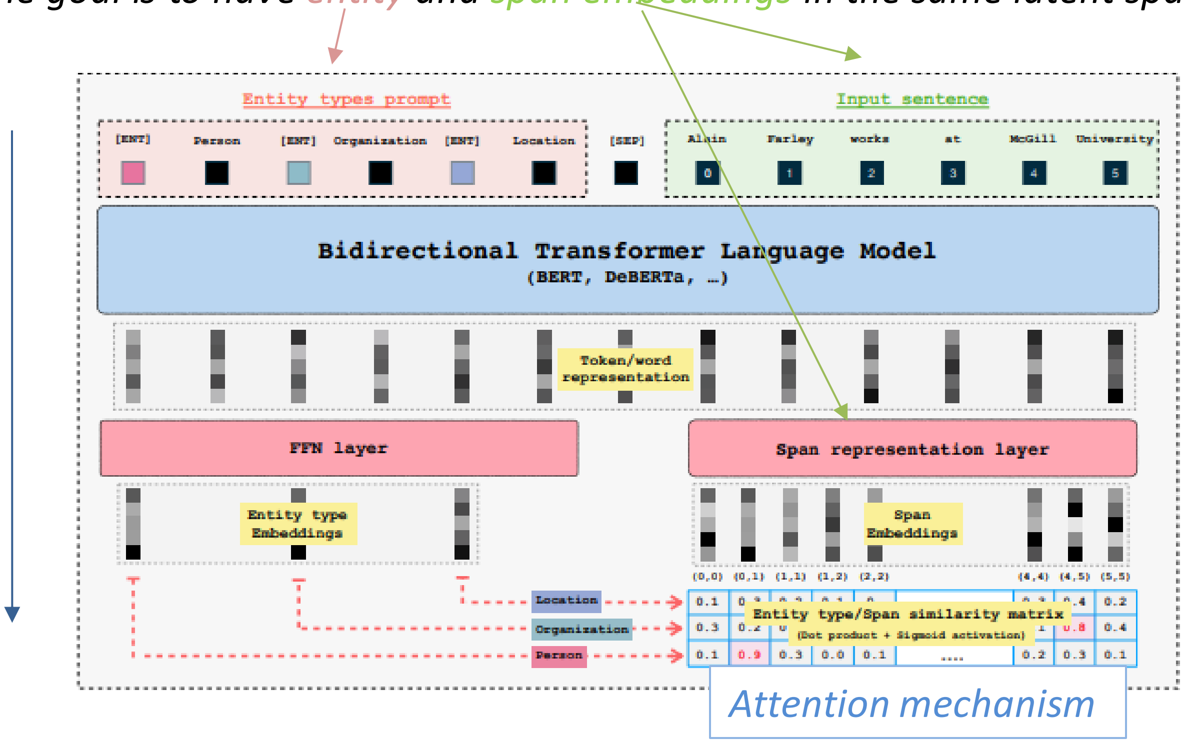
```
(?:\b4imprint|4Imprint\b)\s*(?i:\bGroup\b)?\s*(?i:\bplc\b)?
```

```
(?P<SYMB_FOUR_DOT_L>(?:(?<=\\s)|(?<=^))
(?:(?:\\b4imprint|4Imprint\\b)\\s*(?i:\\bGroup\\b)?\\s*(?i:\\bplc\\b)?)(?=\\s))
```

- Make REGEX-patterns: Classify each term, make optionality dependent on class
- Save and use REGEX-patterns (JSONL) in spacy pipeline

Pre-Trained Model: GliNER

"The goal is to have entity and span embeddings in the same latent space..."



- **Input**: Entity type names -concat- input sentence
- Entity Embeddings • Span Representation: **Learned Similarity Matrix**

DEMO

1) Creation of REGEX patterns
2) Performance REGEX vs. GliNER Pre-Trained model

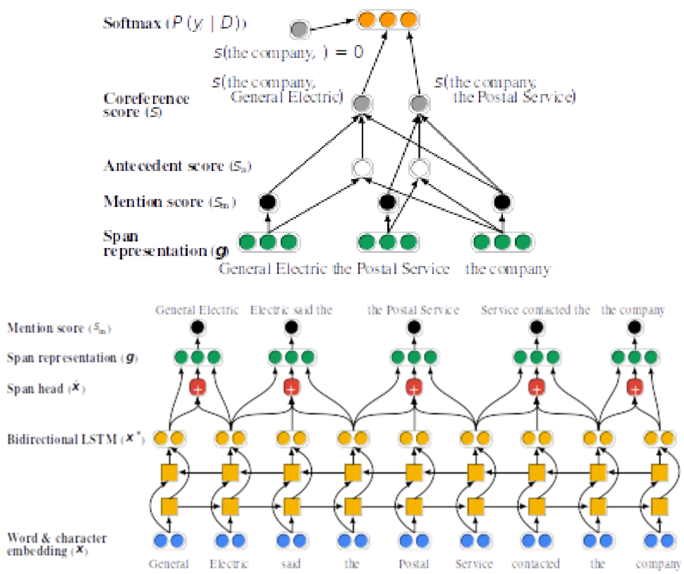
B. Coreference Resolution

Pre-Trained: Crosslingual Coreference

End-to-end Neural Coreference Resolution Architecture (Lee 2017)

2.

1.



Embedding Model:
microsoft/Multilingual-MiniLM-L12-H384

AllenNLP
Coref-Model
(2021)

+

spaCy

POS tagger

Crosslingual-Coreference
(Berenstein)

Output: Clusters:
[[[cl1 start1, cl1 end1], [cl1 start2, cl1 end2]],
[[cl2 start1, cl2 end1], [cl2 start2, cl2 end2]]]

- e2e-model: LSTM, later SpanBERT/MiniLM **->** Coreference Clusters
- Crosslingual Coreference: Only <u>noun-phrases</u> of e2e-clusters

Generative LLM

LangChain

Data Model + Few-Shot Examples + Prompt + GenLLM

- **LLM-Framework**: LangChain

Generative LLM: Data Model

```python
from pydantic import BaseModel, Field

class Coreference(BaseModel):
    """..."""
    coref_text: Optional[str] = Field(default=None, description='The coreference substring in the text string')
    coref_with_surroundings: Optional[str] = Field(default=None, description='The coreference substring plus two words to the left and right.')
class ClusterHead(BaseModel):
    """..."""
    head_text: Optional[str] = Field(default=None, description='The string characters of the cluster head which is a company name')
    head_index_start: Optional[int] = Field(default=None, description='The position index of the first character of the cluster head substring')
    head_index_end: Optional[int] = Field(default=None, description='The position index of the last character of the cluster head substring plus one')

class Cluster(BaseModel):
    """..."""
    cluster_id: Optional[int] = Field(default=None, description='The identification number of the cluster provided by the user. '
                                                                'Always return the same number that was provided by the user.')
    text: Optional[str] = Field(default=None, description='The text to search in')
    cluster_head: Optional[ClusterHead] = Field(default=None, description='The cluster object which is is provided in the user message')
    coreferences: Optional[list[Coreference]] = None

class DataContainer(BaseModel):
    data_list: list[Cluster] = []
```

- **Pydantic BaseModel:** Type checking in Examples and Return Format
- **Coreference surroundings:** Indicate it with two words on each side

Generative LLM: Few Shot Examples

```
examples = [
    Cluster(cluster_id=101, text='Der Abschwung im PC-Markt erwischt auch den Chipkonzern AMD. Im vergangenen Quartal sank der Umsatz
        cluster_head=ClusterHead(head_text='Chipkonzern AMD', head_index_start=44, head_index_end=59), coreferences=[Coreference(core

    Cluster(cluster_id=22, text='MicroVision, Inc., ein fuehrender Anbieter von MEMS-basierten Solid-State-Lidar- und Fahrerassistenz
        cluster_head=ClusterHead(head_text='MicroVision, Inc.', head_index_start=0, head_index_end=17), coreferences=[Coreference(cor

    Cluster(cluster_id=303, text='Der Oelkonzern BP hat im ersten Quartal die niedrigeren Oel- und Gaspreise zu spueren bekommen. Der
        cluster_head=ClusterHead(head_text='Oelkonzern BP', head_index_start=4, head_index_end=17),
        coreferences=[Coreference(coref_text='BP', coref_with_surroundings='Geldzuflusses kuendigte BP am Dienstag'), Coreference(cor
            Coreference(coref_text='Konzern', coref_with_surroundings='setzt der Konzern seine Strategie'), Coreference(coref_text='s
            Coreference(coref_text='es', coref_with_surroundings='an, dass es Geschaefte vereinbaren')]),

    Cluster(cluster_id=54, text="Abivax SA, ein Biotechnologieunternehmen mit einem Produkt in der klinischen Phase 3, das Therapien
        cluster_head=ClusterHead(head_text='Abivax SA', head_index_start=0, head_index_end=9),
        coreferences=[Coreference(coref_text='ein Biotechnologieunternehmen', coref_with_surroundings='Abivax SA, ein Biotechnologieu
            Coreference(coref_text='Wir', coref_with_surroundings='Abivax, sagte: Wir sind stolz'), Coreference(coref_text='Wir', cor
            Coreference(coref_text='uns', coref_with_surroundings='Es ermutigt uns, dass die'), Coreference(coref_text='unsere', core
  ]
```

- **ClusterHead:** Company name found by previous NER pipeline component

Generative LLM: Prompt + GenLLM

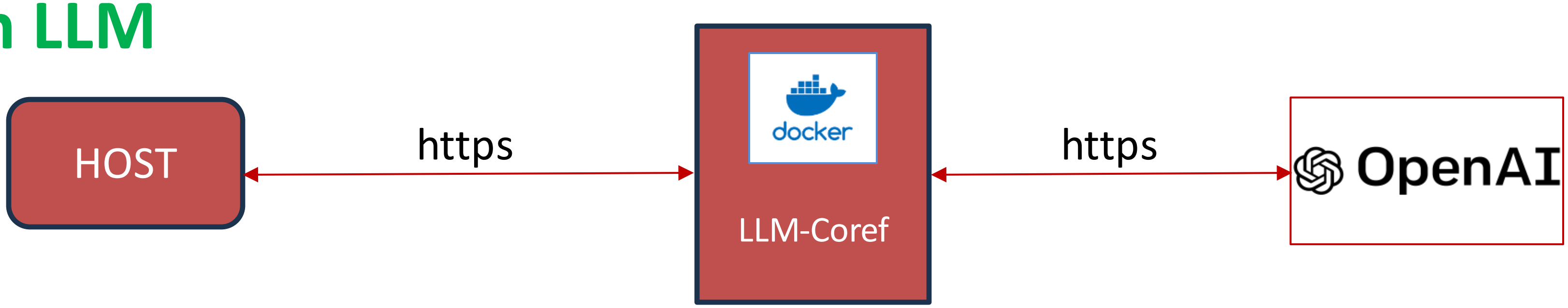
```python
class CorefLangchain:
    def __init__(self, prompt_template: str, model_name: str = "gpt-4o"):
        nest_asyncio.apply()
        self.prompt = PromptTemplate(template=prompt_template,
                                     input_variables=["text", "cluster_id", "cluster_head"])
        self.llm = ChatOpenAI(temperature=0, model=model_name, openai_api_key=os.getenv('OPENAI_API_KEY'))
        self.llm = self.llm.with_structured_output(schema=Cluster)
        self.chain = self.prompt | self.llm
        self.examples: list[BaseMessage] = convert_examples_to_messages()
```

- **Return Format:** Cluster instance
- **Examples**: Converted to LangChain messages
- **Chain**: Prompt + OpenAI *gpt-4o*
- **Input**: Text and ClusterHead : Company name previousely found in NER

Containerization due to dependency issues

Pre-Trained

Gen LLM



- **Crosslingual Coreference:** Request to docker container
- **Generative LLM**: Request to docker container and OpenAI server

DEMO

- Performance Pre-Trained Crosslingual-Coreference vs. Generative LLM

C. Topic Modelling

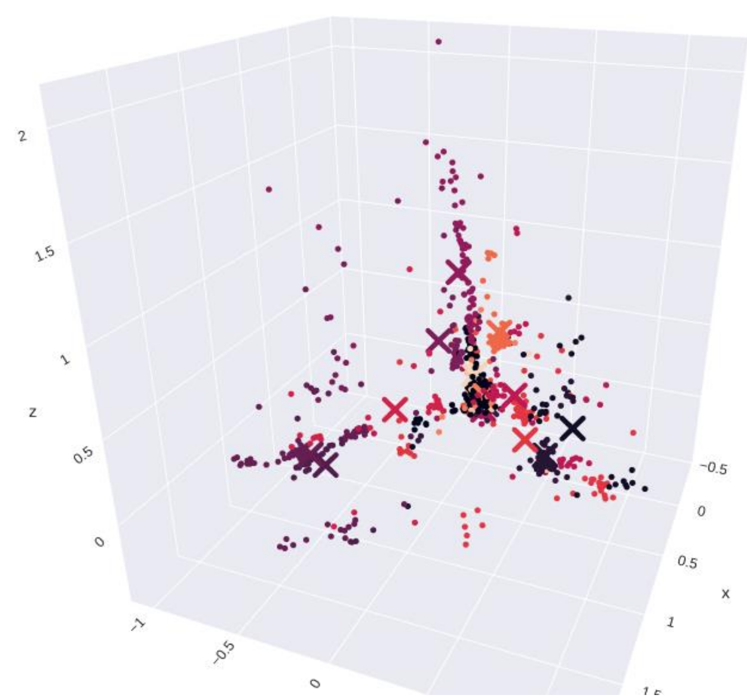
Traditional Topic Modelling und BERTopic

1.A.: Word Vectors (TF-IDF, Bag-of-Words)
OR
1.B.: Embeddings

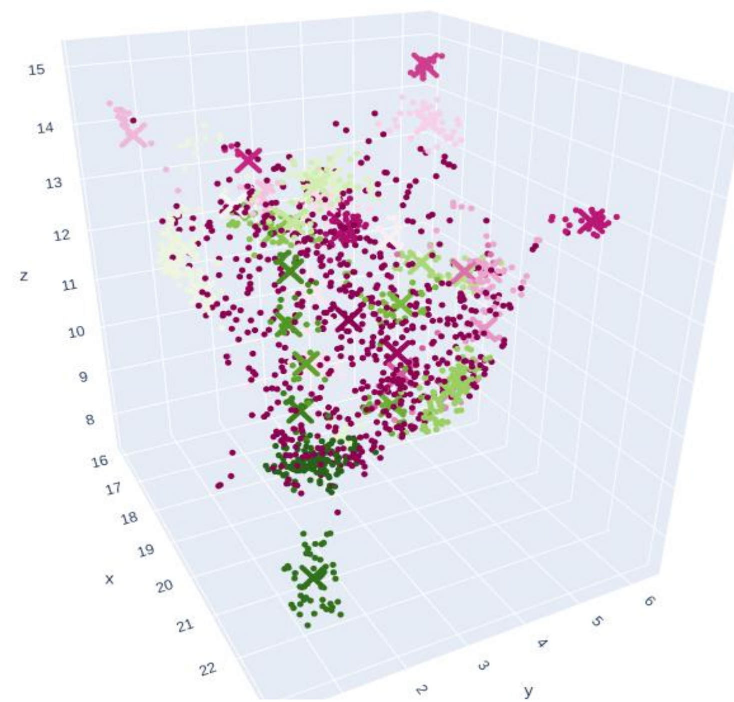
2. **Dimension Reduction**: Choose and apply a dimension reduction method on the embeddings
3. **Clustering**: Choose and apply a clustering algorithm on the dimension-reduced embeddings
4. **Aggregate Text**: Aggregate the text of all documents within each cluster
5. **Apply TF-IDF Vectorization**: Apply TF-IDF vectorization to each of the per-Cluster-aggregated texts [1]
6. **Most Frequent Words**: Get the most frequent words for each cluster according to TF-IDF

- **Traditional**: Features: Word Counts (TF-IDF, One-Hot, Bag-of-Words)
- **BERTopic**: Features: Embeddings

Traditional Topic Modelling und BERTopic



TF-IDF



Embeddings

- Disappointing Results

Generative LLM

LangChain

Data Model + Few-Shot Examples + Prompt + GenLLM

- **LLM-Framework**: LangChain

Generative LLM: Data Model

```
class Frame(BaseModel):
    """ DataFrame that contains the index of the DataFrame and the column "top_sent" which contains the sentences for whic
    indexes: list[int] = Field(description='The indexes of the rows in the pandas DataFrame')
    sentences: list[str] = Field(default=None, description='List of sentences each for which the Topic shall be determined
    topics: list[Topic] = Field(default=None, description='List of Topic enums for each sentence in "sentences". List must
```

```
class TopicExplain(str, Enum):
    """ The Topic of the sentence. Topics can only be one of the following: """
    topic1 = ("Sätze mit konkreten Zahlenangaben aus Quartals- oder Jahresberichten. Die genannten Zahlen beziehen sich auf die Bilanz, den U
              "Beispiele dafür sind EBIT, EBITDA, Gewinn oder Verlust vor Steuern, Gewinn- oder Verlustmargen, der Umsatz, Veränderungen der
    topic2 = "Sätze mit allgemeinen Aussagen und Einschätzungen zu Unternehmensergebnissen, die Bilanzerung und den Umsatz. Dies sind Wertung
    topic3 = ("Sätze, die sich auf eine bevorstehende oder vergangene Hauptversammlung oder die Veröffentlichung von Unternehmensergebnissen
              "Beispiele dafür sind die Ankündigung einer Veröffentlichung von Quartals- oder Jahresberichten oder Informationen zu bzw. Über
    topic4 = "Zukunftsgerichteter Ausblick, Prognosen, Ziele, Strategie und Pläne der Unternehmensleitung."
    topic5 = "Sätze, die Kennzahlen zu Unternehmensergebnissen beinhalten, ohne dass dabei ganze Sätze gebildet werden oder die Zahlen beschr
    topic6 = "Sätze, in denen die Aktivitäten und das Profil des Unternehmens dargestellt wird. Oft dienen die Sätze der positiven Selbstdars
    topic7 = "Stimmrechte, Kapitalveränderungen, Dividenden, Finanzierung, Listing an Börsen, Marktkapitalisierung."
    topic8 = "Sätze, in denen das vom Unternehmen angebotene Produkt, eine Produktentwicklung oder ein neue Neuerung im Hinblick auf ein Prod
    topic9 = "Sätze, in denen die Herstellung des Produkts, der Produkt-Forschung, die Exploration vn Bodenschätzen, Produkt- oder Medikament
    topic10 = "Konzernumbau, wichtige organisatorische Veränderungen, Restrukturierung, Werksstilllegung, strategische Partnerschaften, Übern
    topic11 = "Personalveränderungen im Vorstand, Aufsichtsrat, Betriebsrat oder anderer Organe im Unternehmen, Personal, Gewerkschaftem, Str
    topic12 = "Kunden, Marktanteile, Absatzmärkte, Umsätze, Absatzpreise"
    topic13 = "Einflüsse von Aussen auf die Erfolgsaussichten von Unternehmen etwa durch Subventionen, Staatliche Eingriffe, Umbrüche im Mark
    topic14 = "Einschätzungen Unternehmensfremder/Analysten zu einem Unternehmen"
    topic15 = "Unfälle, Gewalt, Katastrophen"
    topic16 = "Unvollständige Sätze mit einzelnen, nicht-zusammenhängenen Worten, ohne Kontext, die wahrscheinlich falsch formattiert oder im
    topic17 = "Alle anderen topics, die den oben genannten 16 topics nicht zugeordnet werden können."
```

- **Frame:** Instance of pandas DataFrame
- **Topics:** 17 topics. Topic 16: Incomplete sentences, Topic 17: OTHER

Generative LLM: Few Shot Examples

```
# Note: Konzernumbau, wichtige organisatorische Veränderungen, Restrukturierung, Werksstillllegung, strategische Partnerschaften, Übernahmen
top10 = [
'Comp@Name@Placeholder verwies auf die Schliessung eines Comp@Name@Placeholder-Werks in Bridgend sowie die Verlegung der Produktion nach Chi
'Der Kauf der Comp@Name@Placeholder stellt eine hervorragende Ergaenzung zu unserem wachsenden Netzwerk an internationalen Laborpartnern dar
'Nach der bereits erfolgten Verlegung zentraler Funktionen der Gesellschaft an den Standort Hamburg beabsichtigt die Comp@Name@Placeholder, 
'Comp@Name@Placeholder, ein molekulargenetisches Diagnostikunternehmen, das sich auf die Krebsfrueherkennung spezialisiert hat, hat heute di
'Die Comp@Name@Placeholder übernimmt die Schweizer Comp@Name@Placeholder Gruppe und erweitert damit ihre Kernkompetenz im Bereich der Luftqu
'Die Comp@Name@Placeholder ist ab sofort Teil der Ingenieur-, Architektur- und Managementberatungsfirma Comp@Name@Placeholder.',
'Comp@Name@Placeholder, eines der weltweit führenden Marktforschungsunternehmen, hat ein freiwilliges öffentliches Übernahmeangebot für die 
'Comp@Name@Placeholder: Comp@Name@Placeholder und Comp@Name@Placeholder unterzeichnen ihre vierte gemeinsame Vereinbarung.',
'Comp@Name@Placeholder und Comp@Name@Placeholder haben ein verbindliches Eckpunktepapier fuer die erste Phase eines mehrphasigen Projekts zu
'Am 13. Mai 2022 jaehrt sich der Tag, an dem die Comp@Name@Placeholder Insolvenz anmelden musste bereits zum sechsten Mal.',
```

- **Examples:** Multiple examples for each of the 17 topics

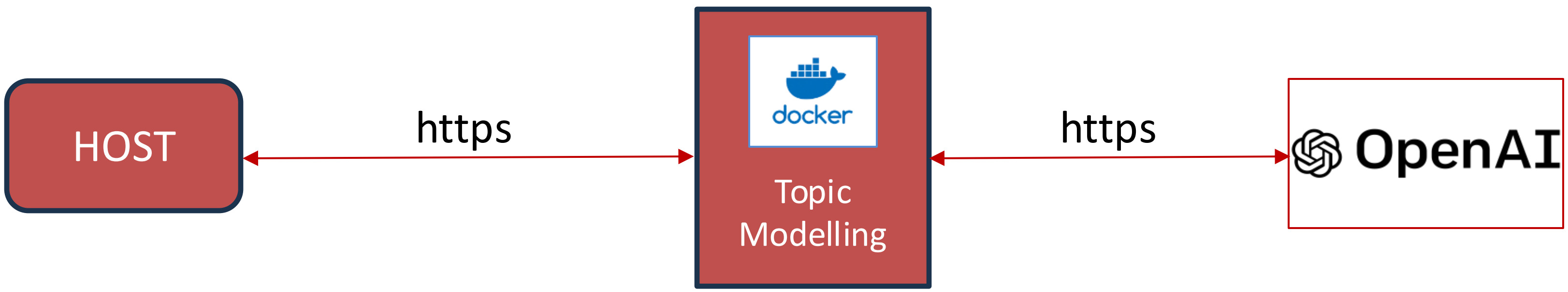
Generative LLM: Prompt + GenLLM

LangChain

```
class TopicLangchain:
    def __init__(self, prompt_template: str, model_name: str = "gpt-4o"):
        nest_asyncio.apply()
        self.prompt = PromptTemplate(template=prompt_template, input_variables=['user_data', "topics"]).par
        self.llm = ChatOpenAI(temperature=0, model=model_name, openai_api_key=os.getenv('OPENAI_API_KEY'))
        self.llm = self.llm.with_structured_output(schema=Frame)
        self.chain = self.prompt | self.llm
        self.examples: list[BaseMessage] = convert_examples_to_messages()
        self.topics: str = str({i.name: i.value for i in TopicExplain})
```

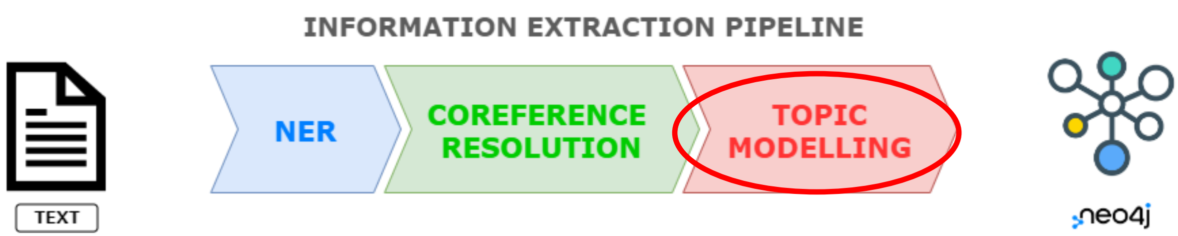
- **Return Format:** Frame instance
- **Examples**: Converted to LangChain messages
- **Chain**: Prompt + OpenAI *gpt-4o*
- **Input**: Topics and user_data, a Frame-converted pandas DataFrame

Containerization due to dependency issues



- **Generative LLM**: Request to docker container and OpenAI server

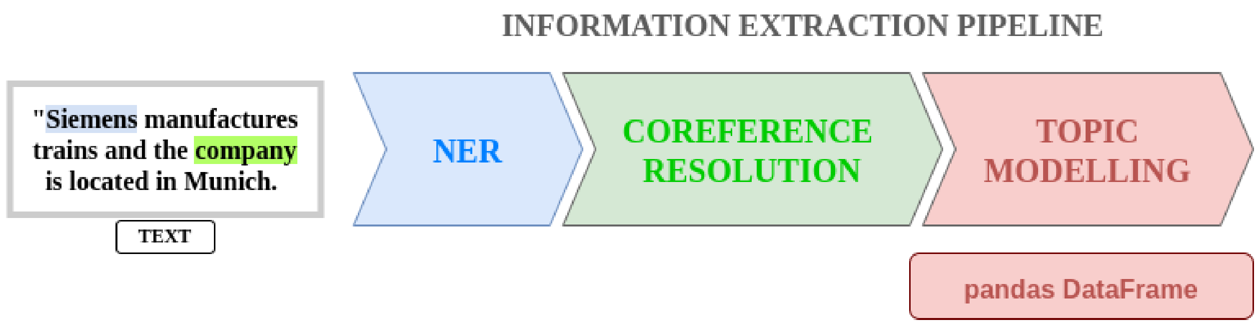
DEMO



- Traditional Topic Modelling: TF-IDF and BERTopic

3. Knowledge Graph

Prepare DataFrame for Knowledge Graph

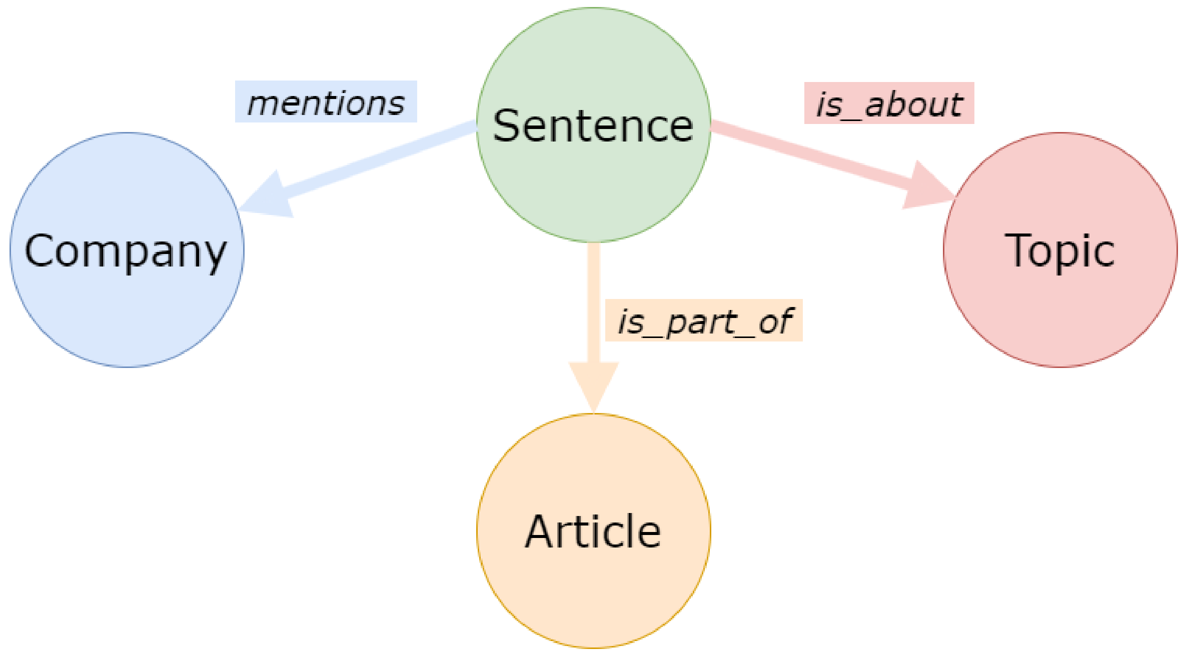


ner_coref	top_sent	topics
{{.......}}	*"Siemens manufactures trains and the company is located in Munich."*	topic6

```python
def prepare_df_for_kg(self, df: pd.DataFrame) -> pd.DataFrame:
    df['ner_coref_entities'] = df.ner_coref.str['entities']
    df = df.explode('ner_coref_entities').reset_index(drop=True)
    df['comp_symbol'] = df.ner_coref_entities.str['comp_symbol']
    df['comp_name'] = df.ner_coref_entities.str['comp_name']
    df['top_description'] = df.topic.apply(lambda x: TopicExplain[x].value)
    df.drop_duplicates(subset=['top_sent', 'comp_symbol', 'comp_name'], keep='last',
    ↪  inplace=True)
    df = self._drop_nans(df)
    df = self._get_isin(df)
    return df
```

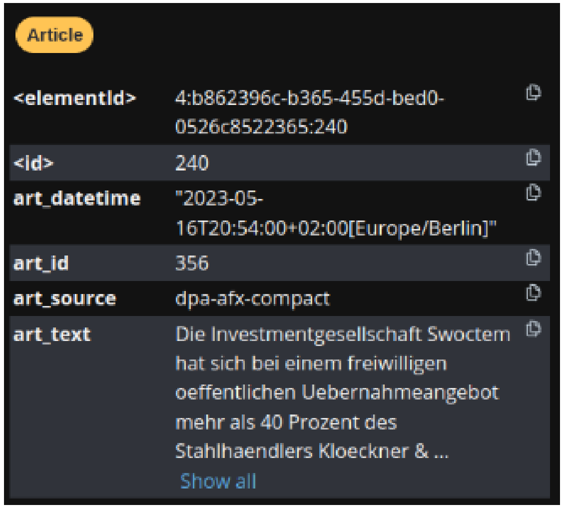
- Prepare pandas DataFrame for Knowledge Graph

Knowledge Graph

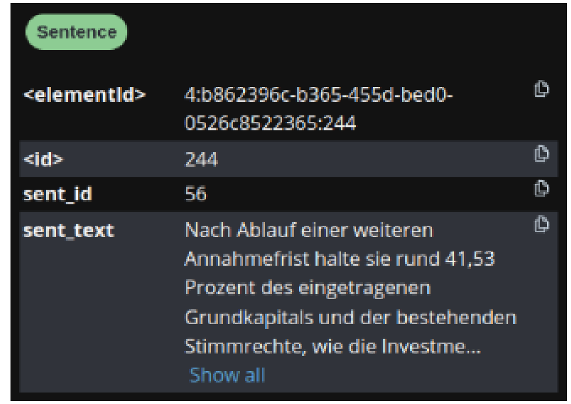


- **Neo4j Graph Database**
- **Nodes**: Sentence, Company, Article, Topic
- **Relationships**: mentions, is_part_of, is_about

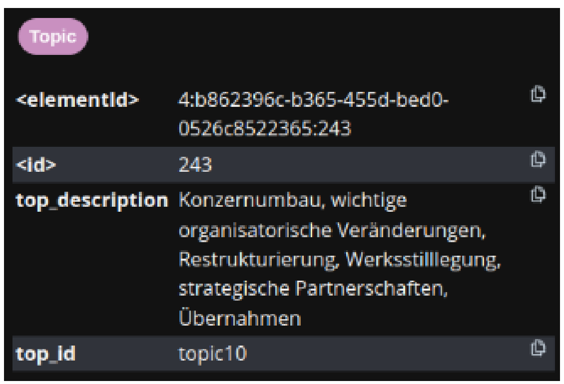
Load data



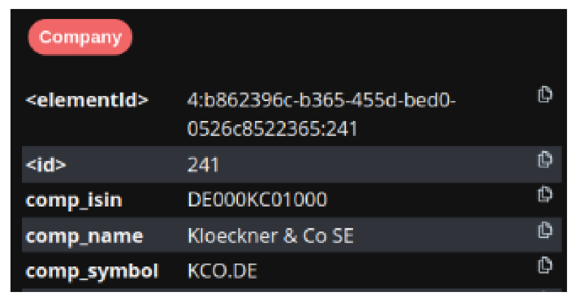
(a) Article



(b) Sentence



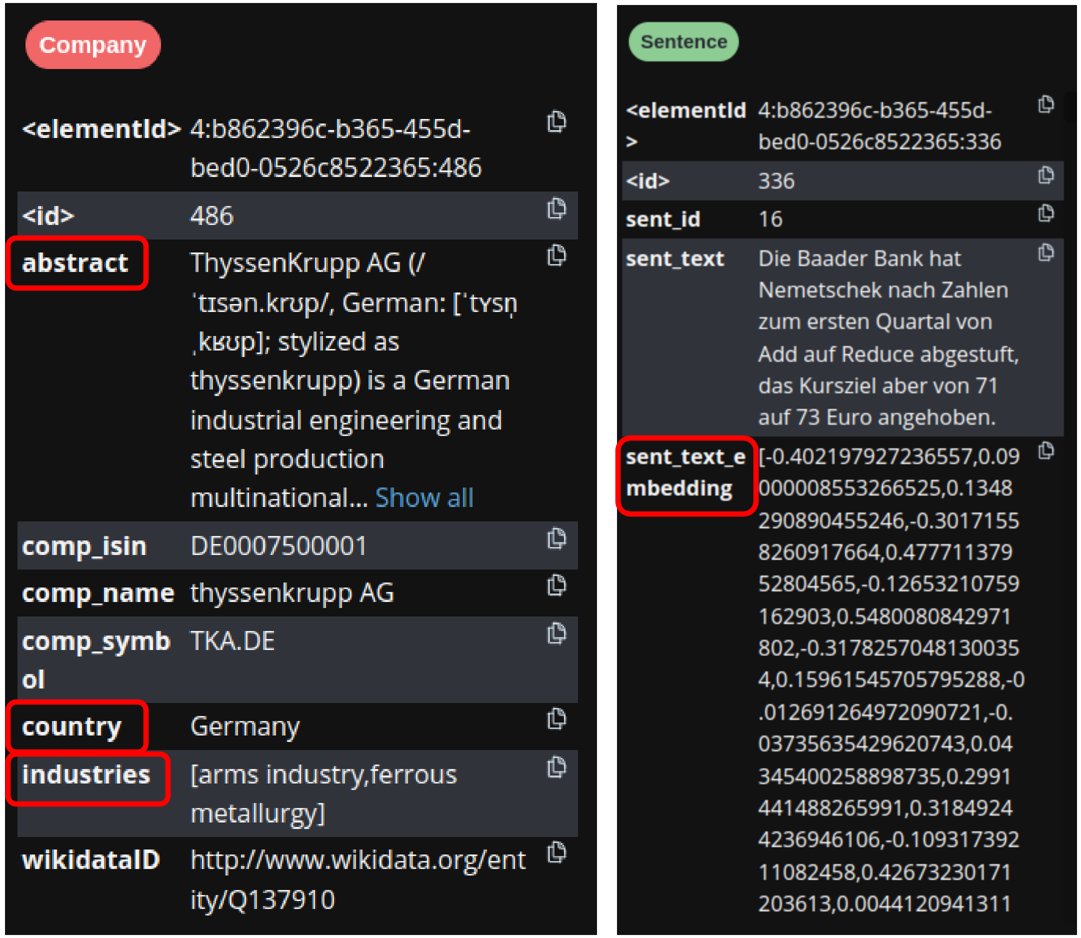
(c) Topic



(d) Company

- Data from pandas DataFrame

Enrich Knowledge Graph with external data



- SPARQL queries: Data from Wikidata, DBPedia
- Sentence Embeddings

Cypher queries can reveal complex relations

```
MATCH (s:Sentence)-[:is_part_of]->(a:Article)
    WITH s as sent, a as article, Date(a.art_datetime) as date
    MATCH (sent)-[:mentions]->(c:Company {comp_name: 'Brenntag SE'})
    WHERE date = Date({year: 2023, month: 5, day: 15})
    RETURN DISTINCT article.art_text
```

(a) Cypher Query 1: Articles about *Brenntag SE*

```
MATCH (a:Article)-[:is_part_of]-(s:Sentence)-[:is_about]->(t: Topic {top_id: 'topic12'})
    WITH s as sent, a as article, Date(a.art_datetime) as date
    MATCH (sent)-[:mentions]->(c:Company)
    WHERE date = Date({year: 2023, month: 5, day: 15})
    RETURN DISTINCT c.comp_name, sent
```

(b) Cypher Query 2: Companies, Sentences about *Topic12*

```
MATCH (a:Article)-[:is_part_of]-(s:Sentence)-[:mentions]->(c: Company)
    WITH s as sent, a as article, Date(a.art_datetime) as date
    MATCH (sent)-[:mentions]->(c:Company)
    WHERE date = Date({year: 2023, month: 5, day: 15}) and 'wholesale' in c.industries
    RETURN DISTINCT c.comp_name, sent
```

(a) Cypher Query 3: Sentences about Industry *Wholesale*

```
MATCH (a:Article)-[:is_part_of]-(s:Sentence)-[:mentions]->(c: Company)
    WITH s as sent, a as article, Date(a.art_datetime) as date
    MATCH (t:Topic)-[:is_about]-(sent)-[:mentions]->(c:Company)
    WHERE date = Date({year: 2023, month: 5, day: 15}) and
    c.country = 'Germany' and t.top_id = 'topic12'
    RETURN DISTINCT c.comp_name, sent
```

(b) Cypher Query 4: German Companies, Sentences about Topic12

Graph Bot

```
Question:
 Show me all the companies and the sentences they were mentioned of articles that were published between 2023-05-02 and 2023-05-03
```

Figure 7.11.: Graph Bot - Part 1: Question

```
> Entering new GraphCypherQAChain chain...
Generated Cypher:
MATCH (s:Sentence)-[:is_part_of]->(a:Article)
WITH s as sent, a as article, Date(a.art_datetime) as art_date
MATCH (sent)-[:mentions]->(c:Company)
WHERE Date({year: 2023, month: 5, day: 2}) <= art_date <= Date({year: 2023, month: 5, day: 3})
RETURN DISTINCT c.comp_name, sent.sent_text
Full Context:
[{'c.comp_name': 'LPKF Laser & Electronics AG', 'sent.sent_text': 'Unternehmen: LPKF Laser &'}, {'c.comp_name': 'LPKF Laser & Electronics A
```

Figure 7.12.: Graph Bot - Part 2: Creating Cypher Queries

```
Answer:
 LPKF Laser & Electronics AG was mentioned in the following sentences:
1. Unternehmen: LPKF Laser &
2. LPKF Laser & Electronics SE.
3. EQS-DD: LPKF Laser & Electronics SE:

Infineon Technologies AG was mentioned in the following sentences:
1. Infineon will 5 Milliarden Euro in die Erweiterung seiner Halbleiterproduktion in Dresden stecken und somit etwa 1000 
2. EU-Kommissionspraesidentin Ursula von der Leyen sieht im Ausbau des Dresdner Chip-Werks von Infineon eine grossartige 

Societe Generale Societe anonyme was mentioned in the following sentence:
1. Societe Generale hat Stabilus nach Quartalszahlen von Hold auf Buy hochgestuft und das Kursziel auf 65 Euro belassen.
```

Figure 7.13.: Graph Bot - Part 3: Answer

DEMO

INFORMATION EXTRACTION PIPELINE

4. Conclusion

- Process can convert unstructured data to structured information
- Knowledge Graph allows for fast and efficient retrieval of previously unstructured data
- Generative LLMs work well for information extraction
- Knowledge Graphs might be an alternative to traditional RAG