Master Thesis

Identifying Covid-19 Shortages with the Help of an Automatically Constructed Knowledge Graph

February 2022

Daphne Theodorakopoulos
M.Sc. Interaction Technology

Examination Committee:
Dr. Mariët Theune
Dr. Shenghui Wang
Dr. ing. Gwenn Englebienne
Prof. Dr. Louise Knight

UNIVERSITY OF TWENTE.

ACKNOWLEDGMENTS

This thesis was inspired by the research project "Managing Critical Supply Shortages" at the University of Twente. I would like to thank everyone involved in that project for giving me a thesis topic: Louise Knight, Esmee Peters, Shenghui Wang, and Gwenn Englebienne. Thank you, Shenghui and Gwenn, for supervising me. I learned a lot and I had a great time working with you. Although our meetings were sometimes a bit frustrating, I always felt energized after them to continue working on a new approach. I appreciate the time and effort you put in to help me complete this thesis. I think that anyone can be happy to have you as supervisors.

I would also like to thank Mariët Theune and Louise Knight for being part of my thesis committee and giving me feedback.

ABSTRACT

Within the Covid-19 pandemic there were severe product shortages in supply chains, e.g. face masks. Early detection of them diminishes their consequences. The aim of this study is to automatically **identify Covid-19 shortages from text supported by a Knowledge Graph (KG)**. The Covid-19 Open Research Dataset (CORD-19) of Covid-19 research publications forms that basis.
The **method** can be split into three main parts:

1. An **ensemble of term weighting schemes over time** was used to identify shortages in text. Those are: monthly term frequencies, monthly TF-IDF, word embeddings, the monthly co-occurrences of certain keywords, and how that changes.

2. **Topic Modeling to select relevant articles** was applied. One topic in a guided LDA model was seeded with keywords. All articles which are part of the seeded topic were selected.

3. A **domain-specific KG was automatically created** from text to improve the identification of shortages. A sub-graph was extracted from DBpedia based on keywords, which was enhanced with open relation extraction from the Topic Modeling (TM)-selected articles. The KG was completed with entity types, super-classes, and text cleaning. Link prediction and neighbor occurrences within the KG were added to the ensemble to identify shortages.

The **shortage identification was somewhat successful**, as around half of the expected terms were retrieved but the list also contained many irrelevant terms. The best weighting schemes were: similar terms from the word embedding, and the KG neighbor occurrences, which is a new scheme.
The **TM selection** of relevant articles **outperformed the standard keyword-selection**. However, the shortage identification on all data was better than on the selected articles, which questions the method. That is predominated by the advantage of saving human effort.
The **KG is domain-related but noisy**, as it contains 70% of the expected entities but also a lot of irrelevant and meaningless data. The shortage identification on the TM-selected articles considering only KG entities was slightly better than considering all terms. However, that did not perform better than the method applied to *all data* without the KG. Most likely, that is due to the topic model not selecting the articles well enough.
An important limitation is that **the ground truth list of shortages is incomplete**. Therefore, the precision of the shortage identification and the KG domain affiliation is underestimated.
In future work, additional KG completion methods, such as entity resolution, fact-checking, and error detection should be applied. Furthermore, a human evaluation of the suggested shortages, the selected articles, and the KG should be done.
We conclude that **the suggested method is a valid approach towards a shortage-identification system but there are still many open challenges to overcome**. The **main contributions** include a shortage-identification method, an automated method to select relevant articles, a method to automatically construct a KG from text, and the resulting Covid-19 KG of product shortages.

CONTENTS

Abbreviations

CORD-19 Covid-19 Open Research Dataset.

Covid-19 coronavirus disease 2019.

KG Knowledge Graph.

LDA Latent Dirichlet Allocation.

NLP Natural Language Processing.

PPE personal protective equipment.

RE Relation Extraction.

SC Supply Chain.

TF-IDF Term Frequency-Inverse Document Frequency.

TM Topic Modeling.

TWS Term Weighting Schemes.

1 INTRODUCTION

1.1 Problem Statement and Motivation

No toilet paper, no pasta, sewing your own face masks, is it the apocalypse? No, it is spring 2020 and the world is facing severe shortages of essential products. The **coronavirus disease 2019 (Covid-19)** spread rapidly around the world, quickly becoming a global pandemic. Predictions of another coronavirus disease have been made since the SARS epidemic in 2002 [1, 2]. Nonetheless, the outbreak of Covid-19 hit companies and governments by surprise [3]. The non-preparedness coupled with the sudden need for certain products caused **severe shortages in Supply Chains (SCs)** for such products [3]. As these products were largely personal protective equipment (PPE), medical products, and vaccines to prevent infections, these shortages had fatal consequences and contributed to the unrestrained spread of the pandemic [3].
According to Morse [4] and others, emerging diseases are more likely now than ever. This is due to several socio-economic, environmental, and ecological factors, such as population growth, global warming, and increased global travel [5, 6]. Therefore, it is important to be **prepared for the next pandemic**. **Knowing about upcoming shortages helps to mitigate them** and reduce their impact, for instance, by increasing the production of substitutes, maintaining stockpiles, and creating a more robust SC [7]. Additionally, scarcities lead to products being more expensive, which means that fewer people can afford them. In the Covid-19 pandemic, at one point prices of surgical masks were six times higher than normal, according to the World Health Organization. There were market manipulations and fraud due to the high demand which could not be met [3].
For these reasons, in future pandemics, an **early warning system for emerging scarcities** could be a great help. Not only the anticipation of a shortage but even the identification of a current one is helpful because as soon as a shortage is known, measures against it can be taken. Moreover, other crises experience shortages [7] as well, so we can imagine using such a system in any kind of crisis. The **retrospective analysis of the shortages within Covid-19** can give valuable insights into the vulnerabilities of the global SC and precautions for possible future pandemics can be taken. Understanding how a shortage occurs can also give early signals to predict potential product shortages. This is a need for a global early warning system of shortages [7], this study contributes to creating it.

1.2 Proposed Method

The aim of this research is to **create a method to identify shortages in SCs based on the Covid-19 pandemic**. A large dataset of Covid-19 articles is its basis. The proposed method consists of three main stages. An **ensemble of Term Weighting Schemes (TWS)** weights terms over time where the high-scoring terms are the suggested shortages. A **domain-specific Knowledge Graph (KG) will be created** and used to enhance the performance of the TWS to identify potential shortages. To achieve domain-specificity of the KG is actually within that domain, **Topic Modeling (TM) will select domain-related articles**.

Text analysis for the SC domain. Several studies in the SC domain emphasize the use of technology, e.g. [8, 9]. The work of Bansal et al. [10] highlights the use of text data and TM to study SCs. There is a lack of studies in SC management that actually use text data. Furthermore, studies analyzed SC shortages in Covid-19, e.g. Ivanov [11]. The approaches used are human analysis, e.g. [12], simulation-based, e.g. [11], or basic Data Science methods, see for example [13]. There is a **lack of work using advanced methods analyzing large amounts of textual data in the SC domain**.

Dataset and Topic Modeling. Within Covid-19, possible sources of data include scientific publications. Even though these include much irrelevant information, they also include details about SC disruptions and resulting shortages. Exactly when a shortage occurred is difficult to determine because it does not appear simultaneously everywhere in the world. This study uses the **Covid-19 Open Research Dataset (CORD-19)** [14] of scientific publications. Furthermore, the dataset is quite large and regularly updated. The CORD-19 dataset is mostly biomedical. Therefore, many articles are not relevant for this use case. Conventionally, SC experts use keywords to find relevant articles. This research proposes to **use TM** instead, in line with Bansal et al. [10]. This would not only save a lot of human effort but also **create a domain-specific dataset** by only selecting articles of a certain topic. The reduced dataset can help the construction of the KG.

Term Weighting Schemes over time. In a preliminary analysis, the combination of **different TWS over time found some items in shortage**. However, the results were still noisy, mostly because the statistical Natural Language Processing (NLP) approaches did not consider the meaning of the text. For instance, just like *"face masks"*, *"patients"* is an especially frequent term in the Covid-related corpus. The first one is a product in shortage, the second one not. A human being would be able to distinguish those terms from the relevant ones. Humans have semantic knowledge about language that statistical methods do not consider. Therefore, a more sophisticated method needs to be used. **KGs are an attempt to model real-world knowledge to distinguish word types and find physical objects** in a concise way for a computer to understand.

Knowledge Graphs. KGs represent knowledge as relationships (e.g. "is_a") between entities (e.g. "face_mask") in the form of triples (e.g. "face_mask" $\rightarrow$ "is_a" $\rightarrow$ "PPE"). KGs can be human-made or built up automatically from large sources of text. Often, they are about a certain domain. Automatic reasoning can be applied to them to derive new knowledge [15, p. 4], [16]. The most well-known KG is probably the one enhancing Google's search engine [17]. They aim to connect all the knowledge available on the web, based on user searches. The extracted information is displayed in knowledge panels on top of the search results. There are large existing KGs that model general knowledge and also some specific to the Covid-19 pandemic. This thesis proposes to **automatically create a KG to improve the identification of Covid-19 shortages**. It is based on Covid-19 research papers and domain-specific to shortages in SCs which is achieved with TM and a list of keywords. With the help of a KG, only terms should be retrieved which are products that can be in shortage. The challenge of the shortage-identification system and the KG to be built is that the **shortages are unknown during construction**. Thus, the methods are unsupervised. To measure performance, a hand-made ground truth of known shortages has been created post-hoc. It is split into **shortage terms, which refer to products that were known to be in shortage** and their suppliers, e.g. "face mask" or "vaccine", and **shortage indicators, which are terms that indicate a shortage**, e.g. "scarcity" or "need".
Furthermore, **the KG can be leveraged in other ways**. The SC domain is just an example. Given a set of keywords, with some slight adjustments, the proposed method to build a KG

should also work for other domains. Moreover, using this KG to identify shortages is just one use case, there could be other applications for the created KG. It can also be seen as an attempt to summarize and connect all the knowledge about Covid-19 shortages. There have been many Covid-19 publications of different disciplines, too many for a human to reasonably comprehend. A KG is an explainable method to process all this data on a large scale.

1.3 Research Questions

The goal of this thesis is to automatically identify potential Covid-19 shortages from data with the help of TWS and a domain-specific KG. This leads to the following research questions:

1. **How to automatically find unknown potential Covid-19 related shortages in supply chains in text?**
    - **RQ1**: *How well can statistical NLP methods alone identify shortages in text?*
    - **RQ2**: *Does Topic Modeling improve the selection of relevant articles in comparison to keyword-based search? What is the difference in performance when applying the shortage-identification method to the reduced dataset?*
    - **RQ3**: *How well can a domain-specific Knowledge Graph be constructed automatically without knowing the shortages? Does such a Knowledge Graph improve the detection of potential shortages?*

The research questions ask for three main parts. The first part is to identify Covid-19 shortages using TWS. The second part is about narrowing down the dataset to be more targeted towards the domain in question using TM. The third part is the creation of a KG to improve the performance of the TWS by including semantic knowledge in the statistical methods.

1.4 Overview of the Thesis

The outline of this thesis is as follows. The first part gives some background on TM and the construction of KGs. Then, related work about all three parts is discussed. Figure 1.1 shows an overview of the methodology and experiments to answer the research questions. It is split into four main parts. Each part corresponds to one chapter including the methodology and experiments to answer the respective research question. An exception is the first box, as it is only the preparation of the data. The methodological chapters each include a more detailed overview Figure at the beginning of the chapter. Finally, the results are critically discussed and future directions are proposed.

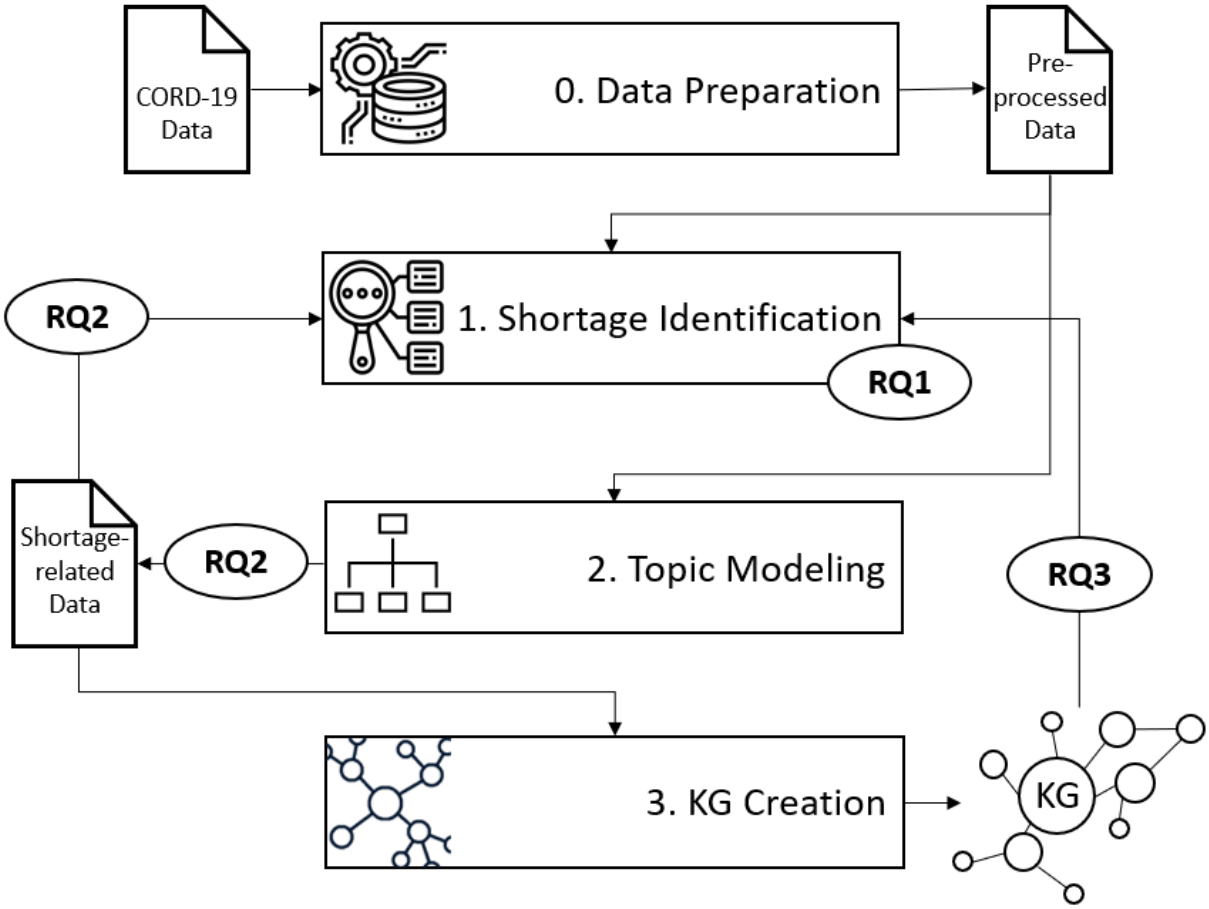


Figure 1.1: Summary of the methodology and experiments overview

0. Data Preparation (Chapter 4)

That includes the introduction of the dataset and the preprocessing of it. It mentions some simplifying assumptions that were taken in this study. Moreover, it will explain how the ground truth list of known Covid-19 shortages in SCs was created. It will also describe how the evaluation with this ground truth list was done. This chapter ends with a first analysis of the data to see if the dataset is well suited to find shortages.

1. Shortage Identification (Chapter 5)

In the corresponding chapter, statistical NLP approaches will be introduced to find shortages in text. These include term weighting schemes over time and word embeddings. In the experiments, the top terms retrieved by the methods are combined and the accuracy is measured based on the ground truth shortage list and process time. The experiments will answer RQ1.

2. Topic Modeling (Chapter 6)

Here, it will be explained how TM is used to reduce the dataset by selecting only articles related to shortages in SCs. In the Figure, the reduced dataset is called “shortage-related dataset”. First, the algorithm and some alternatives are described. Second, the TM is tuned. Third, the TM selection of the articles is compared to a keyword-based baseline. Finally, the shortage-identification method is repeated and compared to the previous results. That will answer the second research question.

3. KG Creation (Chapter 7)

The statistical methods predicted some terms which cannot be a product in shortage and also missed some relevant ones. Therefore, semantics are included based on the creation of a domain-specific KG. The KG building process includes the creation of an initial KG from DBpedia, information extraction from the reduced data, and KG refinement. To answer the last research question, the KG is evaluated in two ways: intrinsically and by using it within the shortage-identification method. The existing method is enhanced by the knowledge of the graph and some KG prediction methods, including link prediction. In the end, the performance is compared to the results of the previous chapters and the final results are closer analyzed.

2 BACKGROUND

This chapter gives some background information as a foundation for the rest of the thesis. First, some evaluation measures are introduced. Second, Topic Modeling will be explained. The remainder of the chapter is about Knowledge Graph creation, refinement, and evaluation.

2.1 Evaluation Measures

Most of the methods in this work are evaluated with precision, recall, and F-score. Given a list of elements retrieved by a method, e.g. predicted terms, and a list of relevant elements, e.g. the terms that are looked for, these measures can be used to score the accuracy of the retrieval method. A perfect score would be if all relevant elements are retrieved. **Precision measures how many items are relevant out of the retrieved ones.** A high precision means that the method is very good at only returning elements that are looked for. **Recall measures how many items were retrieved out of the ones which are relevant**. A high recall means that the coverage of the elements that were looked for is very high. Often precision and recall behave anti-proportionally to each other. **F-score combines the two measures**. In this thesis, the measures score, for example, the list of terms suggested as shortages with a list of known shortages. A weight β can be included in the F-score, weighing precision and recall differently. That is called F_β. In this work, the simplest version with a β of 1, called F_1 will be used. The equations are the following:

$$\text{Precision} = \frac{\text{number of retrieved and relevant elements (true positives)}}{\text{number of retrieved elements}}$$

$$\text{Recall} = \frac{\text{number of retrieved and relevant elements (true positives)}}{\text{number of relevant elements}}$$

$$F_1 - \text{score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

In case there are no retrieved elements or the precision or recall are 0, the respective F-score is set to 0.

2.2 Topic Modeling

The manual efforts made by procurement researchers to find articles related to shortages could be automated by the use of TM. TM is an unsupervised machine learning technique that can discover topics in a document corpus based on the semantic relatedness of the terms within the documents. The model can subsequently be used to classify each document to a topic. In the following, the TM method used in this thesis, namely Latent Dirichlet Allocation (LDA), will be revised, as well as how to interpret the found topics.

2.2.1 Latent Dirichlet Allocation

In LDA, the documents are seen as if they were generated from a mixture of hidden topics where each topic is regarded as a probability distribution over words [18]. LDA randomly assigns a topic to each word in a document and then corrects itself a given number of iterations per document. It does that by calculating the probability of a word being generated given a document and a topic. Subsequently, it reassigns topics to the words based on that probability. The distribution of these probabilities is called the posterior distribution of latent variables. The goal of LDA is to determine it. Most commonly, the approximation methods Variational Bayes or Gibbs sampling are used for that. [19, 20, 18]

2.2.2 Variations of LDA

There are several extensions of TM. One approach is **dynamic TM** which considers time. To retrieve the evolution of topics, the data is split into time slices. A model is created for each slice and the developed topics are based on the previous time slice [21]. The biggest drawback of this approach is that time is viewed on a discrete scale while topics do not necessarily change discretely. There are several dynamic TM works considering the temporal patterns on a continuous scale [18].
There is a variation that guides the LDA by giving seed words for topics. That is called **Guided LDA** or Seeded LDA and was first introduced by Jagarlamudi et al. [22] and Mukherjee and Liu [23]. The topics are then built around those seed words. This is especially interesting if the topics are already known beforehand. In this case, it is known that one topic should contain shortages, so shortage-related terms could be used as seed words.

2.2.3 TM Parameters

There are two main challenges when building a topic model. One is finding the right number of topics k. The other one is overcoming the instability of the algorithm.

Selecting the right number of topics k

TM usually requires a parameter k which corresponds to the number of topics the model should return. There are several methods that try to determine the optimal k for a given dataset externally. However, according to Vayansky and Kumar [18], these methods have the assumption that the optimal k is a given value.
Other approaches do not try to externally determine k but they iterate over different ks and apply TM on a subset of the data. [18] On each iteration, measures such as perplexity indicating how well the model describes the data are calculated. The k with the lowest value is considered to be the best fit [24]. Another measure is the coherence of the topic. There are several ways to calculate it which will be discussed in section 2.2.4.

Stability

One challenge of TM is the instability of the algorithm. Every time that it is applied to the data, the topics discovered will vary slightly [19]. One way to make the algorithm more stable is to use ensemble methods, while wro [25] suggest doing hyperparameter tuning. They investigated the topic instability of LDA with its default settings and found that the default parameters of LDA implementations do not necessarily yield good results while the topics are often unstable. They recommend tuning the most important hyperparameters, namely the number of topics, called **k**, the prior of the document topic distribution, called α, and the prior of the topic word distribution, called β [25].

2.2.4 Topic Interpretation

The topic model returns words within the topics. Moreover, it can be used to classify documents into topics. How these topics are understood is always a subject of human interpretation. To support that, topic coherence can be used. Moreover, visualization techniques can be helpful.
Topic coherence is one way to quantify the quality of the topic model. The coherence score has a high correlation with human judgment and is based on the co-occurrence of words of the same topic [26, 27]. It gives an indication of how well the topics are coherent in themselves measured by the semantic similarity of the top terms in the topic. There are some commonly used coherence scores that are often offered by TM libraries:

- UCI coherence [28]
- $NPMI$ coherence [29]
- $U - Mass$ coherence [27]
- C_v coherence [30]

According to Röder et al. [30], the C_v coherence is the best performing one and will therefore be used in this thesis.

2.3 Knowledge Graphs

KG definition. KGs have been defined as "very large semantic nets that integrate various and heterogeneous information sources to represent knowledge about certain domains of discourse." [15, p. 6]. Usually, the nodes within the graph are called entities, which have certain attributes and are connected to each other by relations. A typical KG representation stores relations as triples of an entity, a relation, and another entity. In particular, the entities can also have a specific type that can be modeled by a relation, such as "is_a" [31]. The first entity of the triple is also called head or subject and the second one tail or object.
In this study, a KG will be automatically constructed from text. In the following, some background will be given on how a KG is commonly constructed including some details on the steps.

2.3.1 Knowledge Graph Construction

Figure 2.1 gives a structured overview of the current research in the KG domain. In this work, the main focus is on *Knowledge Acquisition*, i.e. the construction of the KG. *Knowledge Representation Learning* will be used. Finally, identifying shortages in text with the KG will be a *Knowledge-Aware Application*, more specifically a Natural Language Understanding application. Covering the last domain of the figure, in future work, the KG can be converted to a *Temporal Knowledge Graph* because shortages are time-dependent.

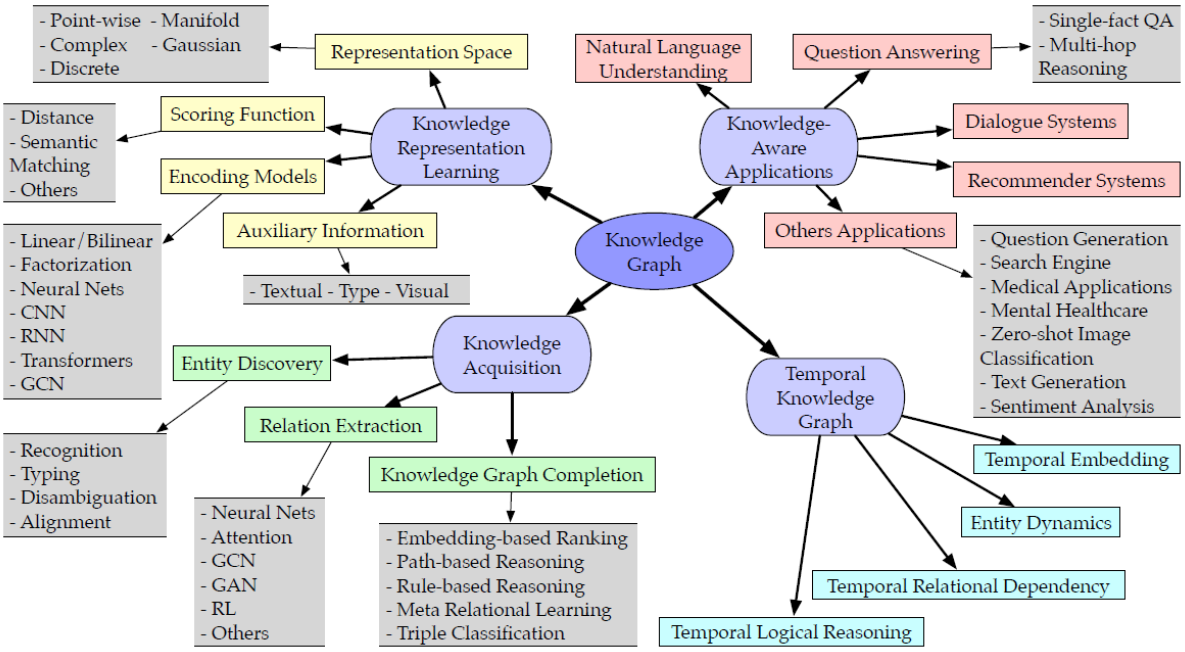


Figure 2.1: Categorization of research on knowledge graphs from Ji et al. [32]

Most sources structure the **generation of a KG** in a similar way (cp. Figure 2.1: *Knowledge Acquisition*). The steps of it are listed in the following.

1. **Knowledge creation** or construction. This includes:
   - data retrieval and preprocessing,
   - entity discovery,
   - relation extraction,
   - entity linking to existing KGs or an initial KG,
   - and evaluation [15, 33, 34].
2. **KG refinement** or curation (*KG Completion*) using KG embeddings among other techniques (*Knowledge Representation Learning*). That consists of:
   - error detection in the graph,
   - KG completion,
   - and sometimes assessment of the graph [31, 15].

Finally, the KG is used in a *Knowledge-Aware application*.

2.3.2 Knowledge Creation

The data is first preprocessed using standard NLP methods. The remaining steps will be explained in the following. This includes entity labeling, relation extraction, and entity linking.

Entity Labeling

A named entity is a word form that is called “an atomic element or member of the semantic class which may vary depending upon the domain of interest.” [35]. An example of that is the “University of Twente” as an *organization*. The classes that a named entity can belong to can

vary. There are some standard ones, such as *location, person, or time*. Depending on the use case, there can also be other classes, for instance, *product*. **Named Entity Recognition (NER)** aims to tag these named entities in text and is a sub-task of information extraction. The named entities are classified as a type, such as “person” or “organization”. Some extensive reviews were written by Goyal et al. [35] and Al-Moslmi et al. [36].
NER models take the raw text as input and identify the entities as part of the labeling process. That results in a small number of entities that are mostly labeled accurately. However, there is no control over which terms will be recognized as an entity. A related task is **entity typing**. Entity typing predicts the types of an entity in a sentence [37]. Whereas NER first recognizes the entity in text and then classifies it to a type, entity typing only classifies the entity with a certain probability. Thus, any term can be typed, with the downside that the labels are less accurate. To reduce the classification error, a threshold should be set on the probability above which the types are considered as correct. Since more entities are labeled with entity typing and there is more control over it, **entity typing will be used in this thesis.**
Goyal et al. [35] list some **challenges of entity detection**. There can be nested entities, where an entity within an entity needs to be recognized. For example, the *organization* “University of Twente” contains the nested entity “Twente” which is a *location*. In this thesis, partial matching of entities and text fragments is used to tackle this problem. Another big problem is ambiguity in text, e.g. a “Hamburger” can refer to a *person* from Hamburg or to *food*. Kejriwal [38] name some more challenges specific to construction of a domain-specific KG. Many methods are supervised and need labeled training data. In practice, however, the entities to be discovered are typically not known beforehand, so the supervised methods cannot be used. Moreover, unsupervised techniques that have been optimized on generic data tend to not work well on domain-specific data. Another challenge can be the heterogeneity of the data from different sources [38, p.10-11]. The methods used in this thesis are all unsupervised because no labeled training data is given. To address the problem of having no types, the roots of noun phrases are extracted to form super-classes.
According to Al-Moslmi et al. [36], there is no standard for the **evaluation** of entity discovery for KGs. They name four important aspects for the evaluation: recognizing entity mentions, assigning a type to mentions, identifying the entity which is meant by a mention, and linking mentions in a KG [36]. Precision, recall, and F-score are mainly used for evaluation if a ground truth is given.

Relation Extraction

“Relation extraction is the task of detecting and classifying predefined relationships between entities identified in text.” [39]. The aim of Relation Extraction (RE) is to find tuples of entities connected by a relation.
There are **relation prediction** and Open Relation Extraction (RE) methods. The relation prediction methods were first. They classify entity tuples to certain preset relationships. This is often done with a pre-trained model. The advantage of this is that there is only a small set of possible relations. However, it might generalize too much. That means that every possible relation that two entities can have needs to fit into one of the predefined categories of the model. Language is usually more complex than that and often the predefined categories do not match what is meant in text. Furthermore, this often requires labeled training data which is quite resource-intensive.
Open Information Extraction or **open RE** was first introduced by Etzioni et al. [40] and freely extracts the relation between two entities. The open RE methods are more detailed and closer to the source sentence. Moreover, they are more flexible as they do not need prior domain knowledge, but the relations can become too specific and cannot be grouped easily. In this study, **open RE was chosen**, to cover as many possible relations as possible.

There are two main forms of **RE evaluation**: prediction of missing links on held-out data and manual evaluation. The held-out method only works for relation prediction and not for open RE because a relation is not predicted but is extracted from text freely. Thus, the input is not a pair of entities but raw text. The manual evaluation is quite accurate but it can only be done on a small dataset, it is resource-intensive and it could be subjective. In addition, the evaluation can be set up to focus on the most frequent relations [41]. In this thesis, frequency analysis is done because it is the easier approach of the two possible ones. In future work, human evaluation should be considered.

Entity Linking

Entity linking matches the entities found in text to the Internationalized Resource Identifier (IRI) of their corresponding entities in a **large existing KG** [42]. According to Al-Moslmi et al. [36], **entity linking is composed of three tasks**:

1. Candidate entity generation extracts all possible entities from the KG which might be referring to a found entity.
2. The candidate entities are ranked and the one with the highest rank is returned.
3. NIL clustering, where NILs are entities that are not in the existing KG.

Shen et al. [42] provide a good overview of the existing methods for entity linking. Entity linking has its limits, as not all entities exist in large KGs and terms cannot always be resolved to being the same entities. In the same way as the entities are linked to existing KGs, they can also be **linked to an initial KG** [34].

2.3.3 Knowledge Refinement

KGs often contain errors because of missing or inaccurate knowledge, which leads to weaker performance when used [43]. KG refinement methods are used to tackle that problem. Knowledge refinement consists of two main parts, error detection, and knowledge completion [15, 31]. In this thesis, only knowledge completion methods will be used.

KG Reasoning

One method to refine KGs is KG reasoning [31]. Chen et al. [44] define reasoning as simulated thinking to deduce conclusions from existing knowledge. Reasoning over KGs tries to complete a KG by using machine learning to find erroneous knowledge and new relations between entities based on existing knowledge. [44]
According to Bellomarini et al. [43], KG reasoning can be split into three dimensions, reasoning for knowledge integration, reasoning for knowledge discovery, and reasoning for application services. **Reasoning for knowledge integration** is the task of integrating knowledge from various sources. When integrating multiple sources of knowledge from different KGs, duplicate entities can occur, thus, entity resolution is necessary. **Reasoning for knowledge discovery** aims to find new or hidden knowledge within a KG. This can also be used for knowledge completion tasks. **Reasoning for application services** is the reasoning over a KG to provide a certain service, such as question answering or recommendation systems. [43]
Ji et al. [32] categorize the different types of reasoning into embedding-based reasoning, path-based reasoning, and rule-based reasoning. KG embedding is one of the most popular reasoning methods used for KG completion or knowledge discovery [43]. The representation learning methods described in section 2.3.3 are mostly used for it. In this thesis, only embedding-based reasoning is applied.

Knowledge Graph Embedding

Many refinement tasks make use of KG embeddings [45, 32]. For example, deduplication or link prediction, like in this thesis. A KG embedding is another representation of a KG granting access to the underlying information of triples [46]. The triples in a KG are embedded into a fixed-length vector of a lower dimensionality to simplify the usage of the graph without losing its structure [46, 45]. The KG embedding is often evaluated with link prediction in the same way as the task itself is evaluated [47].

Embedding Structure. Wang et al. [45] surveyed KG embedding approaches. An embedding method is usually split into three parts: representation space, scoring function, and encoding models. Most methods use vector, matrix, or tensor representation where entities are vectors and relations are operations in that vector space. A scoring function is defined which calculates the plausibility for each fact. The representation learning learns the embedding using encoding models so as to maximize the overall plausibility of the facts measured with the scoring function. In addition, auxiliary information, such as time, can be included in the embedding. [45, 32]

Encoding Models. There are three groups of embedding techniques: Translational and Rotational Based Models, Semantic Matching Models, and Neural Network (NN)-Based Models [43, 45]. Translational and Rotational Based Models use distance-based scoring and loss functions. Semantic Matching Models uses similarity-based scoring functions. NN-based models include many Deep Learning models which were introduced in recent years, see Ji et al. [32].

Knowledge Refinement Tasks

The KG refinement methods, including reasoning, can be used for KG enrichment, i.e. error detection and completion. According to Paulheim [31], Knowledge refinement has three dimensions: 1. completion vs. error detection, 2. the refinement target, for example, entities or relations, and, 3. the data which can be internal (only within the KG) or external. [31]
The automatic construction of a KG can lead to it containing some noise, therefore, **error detection** is performed [46]. The goal of error detection is to improve the correctness of the developed KG by first detecting errors and subsequently correcting them [15, p.46]. **Knowledge completion** tries to find duplicates and conflicting types and aims to resolve them to increase coverage of the KG [15, 31]. This is done by adding and deleting type and relation statements [15, 31]. In the following, different methods will be described. They will be for either one of error detection and completion or both, as it is difficult to distinguish the two completely and the methods are often the same. The methods are complementing each other, thus they could all be used. In this study, only link prediction and deduplication will be used.

Link and Relation prediction (refinement target: entity or relation). Link prediction aims to predict an entity for a known second entity and a known relation, so either the head or the tail of a triple is to be predicted. For relation prediction, both entities are known and the relation should be predicted. Using the learned embeddings, the scores for each possible triple containing the known two parts can simply be ranked and the best one will be selected. To evaluate the ranked list of candidate entities, mean rank or Hits@n can be used on some test triples where one entity or relation is removed beforehand. [45, 38]

Triple classification (refinement target: entity and relation). Triple classification decides for an unseen triple if it is valid or not. This can also be called fact-checking. With the help of the embedding, the score of the new triple can be calculated, the new fact is either considered

true or false. [45, 38] There are other methods that perform fact-checking without embeddings. For instance, Shi and Weninger [48] see it as a link prediction problem and use path-based rule mining to evaluate triples. To evaluate triple classification, standard classification metrics or ranking metrics like mean average precision can be applied [45].

Deduplication (refinement target: entity and triple). Deduplication tries to find out if two entities or triples are equal to each other. For entities, this is also called entity resolution. Nickel et al. [49] proposed to calculate the similarity between the vector embeddings of the two entities. To evaluate that strategy, the area under the curve is often used [45]. There are many other tools not based on embeddings. For example, the Python library Dedupe [50] uses similar techniques in multiple steps, avoiding embeddings, and is also used in this thesis.

External Methods. The aforementioned methods are all internal because they only consider the data from which the KG is built. All of the tasks are solved using automatic reasoning techniques, which only consider the underlying structure of the information and reason based on that. This could mean that something which is logical to the system does not make sense in the real world. This is true for internal methods, there are also external ones. Often, a classifier is trained on large KGs, like DBpedia to predict relations [31]. Moreover, similar methods as for relation extraction can be used. In this work, that approach was not pursued.

2.3.4 Evaluation of KGs

The KG assessment evaluates the quality of the constructed KG. The error detection and completion methods already give an indication of how well the KG was constructed. Paulheim [31] categorized the KG evaluation methods in two broad categories, methods that only use the KG and methods which include external data.
One evaluation method is **manual evaluation as partial gold standard**. That is, either an external large KG is used as ground truth, or an extracted sub-graph is manually labeled. The manual labeling means, for error detection, that the entities and relations are marked as correct or incorrect. For KG completion, the axioms that should exist in the KG are gathered. Both refinement tasks are evaluated with precision, recall, and F-score. Human evaluation gives of course good quality results but is often quite resource intensive. Therefore, large existing KGs, like NELL or YAGO, are sometimes used for evaluation. The constructed KG is then interlinked to the large existing KG. However, there can be errors in the interlinking process and the gold standard KG can also have errors. [31] The evaluation methods do not evaluate the entire KG. This can be problematic, as the sources for the KG are not always of the same quality.
Paulheim [31] names **the KG itself as silver standard**. For example, link prediction can be done to validate how well the KG completion methods can replicate the KG. Another approach is to let the KG embedding score triples in the KG. Either the entire KG can be used which might lead to overfitting or the graph can be split into training and test set. That is not easily done because often the entities need to be present in the test and train set in order to be scored. This method heavily suffers from the KG not necessarily being correct, and the method also not testing for correctness. Thus, the results of this method should be considered with caution, as they might be biased. Furthermore, they depend on the quality of KG embedding. In this thesis, the triples in the KG are scored and evaluated using k-fold cross-validation.

2.4 Summary of the Chapter

This chapter gave some background relevant to this research. First, the F-score for evaluation was introduced. Second, Topic Modeling was explained. The most used method is LDA and hyperparameters need to be tuned for a more stable algorithm. After that, the basics for KG construction were explained. Namely, entity linking, entity, and relation extraction as well as knowledge refinement. The knowledge refinement section talked about different reasoning techniques for KGs and gave an overview of KG embeddings. Additionally, some refinement tasks in order to detect errors in the KG and complete the KG were discussed. Often embedding-based reasoning is used in these tasks which aims to predict missing or wrong information.

3 RELATED WORK

The following will first summarize why shortages in SCs occur. Then, it will describe current methods to anticipate SC disruptions leading to shortages. Some of them are technology-driven and some even use NLP approaches. After that, some related work about TM will be given. Finally, KG work will be discussed. That includes the review of large scale KGs and Covid-related KGs and some knowledge aware applications.

3.1 Shortages in Supply Chains

The shortages in the Covid pandemic aggravated due to the non-global management of them. The **management of shortages** includes national procedures to report possible future shortages to authorities which can then react, for example, with substitutes. Moreover, there are information systems in procurement and logistics that can detect shortages. For the future, they **suggest** international collaboration, a uniform definition of shortages, identifying medicines at risk to be a shortage, and developing a **global early warning shortage notification system** [7]. The aim of this thesis is to attempt a system like that.
Shortages in SCs occur for different reasons. When trying to anticipate shortages it can be useful to look at those reasons as they might indicate an upcoming shortage. Sodhi et al. [3] researched the shortages of essential goods in the US during the pandemic, identified their causes, and proposed a research agenda for responsive SCs. The shortages of essential goods in the US that they analyzed are “Household paper products and disinfectant wipes”, face masks, PPE, and Ventilators. For the first one, they found the following causes:

- “Sharp increase in demand”
- Consumers hoarding because of initial shortages
- Household paper products had a historically stable demand, thus, the “Just-in-time production systems” already worked at full capacity and could not produce more
- China where many suppliers are situated largely stopped exporting because of internal needs in February and March 2020
- ”Shift from demand at workplace to demand at home – inflexible production systems could not cope with change.”

For face masks, PPE and ventilators, they name additionally:

- The US stockpile was not managed properly and many products were unusable
- Outsourcing of PPE production without knowing much about the suppliers leading to no risk management, poor quality, and poor reliability
- “Inadequate in-country capabilities for not just manufacturing but also for design and development”

There were many makeshift responses trying to adapt production to the loss of quality and to manufacture domestically. Nevertheless, all efforts were “either ineffective or unsustainable”. Therefore, they **call for responsive SCs** to handle future pandemics and list how to do that. [3]
The causes for shortages within the Covid-19 pandemic can be helpful for the automatic detection of them. Since finding these causes in text data could be a warning for an uprising shortage. For example, China stopped exporting meant a lack of several products. Or more generally, manufacturing problems could be an early warning.

3.2 Covid-19 Shortage Identification

Traditionally, **no advanced technology-driven methods** are used in the analysis of shortages. Before the pandemic, anticipating SC disruptions was often implicitly included in demand forecasting or risk management. Furthermore, statistical or machine learning techniques were used to predict the demand but it was usually based on historical sales data and other numerical data but not on text, see for example [51]. Qualitative methods rely on experts’ opinions or questionnaires, for example, the Delphi method, but are usually not technology-driven [12].
These methods are a lot of human effort, which can also be biased. In addition, they are specific to a product and they are not very useful in a crisis. In a crisis, the data will deviate from historical data and experts are not of great help because many unpredictable events can happen. A method that is learning from real-world data represents the current events. Given the large amount of available data, this task could not be done by a human. For these reasons, an **automatic data-driven method is more suitable**.

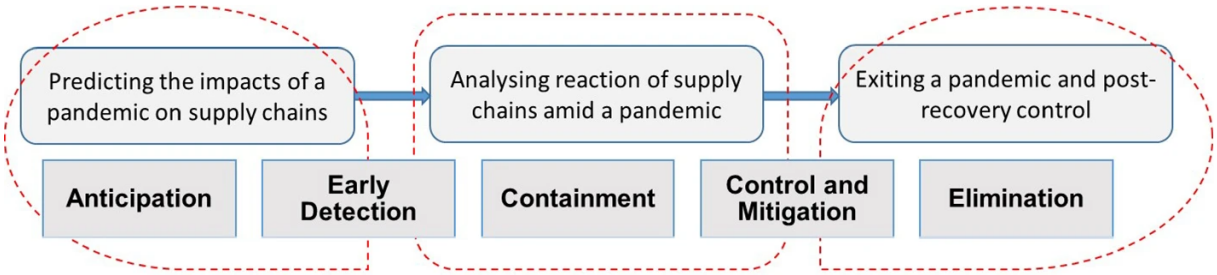


Figure 3.1: “Timing of supply chain disruption management decisions through a pandemic” [52]

There are several pieces of research in the domain of SCs that recognized the shortage issue within the Covid-19 pandemic. Ivanov [52] defined the timing of managing SC disruptions in a pandemic. That can be seen in Figure 3.1. The identification of potential SC shortages can be placed in the stage “Early Detection” and in the best case even “Anticipation”. Throughout Covid-19, there have been works in all of these stages. For example, Paul and Chowdhury [53] created a recovery plan for products that were in high demand during the pandemic.
Several works, such as Chowdhury et al. [8] and Queiroz et al. [9] reviewed papers about SC issues during the pandemic. They suggest different solutions. Among them, the use of AI and data analytics is often mentioned. As one of the research opportunities, Chowdhury et al. [8] name technology. More specifically, they pose the possible research question “How can emergent technologies support various supply chains [...] to manage the impacts of the COVID-19 pandemic and improve responsiveness?”. Similar work by Queiroz et al. [9] proposes the question “How can AI techniques contribute to developing responsive SC models in epidemics scenarios?”. The present work can be seen as an answer to those questions. These and other papers make it evident that **using technology to tackle SC problems is a research gap**, especially in pandemics.

Scenario-based Simulations

A popular technological approach in SC analysis is scenario-based simulations. Ivanov [11] aim to predict the impacts of the virus on global SCs with a simulation. Like other works, they used a dynamic simulation model to observe SC behavior over time. The dynamics of simulation models allow to include time-dependent changes. Another advantage of simulations is that very complex problems can be modeled, analyzed, and optimized. In their work, they predicted the performance of finance, customers, and lead-time as well as product inventory dynamics. They did not identify specific products which are in shortage due to these disruptions. Similar works followed this approach. For instance, Singh et al. [54] simulated the impact of Covid-19 on the food SC for certain products.
Simulations try to include all relevant outside factors, such as market disruption length or where the disease will spread, to model a scenario as realistically as possible. It is very **difficult to include all real-world influences in a model**, because even if everything is thought of, there are always unknown events, especially in a crisis. A better approach, to try to include all real-world components, is to look at text data describing the circumstances.

Statistical NLP approaches

Other papers in the area of SC analysis name especially NLP technologies to support the domain, e.g. [55, 56]. Before the pandemic, data analysis was already used in some cases for SC disruptions in disasters. For example, Papadopoulos et al. [57] analyzed public data like tweets after an earthquake to study the emerged problems in SCs, such as fuel shortages due to the earthquake.
Another example is the research of Khare et al. [58]. They claim that gasoline shortages are frequent in a disaster. They built a model from social media data to **predict gasoline shortages**. The model is based on the number of tweets about gasoline shortages. To classify that a tweet is about gasoline shortages, they used TM and keywords found via Term Frequency-Inverse Document Frequency (TF-IDF). Similar to their approach, the work at hand uses TM and TF-IDF to identify shortage-related texts and keywords. Different from what they did, this thesis focuses on several types of shortages and more TWS are applied than just TF-IDF. While the present work uses TM to find relevant texts, they used keyword search for that and only used the TM within the relevant tweets. However, what makes their research more practical than this one is that they considered the time and place of a shortage.
NLP methods are also applied to **analyze SCs in Covid-19**. For instance, Meyer et al. [13] split Covid news in three time frames. Subsequently, they weighted the terms by frequency per time frame and measured the Sentiment. This is a basic version of TWS over time which are used in the present work.
The existing NLP research around TWS has already progressed more than what is used for SC analysis. In this thesis, TWS over time are used in an ensemble to identify shortages. Other works use similar approaches to weight terms. For example, Alsaedi et al. [59] created a measure called **temporal TF-IDF**. They use time intervals of data to calculate the TF-IDF. This is done in almost the same way as the TF-IDF over time is calculated in the present work.
Just like the present work, many of these works use frequency and context-based methods. Moreover, TM is used sometimes. However, the methods are not very advanced. Most of the time, only a few methods are used. This is also due to the fact that these are SC researches using NLP approaches instead of NLP research applied on a SC problem. The idea to combine TWS over time to identify relevant terms is not new. However, the implementation and combination of the schemes are unique. **The found research gap is in the application of advanced NLP methods for SC analysis**. While there are some SC works making basic use of NLP, the state-of-the-art methods available are not considered. SCs is one of many areas where this is the case. Therefore, this study is an example of applying the state of the art in computer

science to real-world problems and making it available to other fields than computer science. Furthermore, the identification of Covid-19 shortages on large scale has not been done.

3.3 Topic Modeling

Procurement experts conventionally use **keyword searches to find relevant articles** which they manually analyze afterward. Bansal et al. [10] emphasize the use of TM to study SCs. TM assigns text to topics based on similarity. They believe that TM can find new constructs. Further, the analysis of the relationships between the articles helps to understand SC over time and anomalies can be detected. Moreover, a-priori bias by researchers can be avoided but the interpretation of the topics is still biased. Only a few studies in SC management actually used TM. Therefore, there is a need for more studies making use of TM in the SC domain. The paper of Stephany et al. [60] is an example of recent research making use of TM in the context of Covid and industry risks, such as SC interruptions.
There are some works that use **TM to analyze the topics over time throughout a disease outbreak**. For example, Chandrasekaran et al. [61] and Ghosh et al. [62] investigated topic trends over time on Twitter and in the news respectively. Like in the present work, both works also used seed words to retrieve the expected topics. Interestingly, Chandrasekaran et al. [61] identified the topic “shortage of products” within the Covid pandemic as well, just like in this thesis.
The research of Ebadi et al. [63] applied TM on Covid-related scientific articles like in this work. They found 7 topics, among them “Personal Protective Equipment”, “Rehabilitation - Panic” and “Intubation - Oxygenation”. These three topics can all be related to shortages within the pandemic. Similar to the current work, they also analyzed the most frequent terms in the dataset per month.
TM is often used to analyze topic trends, find keywords, or group data. This could also be seen in Covid-related work. Using TM to reduce a dataset is a rather rare application. Therefore, the use of **TM for SC analysis to create a domain-specific dataset rather than keyword-search is a research gap**.

3.4 Knowledge Graphs

What a KG is and how it is commonly constructed automatically was extensively discussed in the background section (see 2.3). Figure 2.1 gives an overview of the existing research in the domain of KGs. Chen et al. [64] analyzed the topics within KG research over the past 30 years. One of the future directions that they found is the **automatic construction of KGs avoiding expensive domain-specific labeling**. They call it “exploiting domain knowledge embedded in knowledge graphs of real-world applications”. Shortage identification is an example of such a real-world application making use of such a KG. Moreover, they call for enriching KGs with NLP methods. In some way, that is also done in this work.
There are several existing KGs that are very large and cover general content. There are also KGs specifically built for a certain domain, like the Covid-19 pandemic. Some of each category will be reviewed in the following.

3.4.1 Large scale Knowledge Graphs

A very large KG, aiming to cover as much knowledge as possible is **DBpedia**. It extracts structured information from Wikimedia projects, such as the infoboxes in Wikipedia, structures it to form an open, cross-domain KG, and links it to other public datasets [65]. Different instances of DBpedia can be downloaded or queried on their website which is continuously updating [66]. A

related project is Wikidata which is a KG that can be edited by anyone and supports Wikipedia, for example by creating infoboxes, instead of extracting information from it [67].
Gawriljuk et al. [34] **divide automatically constructed KG into three groups**: the KG can be built based on Information Extraction using a fixed schema, open RE without a fixed schema, or based on structured data sources. An example for the first group is NELL which uses a machine learning agent which is continuously reading the web to extract information, adding it to the KG, and improving itself on that task [68]. NELL extracts beliefs with a level of confidence, instead of facts. The second group uses open RE. That approach allows for the use of large corpora where there are many target relations or they are unknown beforehand [69]. This approach is also used in the work at hand. DBpedia, as well as YAGO, can be classified into the third group. YAGO is a KG, which is also based on Wikipedia, but it is more concise than DBpedia and aims at being very accurate and consistent [15, p.7].

	DBpedia [65]	**Wikidata [67]**	**YAGO3 [70]**	**NELL [68]**
Automatically constructed KG group [34]	based on structured data	manual	based on structured data	IE with a fixed schema
Data retrieval [71]	structured information from Wikimedia projects	manually maintained by users and bots	extracted from Wikipedia, Wordnet, Geonames, Wikidata	AI agent that reads the web
Fact representation [71]	triple	entities with multiple statements	triple with time and location	triple
Number of facts	7 billion	-	120 million	50 million candidate beliefs, confidence in 2,810,379
Number of entities	228 million	96 298 081	>10 million	-
Dynamicity [71]	dynamic	continuously updated	static	agent continuously learns and adds beliefs daily
Temporal aspect [71]	no	yes, the valid time of facts	yes	no
Content classes (top level) [72]	agent, place, time period, work, species, others	artificial entity, object, spatio-temporal entity, individual, structure, subject, quality	physical entity, permanently located entity, legal actor/ geographic entity, abstraction	abstract thing, location, item, geolocatable thing, visualizeable thing, agent
Quality of facts [71]	depends on Wikipedia and on extraction algorithm	contolled by community but users should only add verifiable information	95% accuracy, manual evaluation	assigns confidence score to extracted beliefs
Linkage [72]	links to Wikidata, YAGO, weak links to NELL through Wikipedia	links to DBpedia	links to DBpedia	weak links to DBpedia through Wikipedia

Table 3.1: Comparison of large public Knowledge Graphs

Some of the most widely used open-source KGs and their characteristics are summarized in table 3.1. Färber et al. [71] and Heist et al. [72] compared large KGs. The information in the table is either from those two reviews or from the respective source of the KG. It is noticeable that Wikidata is the most different from the other KGs. It is the only manually constructed KG and it does not store its information in triples which makes it difficult to connect to other KGs in triple format.
DBpedia is by far the largest KG out of the four. Which could mean that it has a wider coverage and is more complete but it could also mean that the data is less accurate and that it contains a lot of knowledge which is irrelevant for the task a hand. A positive aspect about Wikidata is that it is dynamically changing, like NELL and DBpedia, which on the one hand might be quite useful in a crisis where new circumstances happen every day. On the other hand, this makes it a lot harder to maintain a good quality of the data.
Looking at the classes by which the KGs are structured, DBpedia is the only one that does not have an explicit class for tangible objects which shortages usually are. YAGO3 has the best

quality of its facts as it got manually evaluated. Finally, all KGs are linked through DBpedia, which makes **DBpedia the best choice for this research**.

3.4.2 Covid-19 Knowledge Graphs

There are many KGs about the Covid-19 pandemic. Some aim at creating a general overview, others are more focused on the biomedical aspects and how they relate to each other. In the following, some of them will be discussed. It will also be concluded whether one of them is suitable as an initial KG.

KGs based on CORD-19

Many works used the CORD-19 dataset as the basis for their research, some of which are also KGs. Probably the biggest Covid KG project [14] is the **CovidGraph** which was built up collaboratively by several research institutes and companies aiming at structuring and visualizing the available Covid-19 data for researchers [73]. Next to the CORD-19 data, they added other sources. The graph contains knowledge about Covid-19 related or general biomedical papers, patents, bio-medicine, and clinical trials as well as statistical and geographic data. The team also created an interactive visual graph explorer of the KG which is freely accessible and data can be downloaded. [73]
Many other works built a **biomedical Covid-19 KG** based on the CORD-19 data and often enriched it with existing general or biomedical KGs or databases. Reese et al. [74] developed a framework to customize the KG but it mostly integrates different annotated data sources and prepares them to download as one KG. Michel et al. [75] created two KGs, one based on biomedical named entities in the CORD-19 data and the other one based on arguments mentioned in the data. Wise et al. [76] also built a KG based on biomedical entities within the CORD-19 data. Moreover, they trained a topic model on the data and added topic entities to the graph. They identified ten general topics, for instance, virology, as entities and trained a classifier based on the topic model. These graphs are mostly focused on the biomedical domain and do not contain much information on products, let alone shortages.
Directly transforming the annotated CORD-19 data, Steenwinckel et al. [77] created a **KG of the meta-data**. The entities are, for instance, papers or authors and the relations, for example, “cited by”. The data was further enriched by DBpedia and other external resources. The KG is available for download and they even provided code to use the KG [78]. Again, this KG has another focus than the work at hand.

Other Covid-19 KGs

There are other Covid-19 KGs that are not based on the CORD-19 data. Kim et al. [79] created a Covid KG with all kinds of entities related to Covid-19. Similarly to this thesis, they used **open RE** to extract triples from text. They used an entity dictionary as a seed for extraction. That is comparable to the seeding with the initial KG which was done in this work. Moreover, Covid-19 is a large topic. Therefore, the KG is **not domain-specific** and the KG could contain a lot of irrelevant information.
There is only one project, to the best of my knowledge, which built a **Covid-19 KG including information on SCs**. They tried to find disruptions in SCs. Nonetheless, as it seems, the project has been discontinued and the available KG only contains examples of that idea. [80]
Many companies use **Enterprise KGs** to represent their internal data, such as SCs or products. These KGs, however, only contain the companies’ products and suppliers and are usually not available to the public. [81] For example, aut [82] built a Product KG of all of Amazon’s products and product types.

Conclusion

Looking at the existing KGs using the CORD-19 dataset and the ones representing products, there is no KG representing products or shortages in SCs within the Covid-19 pandemic. Therefore, **there is a need for a KG containing objects or products relevant to Covid-19 to identify shortages**.
None of the Covid-related KGs seem to be a good fit for an initial KG because they are too far away from the SC domain. Therefore, in this study a **subgraph of DBpedia was used as an initial KG**. It is not only very large and covers almost any domain but it is also human-curated. Moreover, it connects other existing KGs. The downside of using DBpedia is that most of the relations are not very meaningful for this application. A large part of the relations is the link to another Wikipedia page. If an entity is mentioned on another entity's Wikipedia page, it is most likely related but it does not indicate in what way.

3.4.3 Knowledge-Aware Applications

KGs have a variety of applications. Figure 2.1 shows some examples of them. In that figure, the application of the KG developed in this work could be categorized as Natural Language Understanding. The created KG supports a term weighting problem. It could also be classified as "other Application", namely shortages in SCs.
There are many **other works that leverage a KG for related tasks**, such as text classification or prediction. For example, Li et al. [83] classified symptoms as written text to diseases with the help of a KG. Another example is the work of Jiang et al. [84]. They used a KG to extract features from text and used those features for prediction.
There are no examples of applications of KGs in the SC shortage domain. Nonetheless, according to [85] KGs in the SC domain are gaining popularity during the pandemic. Therefore, this **KG can be seen as a new contribution to SC shortage domain**.

KG embedding library. There are several libraries that implement KG embeddings. The one used in this thesis, AmpliGraph [86] offers embeddings for KGs and downstream tasks which can be applied to them. Some of the other libraries were compared by looking at the API documentation. The advantages of AmpliGraph over the others are: 1. it is not a command-line interface, 2. GPU training is possible, 3. it offers several downstream tasks, 4. it offers different embedding algorithms and it is easy to import the data. PyKeen [87] also fulfills the requirements and has even more KG embedding algorithms but it does not offer deduplication. In addition, it is not possible to query the top n results for link prediction, it will always return all. This is why AmpliGraph is used.

3.5 Summary of the Chapter

This chapter gave some related work to each of the parts of the method. First, some causes of SC shortages in Covid-19 were stated. Some of them are: an unexpected high demand, hoarding, and missing stockpiles. Second, the current methods to anticipate SC disruptions were reviewed. The pandemic caused a surge in technology-driven approaches aiming to identify shortages. Nevertheless, NLP approaches are still rare and advanced NLP methods can be seen as a research gap. More commonly, human evaluation or simulations are used. Thereafter, some studies related to Covid-19 using TM were reviewed. They made use of TM to analyze Covid-19 but not to create a domain-related dataset, as is done in this thesis. The last part compared some large-scale KGs and some related to Covid-19. None of them are about Covid-19 shortages, which is, therefore, a research gap. It was concluded that a sub-graph from DBpedia is the best fit for an initial KG.

4 DATA PREPARATION

This chapter is about the dataset. First, the preprocessing is explained. Second, some simplifying assumptions are mentioned. The third part is about the creation of the ground truth shortage list from external sources. The last part is a quick analysis of the data to show that the data fits the purpose of identifying shortages well.

4.1 Data Preparation

For the analysis and for the construction of the KG, the **Covid-19 Open Research Dataset (CORD-19)** of Covid related publications was used [14]. The dataset is an initiative of the Allen Institute for AI in cooperation with other partners. It is hosted on Kaggle and regularly updated. Each instance of data refers to a paper somehow related to corona-viruses, containing additional information, such as authors and abstract. The metadata contains 19 columns in total, namely unique id (Cord-uid), title, doi, source, license, abstract, publish time, authors, journal, the path to the full text, the URL, and ids of the article from other databases. In addition, there is the full-text to some of the articles. There is also a JSON schema definition for the full-text structure.

Removing Duplicates

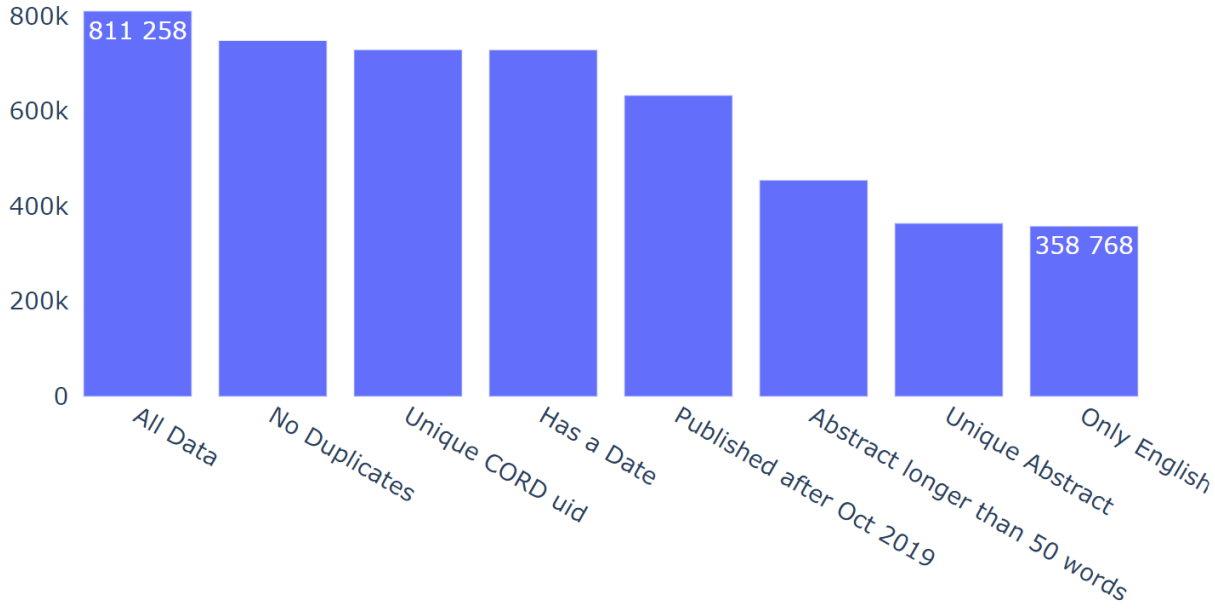


Figure 4.1: CORD-19 dataset amount of data after preprocessing

Figure 4.1 shows the total number of instances (811 258 on 03/11/2021) in the dataset and the subset used for the task. First, the duplicate rows were dropped, and then the articles with the same Cord-uid. Also, only articles that have a date entry were kept. The date entries of the

remaining articles were transformed to year and month only. Around 50% of the dataset only has a year and no month. Since some of the analysis uses the month and year of the articles, that information was saved in an extra column “has_month”. That way the time-dependent methods can only use the articles with a month.
The Figure 4.2 shows the number of publications per month for the articles which have a month. It can clearly be seen that there was a large surge in publication numbers about the coronavirus a few months after the pandemic began at the beginning of 2020 which stayed constant until at least October 2021.

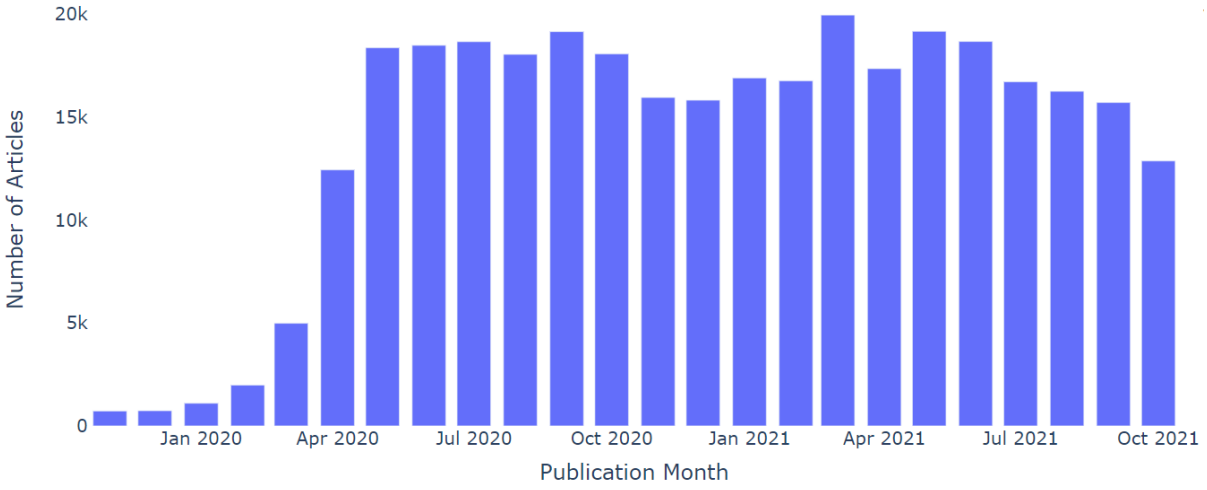


Figure 4.2: Number of articles per month from Nov 2019 until Oct 2021

Keeping only relevant Articles

For most of the analyses, only the abstracts were considered because of two reasons. First, only about half of the articles in the preprocessed dataset contain the full text. Second, the abstracts contain the relevant information in a compact form. The full text might contain more details but it also introduces noise. To select only relevant abstracts, additional steps were taken. Only publications during the Covid-19 pandemic (from November 2019 until October 2021) were kept. The reason is that the dataset also contains articles about the coronaviruses before the pandemic that probably do not contain information about shortages during the pandemic. The end month of October is simply the month when the dataset was extracted. Furthermore, only abstracts that are longer than 50 words were kept because shorter texts were often only a default sentence or some other meta information. When removing all the special characters and punctuation, a lot more duplicates could be identified and removed. This could be, for example, due to double white space. Finally, only English abstracts were considered (based on the first 100 characters). The language check is needed because there is a minority of other languages than English in the dataset and the analyses will only be done in one language. In the end, from the entire CORD-19 dataset, only 44% (**358 768 articles**) were actually relevant for the task at hand.

Preprocessing of the Abstracts

The next step of the preprocessing is the cleaning of the texts. Abstracts often begin with a word such as “background” or “abstract” followed by a colon. These words are not relevant and were discarded. After that, special characters, standalone numbers, and also double white spaces were removed (Regex: `[^\w.?! ]`). At first, more preprocessing was done, such as removing stop words or lemmatizing. However, many of the advanced NLP methods consider

context, semantics, punctuation, and casing. Applying heavy preprocessing could thus worsen their performances.

Noun Phrase Extraction

From these texts, the noun phrases were extracted. This was done because a lot of NLP tasks are based on terms. Single words often do not represent the meaning in which they were used in text, e.g. “face mask” instead of only “face”. For that, two different Python libraries were used (TextBlob [88] and Spacy [89]). Surprisingly they did not find the same noun phrases, therefore both were used and combined. However, noun phrases from the second method were only added if the other method did not find the phrase yet. This way, the number of occurrences of a noun phrase within an abstract is not doubled if both methods find the same one. The noun phrases were extracted per abstract and per sentence which has some advantages for methods which work on sentence-level. Next, the noun phrases were set to lower case and special characters, articles (e.g. the) and single numbers were removed from the noun phrases. Lastly, general paper terms that occur in almost any abstract (e.g. abstract, objective) and synonyms for Covid-19 were removed from the noun phrases (see B.1 for the exact lists). The reason for that is that the dataset is about Covid-19 research publications. Therefore, almost every article contains information about these two components and that does not give any valuable information. These terms are very frequent and can be distracting for certain methods.
The **final dataset** contains 358 768 articles each containing the following columns: “Cord-uid”, “abstract_processed”, “date”, “has_month”, “noun_phrases”, “noun_phrases_sentence”. Most of the methods use the noun phrases per abstract, some the abstract. How exactly they are used will be described in the respective method.

4.2 Simplifying Assumptions

During the pandemic, there were several shortages in SCs. It is difficult to define if something is in shortage because of the complexity of SCs and time and place restrictions. The focus of this study is to identify *potential* shortages. Therefore, some simplifying assumptions were taken neglecting these challenges. They will be explained in the following.

Complexity of SCs. SCs involve many stakeholders, often spread out across the world. There is no formal definition of a shortage. Thus, every stakeholder might define a shortage differently. Furthermore, the severity of a shortage should play a role, i.e. how large is the gap between supply and demand. In addition, there is no central shortage warning system [7]. So if someone reports a shortage, some stakeholders along the SC might not know about it. Moreover, every country has a different way of reporting them and not every shortage is reported. A supplier might not want to report a shortage to avoid clients looking for another supplier. All these aspects make it difficult to identify products in shortage. As a simplification, anything which was mentioned to be scarce somewhere along the SC is assumed to be a shortage.

Place. If a model were to predict a certain product to be in shortage, then that can be true for a certain region but false for other places. The pandemic is worldwide, if certain products are said to be in shortage because of it, most likely that will not be the case for every part of the world. To tell where exactly something is in short supply is difficult enough for a human to do and is left out in this thesis.

Time. Another constraint is given by time. Every shortage is temporary. Similar to the place, discovering the exact start and end of a shortage is not trivial. Thus, also the time frame will be disregarded in this work.
In conclusion, anything which is hinted to be a shortage somewhere in the world at some point in time will be assumed a shortage in this study. In addition, something which is not in shortage currently or not in all places but it was scarce before at a certain point in time and place is in danger of becoming scarce again. Hence, a shortage warning system would be right in suggesting these things.

4.3 Creating the List of Shortages

In order to measure the performance of the algorithm, an extensive list of known shortages during the Covid-19 pandemic was created manually. The shortages in the list are not guaranteed to have been an actual shortage, since the goal is to find potential shortages. Thus, if a product is predicted which was no actual shortage but it was somewhere mentioned to be one, then it is still a potential shortage and therefore correct. The list should be considered a collection of actual and potential shortages. So if the algorithm were to predict any of them, it would be considered correct. Additionally, since shortages are difficult to determine, this list is not complete. Furthermore, another list of terms that indicate a shortage was made. That list is helpful to find shortages and set the context around them.

List of Shortages vs. List of Shortage Indicators

There are two super-categories: shortage indicators and shortage terms. The shortage indicators are terms without any relation to the Covid-19 pandemic, e.g. “scarcity” or “increase” but also terms related to SCs like “procurement”. **The shortage indicators were used for seeding and tuning the methods because they do not include any bias towards Covid-19**. The second category contains the shortages within the Covid-19 pandemic, e.g. “face mask” or “PPE”. That also includes company names of products in shortage. **The list of actual Covid-19 shortages was only used for evaluation.**
The lists were partially created by a SC expert and then extended based on some sources [90, 91, 92, 93]. Furthermore, terms that were found in the dataset by visual inspection were also added to the list. The sources are sometimes just blog posts and therefore not necessarily reliable but the goal was to find potential shortages.
Since exact text matching was performed to find the terms in text, **false positives occurred**. For example, the word “chip” (computer chip) can also be part of another word. Therefore, spaces were added before and after these words. Ambiguous words were filtered out completely. This of course does not eliminate false positives entirely.

Types of Shortages

Each shortage was classified as a certain type to make sure the algorithm finds different types of shortages and not just synonyms for the same one. For the list of shortage indicators, the following types were identified: *shortage, product, procure, stock, require and increase*. Each category contains synonyms to the respective term. For the list of shortages, the subcategories are: *company, ppe, mask, test, ventilation, sanitize, raw material, paper, consumer good, blood, chlorine, vaccine, container, gas, chip shortage, swab, medicine, and food*. In this case, each term in a category has the relation “is a” to the category. The company category is special because the company itself cannot be a shortage. However, a certain product they produce can be. Knowing the company name can help with the analysis of a shortage. Moreover,

sometimes a company name is used to refer to the product (e.g. “AstraZeneca” as company name referring to the Covid vaccine). The complete list can be found in the appendix A.1.

Evaluation of the Methods

The components of the algorithm are evaluated with these lists. Generally, the overlap to the shortage terms and sometimes also to the shortage indicators is calculated. Based on that, precision, recall, and the F-score are calculated. The retrieved terms are all terms returned by a method. The relevant terms are the shortage terms or indicators and the true positives are the overlap.

4.4 Quick Analysis of the Data

To get insights into the value of the CORD-19 dataset, a first quick inspection was done. Out of the 210 terms in the shortage term lists 204 were mentioned in the dataset. The terms which are not in the dataset are: fisher paykel, oropharyngeal nasopharyngeal swab, heating ventilation system, thermofisher inc, adenovirus serotype vaccine, and ad26cov2s. Moreover, around 109 000 articles contain at least one of the terms. Therefore, the dataset is actually talking about the terms in question. This shows that **the dataset is a good fit to analyze shortages**.

First occurrence of the shortages

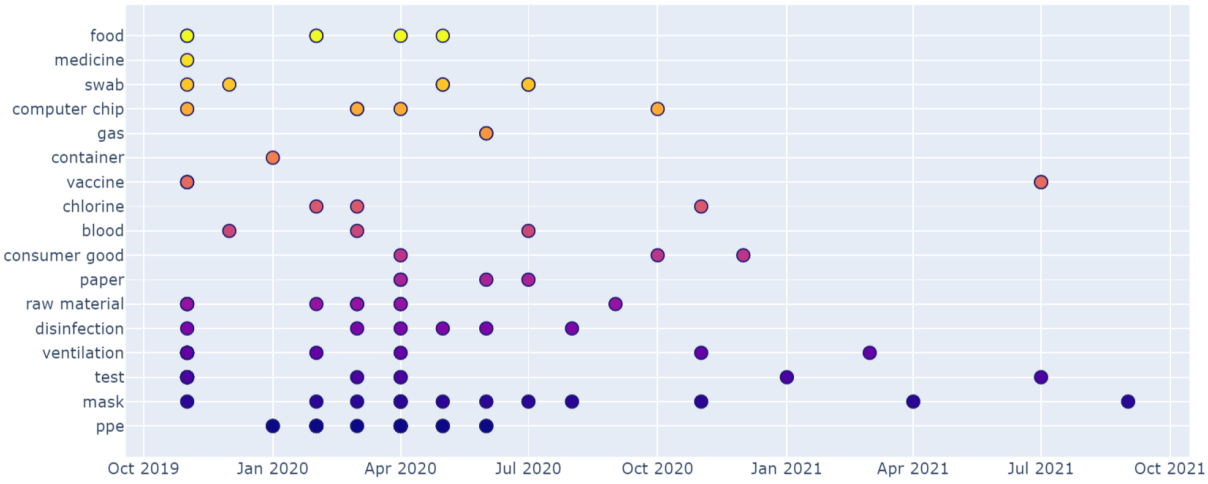


Figure 4.3: First occurrences of products in shortage in the CORD-19 data by type

It is interesting to look at when the shortages were first mentioned in the dataset. This helps to see the distribution of the terms over time and to get an estimation of when the shortages first occurred. Figure 4.3 shows the first occurrence of each shortage term in the Covid-related shortage term list by type where each dot represents a different term of that type. For example, a dot of the type “mask” could be “face mask” and another dot first mentioned later in time could be “N95-respirator”. Only articles that have metadata indicating the month of publication were considered. This does not necessarily mean that the shortage was at that time, it only means that within the CORD-19 dataset it is the first time that the term was mentioned. It is possible that the product was in shortage earlier but it was not mentioned in the dataset or a synonym for the product was used. Moreover, not every dot in the figure means it is actually in shortage, it could also just be mentioned in the data for another reason. However, the figure gives a first indication of the Covid-19 shortage term distribution over time.

Development of the Shortage term Occurrences

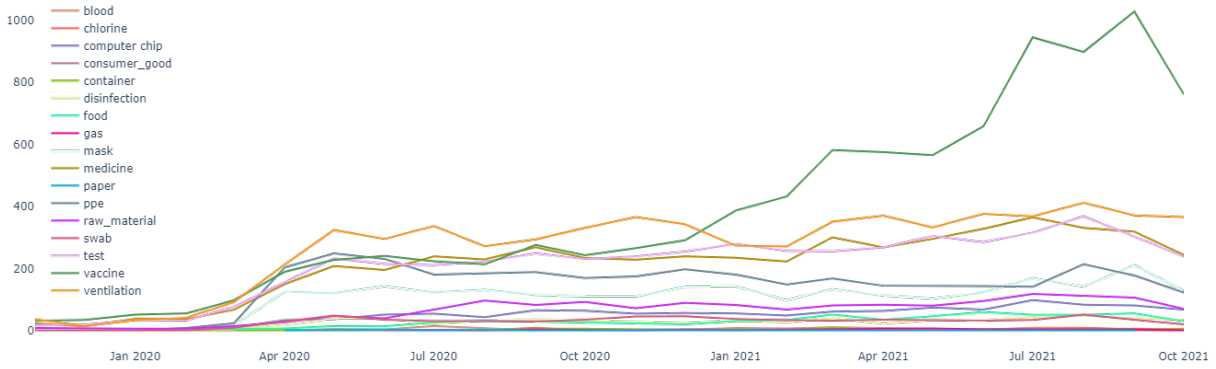


Figure 4.4: Number of articles that contain a certain shortage type in the CORD-19 data

The Figure 4.4 shows how the shortage term occurrences developed. For each type, the number of articles was counted that contain at least one of the terms belonging to that type. It can clearly be seen that the shortage mentions increased throughout the pandemic. Most types remained constant after a first surge. An exception is the **vaccines which experienced a heavy gain throughout 2021**. This corresponds to the real world, as Covid vaccines were inequitably distributed in the world and thus a shortage in some places in 2021 [94].

4.5 Summary of the Chapter

This chapter contains four parts. First, the preprocessing of the abstracts and extraction of the noun phrases from the CORD-19 data was described. Second, three assumptions about shortages were taken as a simplification. Anything is assumed to be a shortage which was claimed to be one independent of time and place. Third, two lists of shortages were created as a ground truth. A list of Covid-19 shortages and a list of shortages indicators. Both include a type. The evaluation using the ground truth lists with the F-score was defined as well. Finally, the first mention of the shortages and the development of the number of articles that mention a term in the CORD-19 data were analyzed. The conclusion of that was that the data is a good fit to analyze shortages.

5 STATISTICAL APPROACHES TO IDENTIFY POTENTIAL SHORTAGES

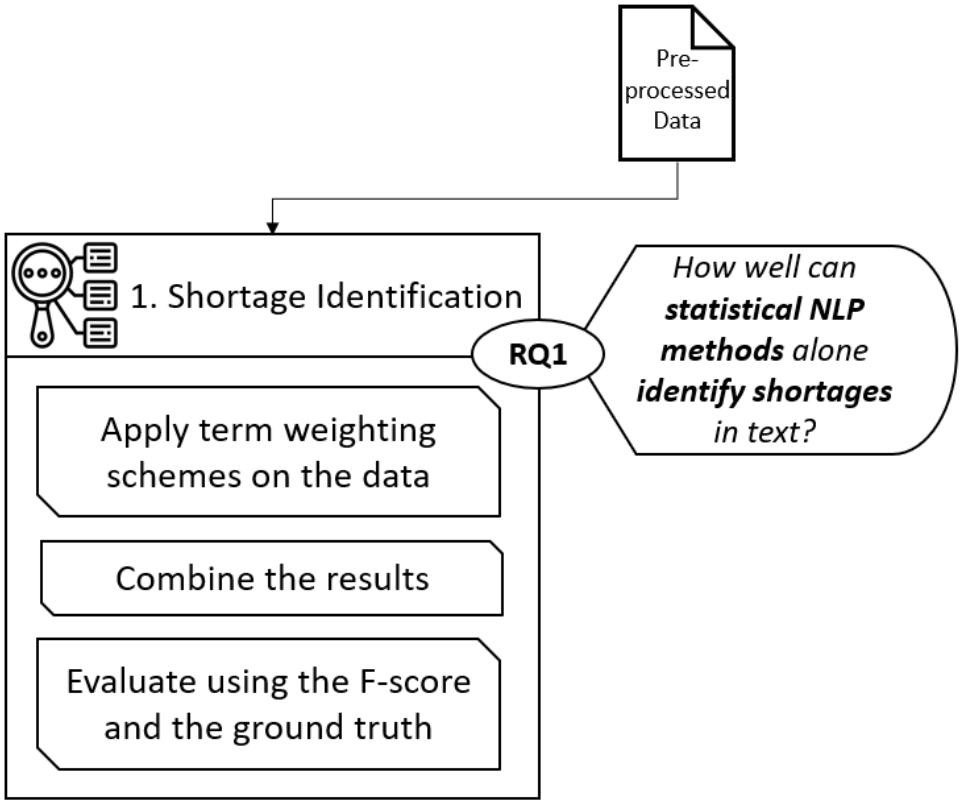


Figure 5.1: Overview of the methodology and experiments of this chapter

Figure 5.1 gives an overview of the chapter. It will answer the first research question of *How well can statistical NLP methods alone identify shortages in text?*. The methodology explains the evaluation of the predicted list of terms and the TWS over time that identify shortages. The experiment of this chapter is the identification of shortages. The TWS over time are applied to the entire dataset and combined. The measures F-score, recall and precision evaluate the results with the help of the ground truth shortage lists.

5.1 Methodology

To identify shortages, term weighting schemes over time are applied on the CORD-19 data. The first part of the methodology describes how the suggested shortages are evaluated. The second part describes how the methods retrieve these terms.

5.1.1 Evaluation

F-score, precision, and recall and the **time** spent per method evaluate the effectiveness of the methods and the different settings. The time spent per method is measured empirically, in seconds.
To calculate the **evaluation measures**, all methods are executed and the resulting lists of top terms are evaluated **individually and cumulatively**. The retrieved terms are all unique terms within the list of top terms. The true positives are all retrieved terms that contain a shortage term. Since relevant hits should not be missed only because the term is part of another entity, **partial matches** are counted as well. For example, if the list of top terms is [“face mask”, “the face mask”, “mouth”, “face mask”], then the retrieved terms are [“face mask”, “the face mask”, “mouth”] and the true positives are [“face mask”, “the face mask”]. “Mouth” is not part of the relevant terms because it does not contain a shortage term. The relevant terms need to be extended by the true positives because partial matches count as well. This means that there can be more matches than shortage terms which would lead to a recall of more than one. The precision, recall, and F-score are calculated based on those numbers. Thereafter, all retrieved terms of all methods are combined to one unique list and the evaluation measures are calculated again.
In addition, the types of shortages are extracted. Each shortage term was given a type when the lists were created. Since the algorithm should predict terms of different types, the **number of types** is considered in the evaluation. This is done per method and for all methods together.

5.1.2 Term Weighting Schemes over Time

To automatically recognize that a product is in short supply, two main features are leveraged: context knowledge and term frequency. First, if something is explicitly said to be in need, it is a good indicator that it is a shortage. Therefore, the context of certain terms like “shortage” or “demand” most likely also mentions what is missing. Second, a product that is in shortage is likely to occur more often at a certain point in time than the rest of the time because the shortage is discussed. As a consequence, more articles will mention the product in question. Based on these two hypotheses, several methods were developed and applied to the dataset to extract potential shortages. The different methods are not meant to compete with each other, resulting in one best method which finds the most shortages. Rather, they should work as an ensemble. Moreover, since the methods have a different way of weighting the terms, one method can find relevant terms that are not found by the other ones.
The following will explain, the evaluation and the implementation of the TWS. The methods use the noun phrases within the abstracts instead of single words because they are often more meaningful. If, for example, the term “face mask” is very frequent, then it would not really help to know that the word “face” is mentioned a lot.

Term Frequency

The easiest approach is counting the occurrences of terms within the dataset. Since shortages are time-dependent, this is best done over time. If a certain term is mentioned a lot within a limited time frame, it is potentially in shortage. Of course, there can be several other reasons why a term has a high frequency in a certain time interval. However, if this method is applied to a dataset only related to shortages, the most frequent terms are more likely to be shortages. This dataset will be created in chapter 6.
To calculate the term frequency, the articles were grouped by month and the **noun phrase mentions within the month** were counted and normalized with the total number of noun phrases in that month. This is done so that the relative frequency is taken which avoids that a term gets

a higher frequency in a certain month only because there are more articles in that month. The normalization makes the term frequencies of different months comparable.
Per term, this results in 24 values, i.e. one relative frequency per month. Only the value of the month with the highest frequency was kept per term. This results in one relative frequency per term. That avoids selecting a term more than once if it has a very high frequency in several months. The terms which are returned as a potential shortage, are the **top n terms ordered by maximum monthly relative frequency throughout all months**. This means terms that are very frequent in a certain month and thus have a high relative frequency are returned.
Figure 5.2 shows an example plot of the relative frequency of “PPE” over time. PPE appeared in the top 100 terms weighted by the described method. The curve has a similar course to the actual number of requests of PPE at the beginning of the pandemic [95].

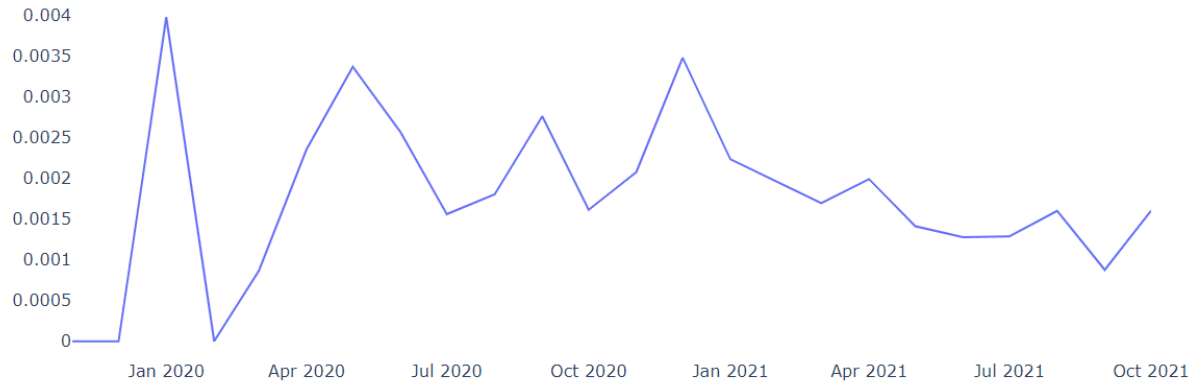


Figure 5.2: Example relative frequency over time of “PPE”

Term Frequency Inverse Document Frequency

The most frequent terms per month contain many terms which are always frequent in normal texts or within this dataset. For example in this dataset, the most frequent noun phrases are “we” and “patients”. Hence, a term being frequent can mean not only that it is important in a certain time frame but it can also just mean that it is always used a lot. It is more interesting to see **which terms are frequent within a certain time frame but not within others**.
A slightly adjusted version of **Term Frequency-Inverse Document Frequency (TF-IDF)** can be used for that. A definition of TF-IDF can be found in [96]. It represents the terms in a document as vectors by giving every term in a document a weighted score. Inverse Document Frequency (IDF) is the inverse of the relative number of documents that contain a term. Term Frequency (TF) is the frequency of a term per document. While IDF is calculated for the entire dataset, TF is calculated per document. TF-IDF is the product of the two. As a result, it finds terms that are frequent in one document but not in others.
In this work, TF-IDF was changed so that it is valid for all documents in a month instead of just for one document. More specifically, **each month was considered one document**. So the documents in that month were concatenated and the TF-IDF was calculated. Only terms were considered that appeared in at least five months and in not more than 19 months (out of 24 months in total). This way, rare terms, and very frequent terms would not be considered. To implement this, the TfidfVectorizer from Scikit-learn was used [97]. The equations below explain it mathematically.

$$\text{TF} - \text{IDF}_{\text{month}} = \text{TF}_{\text{month}} \times \text{IDF}$$

$$\text{TF}_{\text{month}} = \frac{\text{raw term count in all docs in that month}}{N_{\text{terms in that month}}}$$

$$\text{IDF}_{\text{month}} = \log\left(\frac{1 + N_{\text{months}}}{1 + N_{\text{months that contain a certain term}}} + 1\right)$$

N denotes the number of documents or terms. 1 was added to all parts of the IDF equation for smoothing, this prevents division by zero.
To obtain the top n terms with the highest TF-IDF weighting from this, the **maximum monthly score** was taken per term and subsequently the **top n terms** were selected. I.e.

$$\text{top terms} = \text{top}_{\text{n}}(\max(\text{TF} - \text{IDF}_{\text{month}}))$$

Figure 5.3 shows an example of the term “ventilators” which was part of the top terms. There is a high peak in April 2020 which suggests a shortage at that time.

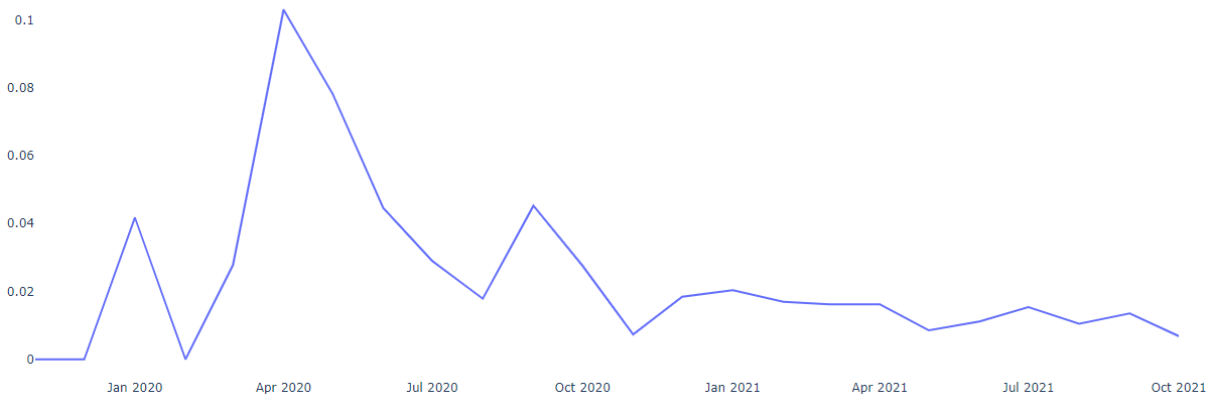


Figure 5.3: Monthly TF-IDF for “ventilators”

Context Occurrences

As already stated, in addition to frequency, the context of certain shortage indicators can also be a way to find potential shortages. Another term weighting scheme follows from that: the **top terms in the context of certain keywords**. These keywords can be, for instance, the shortage indicators. For example, when the demand for the term “mask” is high, then it is probably often mentioned in the context of “demand” or “need”.
To calculate this, all terms mentioned in the context of a given list of keywords have to be extracted. For that, all mentions of the keywords in the dataset are found, and subsequently, **all noun phrases around the keyword are extracted within a certain window size** (measured in the number of words). This is done per document so that the structure of the dataset remains the same. The resulting reduced dataset only contains a list of noun phrases per document that were found in the context of the keywords. The documents that did not contain any of the keywords are discarded.
To retrieve the shortage predictions, the described term frequency method or the monthly TF-IDF is applied on the reduced dataset. The top terms to be returned are calculated as part of the frequency or TF-IDF method.

Figure 5.4 shows an example of that. Within the context of the shortage indicators, the term “medical resources” occurred quite frequently with a peak in the month of February. From this, it could be hypothesized that the shortage of medical resources was the worst in February 2020.

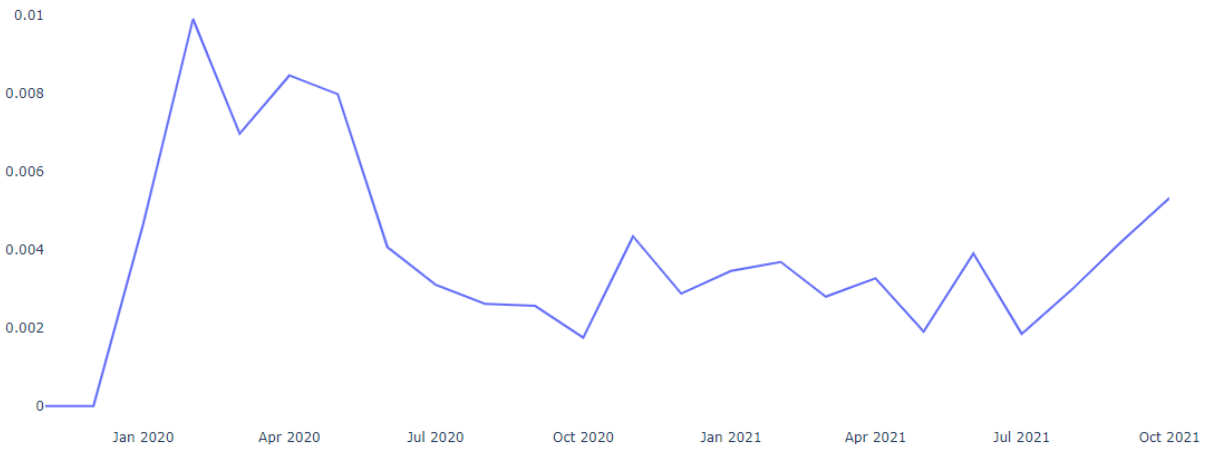


Figure 5.4: Frequency of “medical resources” in context of the shortage indicators

Differences in Context Occurrences

The previous method identifies potential shortages by frequent mentions in the context of keywords. When looking at their time series, it can be seen that the curve suddenly increases which could mean that the product is in shortage. In the example figure, that would be the transition from January to February 2020. To identify these sudden increases, the maximum difference of time frames can be considered. **A large increase in mentions means a large difference from one timeframe to the next**.
We calculated this by computing the element-wise difference of the dataset originating from the context method shifted by one month and the unshifted one. To retrieve the top terms, per term, the maximum difference is taken and from those, the top n terms are returned. The list of terms thus contains the **biggest change from one month to the next in occurrences of mentions within the context of a keyword list**. Instead of frequency, it can again also be the change in TF-IDF per term and month.

Word Embedding

Another way to look at the context of terms is to create a word embedding. A word embedding is a representation of the meaning of words learned from their distributions in texts. With such an embedding, similar terms, so **terms which are used in a similar context**, can be retrieved. In the case of shortages, it can return similar words to a seed shortage (e.g. “face masks”) or similar words to the shortage indicators. Moreover, calculations with words can be done with an embedding. For instance, “product” + “scarce” could result in a product in short supply.
The **embedding is built** on the entire data. Additional preprocessing is done on the abstracts to improve the embedding. First, all numbers and punctuation are filtered out. Second, the noun phrases found in the text are concatenated with an underscore, so that they are considered as one word. Finally, each word is lemmatized, so that alterations of words, like plurals, are resolved to one word in the embedding.
To find the top terms, the **n most similar words were retrieved for each term in a list of keywords**. Subsequently, the keywords were removed from that list and the n most common terms within the list were returned. Terms occur more often in this list because they can be similar to more than one keyword. Similar terms were retrieved for each keyword separately.

5.2 Experiment: Identifying Potential Shortages

The following experiment was done to answer the first research question. The research question was *How well can statistical NLP methods alone identify shortages in text?*

5.2.1 Experimental Setup

We executed the methods described in the methodology on the **entire dataset**. For now, **all lists** of top terms resulting from the methods were **combined**. This can lead to many false positives, so elements that are retrieved but not relevant. In this case, that could be desirable because the retrieved elements which are not in the list of known shortages could be an unknown shortage or a potential one.
We set the parameters for the methods as follows. This is not necessarily the best setting for the parameters. It is a possible setting that performed well. In future work, some extensive tuning of the parameters could be done. Three different **numbers of top terms** were compared (100, 500, 1000). The top terms were the same for all of the methods. Some other parameters were set and stayed fixed for all runs.
We calculated the **context** occurrences and differences once with the term frequency and once with TF-IDF. The context size was always set to 30 and the seed terms were always the shortage indicators.
For the **word embedding**, the seed terms were once the shortage indicators and once the terms “mask” and “shortage”. The Covid-related terms and the paper terms were filtered out for the same reason they were filtered out in preprocessing. This is only done for this method because the other ones are based on the noun phrases from which these terms were already removed.
We tested other parameters for the embedding and other similar terms. In addition, an alternative implementation for word embeddings called “Temporal Word Embeddings with a Compass” [98] was tested. That algorithm trains a new embedding per time slice and aligns them. It seemed to be very promising for this application but the results were not better. From all the tried things, the described approach performed best. However, this does not mean that there is no better approach.

5.2.2 Results

Top Terms	Precision	Recall	F-Score	Number of Types	Time in seconds
100	0.07	0.22	0.11	6	1383
500	0.05	0.5	0.09	12	1366
1000	0.04	0.63	0.08	13	1173

Table 5.1: Evaluation measures for all methods together on the entire dataset

Table 5.1 shows the cumulative results of the shortage identification. That means all returned terms of the different methods were assembled and considered the retrieved terms. The results of each individual method can be found in appendix C.1. The results show that the **TWS over time can actually identify some of the shortages** which we looked for. The highest **F-score of 0.11** is reached when selecting 100 top terms.
The more terms are retrieved, the higher the recall and the lower the precision. This makes sense because retrieving more terms means retrieving more relevant but also more irrelevant terms. The precision degrades relatively quicker than the recall, which can be seen in the decrease in F-score.

The **most accurate list was returned for 100 terms**. However, only around **7% of the list is relevant**, the rest is noise or additional shortages which are not part of the list. As mentioned before, the ground truth is not complete, thus the precision might be underestimated. The **highest number of shortage terms was retrieved at 1000 terms with a recall of 0.63**. These numbers can still be improved by the use of a KG. The processing time could be decreased by reducing the dataset to only relevant articles.
It can be seen that the **number of different shortage types increases with the number of retrieved terms**. Interestingly, the number of types doubled from 100 to 500 top terms but only one more type was discovered by retrieving the top 1000 terms instead of 500. This behaves similarly to the recall which also more than doubled from 100 to 500 and only increased slightly from 500 to 1000 terms.

5.3 Summary of the Chapter

This chapter answered the first research question by measuring the performance of only statistical NLP methods to identify shortages in text. The applied TWS are term frequency, TF-IDF, context occurrences, the difference in context occurrences, and similar terms based on word embeddings. The methodology described the implementation and evaluation of these schemes. In an experiment, the TWS were applied and combined. The proposed method succeeded in identifying shortages with the highest F-score being 0.11.

6 TOPIC MODELING TO REDUCE THE DATA

The experiments regarding the first research question showed that it is possible to identify shortages with the proposed TWS over time using the CORD-19 dataset. However, the **TWS find many terms which cannot be a product in shortage because the semantics, so the meaning of the terms, are not considered**. Therefore, this thesis proposes to encode the lexical knowledge of the text in a **KG** and make use of it when identifying the potential shortages.
To make the **KG domain-specific, the source data should be about shortages in SCs**. The CORD-19 dataset contains around 360 000 articles about Covid-19. As the first analysis of the data showed, many of them mention shortage terms. This does not mean that they are mentioned as a scarcity. However, the context around these terms and their frequency can help to identify them as a shortage. The rest of the articles are most likely irrelevant for this study and can be discarded for further analysis. This helps to **reduce noise** greatly and also **saves computing power**. Nevertheless, some shortages might not be mentioned in the smaller set of articles and will not be returned. Thus, while this might increase precision and reduce the processing time, it could decrease recall.

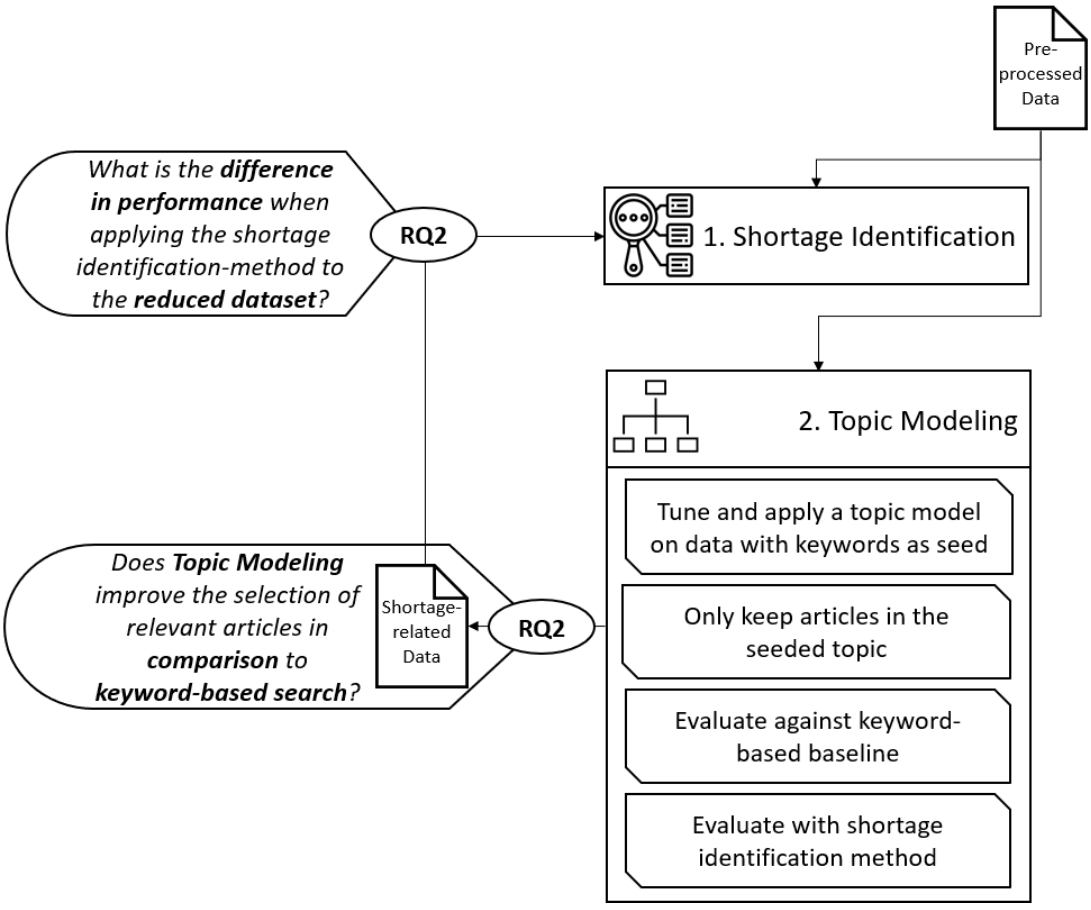


Figure 6.1: Overview of the methodology and experiments of this chapter

Figure 6.1 displays the structure of this chapter. The methodology explains the TM approach and how it is evaluated. There are four experiments in this chapter. The first one evaluates which part of the articles to use for TM. The second experiment tunes the TM algorithm. The next experiment evaluates the selection of relevant articles using TM. That answers the first part of the second research question: *Does Topic Modeling improve the selection of relevant articles in comparison to keyword-based search?* The last experiment repeats the shortage-identification method on the reduced dataset and compares the results to the ones of the entire dataset. That answers the second part of the question: *What is the difference in performance when applying the shortage-identification method to the reduced dataset?* The result is a shortage-related dataset which will be used as a basis of the KG.

6.1 Methodology

How can relevant articles be found if the shortages are unknown? That is the scenario for which this shortage prediction system is developed. In the field of SCs, practitioners usually perform **keyword-based search**. In this case, those are **all articles that contain a shortage indicator**. Keyword searches require a lot of work as keywords have to be identified and the articles have to be scanned manually to know if they are relevant. Moreover, the keyword list is never complete. As the motivation was to improve their working habits by the means of NLP methods, a more sophisticated approach was tested.
This research proposes to use TM to select relevant articles as **TM sorts data into topics by relatedness**. The basic idea behind TM is that two articles that have a similar distribution of words are related. There are some advantages over keyword-based search. First, in addition to articles that mention a keyword, articles can be found that do not mention a keyword but are related to those that do. This is helpful because the list of shortage indicators might not be complete. Second, there might be articles that mention a keyword but are not about shortages. Those articles would not be selected because they are not related to the rest.
The following first explains the algorithm including an alternative approach and an evaluation method. After that, the evaluation of the article selection with a baseline will be described.

6.1.1 Topic Modeling Algorithm

The following describes the design of the TM algorithm to select the relevant articles. That includes the steps to create a topic model and the introduction of the hyperparameters to be tuned. To ensure that one of the topics is about shortages, it helps to seed it with a list of terms. This is called guided LDA. In this study, TM with guided LDA is implemented as follows:

1. Create a **count vectorizer** (implementation [97]) that ignores terms that are only mentioned once and terms with a higher document frequency than 0.95, only consider the top 10 000 features with no tokenizer and no preprocessing
2. **Fit the count vectorizer to the noun phrases** per document to retrieve a matrix of term counts
3. Keep only the **seed terms** (given as an input) that are in the vocabulary of the count vectorizer
4. Optionally, keep only a **random sample from the list of seed terms** of size x, with $x \leq$ length of seed terms in the vocabulary
5. **Create a dictionary of seed terms** where the word id (retrieved from the count vectorizer) for each seed term is the key and the value is 0 for all terms (corresponding to **topic 0**)

6. **Create a guided LDA model** (implementation [99]) with a number of iterations, k number of topics and the parameters α and β
7. **Fit the model** to the matrix of term counts with the dictionary of seed terms as input and a seed_confidence indicating the confidence of the seed dictionary
8. **Classify the entire dataset** with the topic model
9. **Create the reduced dataset** by only keeping the articles in topic 0

Evaluation based on topic terms.

The topic models need to be evaluated and compared for two reasons. Since the LDA algorithm is unstable, the topics are distributed differently every time. Moreover, hyperparameter tuning is performed which will be described in more detail in the experiments. The analysis of the correlation between the model parameters and the evaluation measures will be part of the experiments as well. This evaluation is based on the topic terms and not on the selected articles.
Topic coherence evaluates the models (C_v coherence). However, that does not necessarily measure whether a shortage topic is included or not. Therefore, in addition, the **F-score based on the shortage indicators** is used. It is only calculated for topic 0, so the seeded topic. Note that the shortage *indicators* are used to not bias the parameter tuning with the actual shortages. The retrieved terms are the number of terms within a topic. Since each topic contains all the terms but with a different probability, this is not so easy to determine. In this thesis, the terms are sorted by their probability and the top n terms are retrieved. Thus, the **number of retrieved terms is always equal to n**. The **true positives are all retrieved terms that contain a shortage indicator** (also partial matches).

Dynamic Topic Modeling

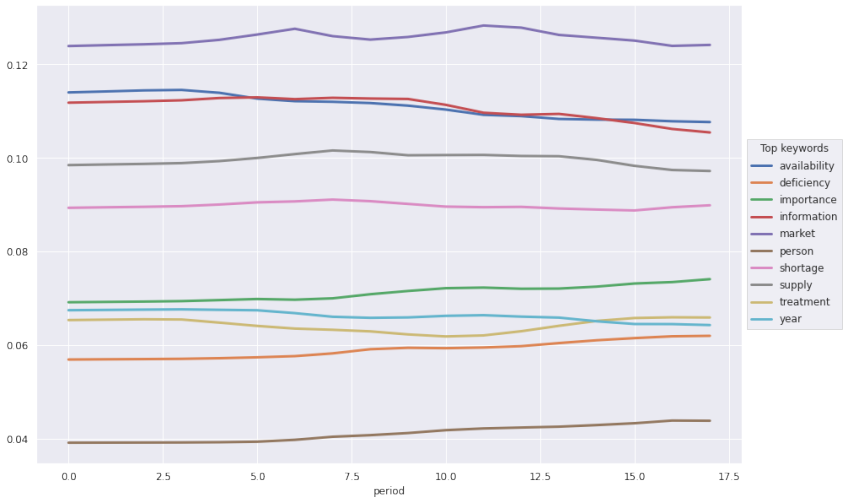


Figure 6.2: Example of Dynamic Topic Modeling visualization. It shows the evolution of the top words in a topic over time.

As an alternative algorithm to guided LDA, we tested dynamic topic modeling because it includes time. It shows the evolution of topics over time which could be interesting because the shortage topic is most likely strongly changing.
Figure 6.2 shows an example of a topic's top words evolution over time. The period refers to the month (Nov 19 until April 21). The y-axis displays a weight given to a word in that time period.

It can be seen that the **words do not change much over time**. It looked similar in other runs and other topics and also when looking at the topic evolution over time. Moreover, the dynamic TM implementation **uses a lot of memory** and could only be run with 5000 articles which is not really representative for the entire dataset. And finally, the expected **shortages were mostly not in the top words** of any topic. For these reasons, this idea was not further pursued.

6.1.2 Evaluation of the Article Selection

The selection of articles using the TM is compared to a keyword-selection **baseline**. All articles that mention a shortage indicator in the abstract make up the baseline. This method can also retrieve some false positives because some words are ambiguous and some shortage-related articles might be missed.
Subsequently, the **precision, recall and F-score** were calculated. This time, the number of articles is counted instead of the number of terms. The number of retrieved articles is equal to the size of the reduced dataset or of the baseline. The number of **relevant articles is given by all articles that contain a Covid-related shortage term in the abstract**. A keyword alone does not necessarily mean that it is actually mentioned in the context of shortages. Nonetheless, those articles are relevant since they give the context in which the shortage term is discussed. This can be translated to these two equations:

$$\text{Precision} = \frac{\text{number of articles that contain a shortage term and were selected}}{\text{number of articles that were selected}}$$

$$\text{Recall} = \frac{\text{number of articles that contain a shortage term and were selected}}{\text{number of articles that contain a shortage term}}$$

6.2 Experiments

Four experiments were done they tune the algorithm and answer the second research question. The first experiment compares TM trained on different parts of the article. The second experiment tunes the hyperparameters of the guided LDA model. The third one compares the selection of documents via TM to a baseline. That answers the question of whether TM or keyword-based search performs better at selecting relevant articles. The last experiment repeats the shortage-identification experiment of the first question only on the reduced dataset. The results of the experiment are compared to the previous results of the same experiment on the entire dataset. This will answer the part of the second research question asking what the difference in performance is.

6.2.1 TM on different Text Fragments

We tested alternatives to training a TM on the abstracts. The TM algorithm described in the methodology was applied on three different parts of the article: the **entire article**, only the **titles**, and the **sentences per abstract**. We trained a TM for each of the text fragments and the **abstract** method on the same sample of 10 000 articles. A short optimization of ten iterations was done. Ten models were built on the best parameter setting for each method. For each of the ten models, the F-score, precision, and recall for selecting the relevant articles were calculated as described above and the mean was taken. Table 6.1 shows the results and compares them to the keyword baseline on these 10 000 articles.

	Mean Precision	Mean Recall	Mean F-score
Baseline	0.34	0.38	0.36
Abstracts	0.43	0.31	0.36
Title	0.08	0.36	0.13
Entire Article	0.32	0.55	**0.40**
Sentences	0.08	0.36	0.13

Table 6.1: Alternative TM results on a sample of 10 000 articles

The table shows that **the more information is given, the better the model**. The title and sentence model score a lot worse than the baseline. It can be seen that TM on the entire article scores best in this small sample. However, **only about half of the data contains the text of the entire article**. Therefore, we kept the abstract method. For another dataset where more documents with the entire text are present, TM should be applied to the entire article. Another reason to only consider the abstracts is that it saves time and memory.

6.2.2 Model Parameter Tuning

In this experiment, the model parameters are tuned, namely the number of topics k, α, β, and the parameters for seeding: the seed_confidence and the number of seed terms x. Hyperparameter tuning for TM is recommended by several sources to ensure stability, e.g. [25, 100]. First, the experimental setup including the search space is described. Next, the results are reported and discussed. The last part analyses the influence of the model parameters on the model performance.

Experimental Setup

The TM algorithm was executed as described with different parameter settings. We used the package “Hyperactive” [101] to optimize the problem. A **Random Restart Hill Climbing Optimizer** with 8 neighbors was initialized. The Random Restart Hill Climbing Optimizer was chosen because the hyperactive library recommends it as “Good as the first method of optimization”. In addition, shc [100] compared search approaches to tune LDA parameters and recommend stochastic hill-climbing for small to medium size projects as well.
Five different samples of 10 000 articles were used to optimize the parameters with 20 iterations each. In each iteration, a new model was created with the parameters selected by the optimizer. Then, the C_v coherence and the F-score were calculated. The model was saved including the respective hyperparameters and evaluation values. The optimizer needs one value to optimize over. We tried different ones: coherence, the F-score, and a mixture of both. In the end, the **coherence and F-score were combined and equally weighted for the optimization**. The reason for not using the F-score alone is that it turned out to not always behave proportionally to the F-score of the evaluation. Sometimes it was low when the evaluation F-score was high (see section 6.2.2 for details). This suggests that the list of shortage indicators is not actually a good indicator of shortages.
Table 6.2 displays the **search space**. The number of topics k was set based on the size of the dataset in relation to the size of the part of the dataset which contains a shortage term. That was done to get an estimate of the number of articles to select. The result of that is around $\frac{360000}{116000} \approx 3$. Therefore, k was set relatively low from 3 to 10. For α, β and the seed_confidence the entire range was used. For x, any number of terms can be selected from the shortage indicators to seed with. A seed_confidence of 0 or an empty seed list leads to no seeding. Thus, the **standard LDA model is part of the parameter search space**.

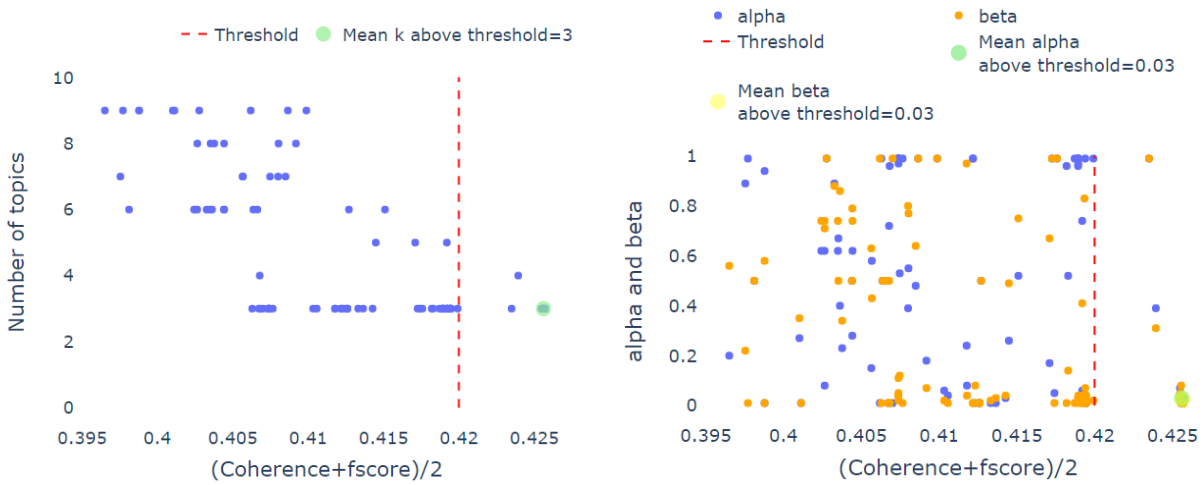
Parameter	Value Range	Step Size
k	3-10	1
α	0.01-1.0	0.01
β	0.01-1.0	0.01
seed_confidence	0.01-1.0	0.01
x	0 - 46 (number of shortage indicators)	1

Table 6.2: The search space to optimize the Topic Modeling problem

Due to the LDA algorithm being unstable, the allocation of documents can be different for the same parameters. Thus, the **average of the best models was taken as the final parameter set**. The best models were selected with a certain threshold. Moreover, the influence of the model parameters on the model performance is analyzed with a correlation matrix.

Results

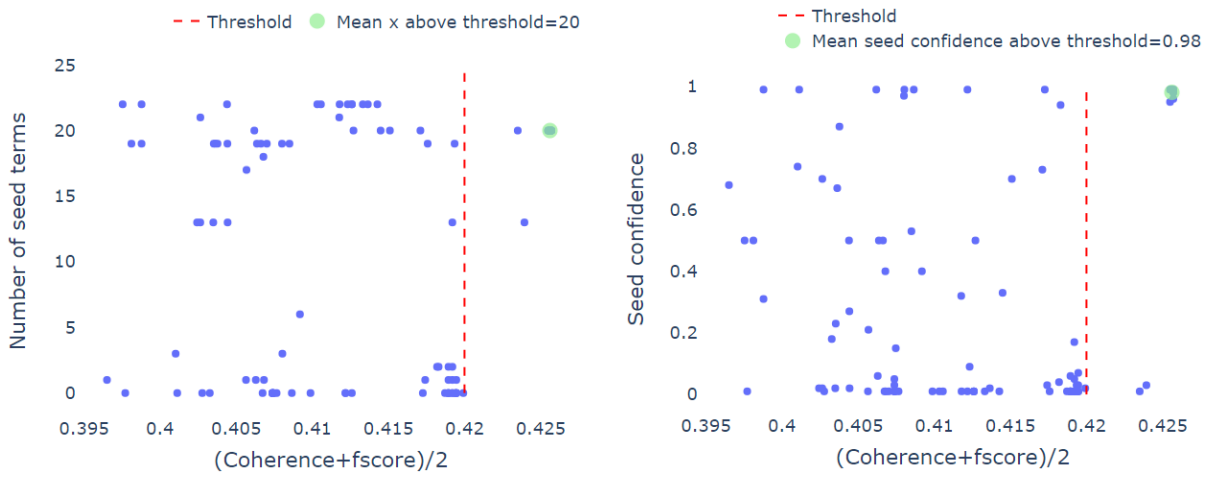
When looking at the models with the highest scores, we could observe a pattern. The models either had a very high seed_confidence (over 0.9) and very low values for α and β (less than 0.1) or a very low seed_confidence. Since the two groups of models have values on opposite ends of the scale, it does not make sense to average them all. There were more models with a high seed_confidence, therefore those were kept.



(a) Number of topics

(b) α and β

Figure 6.3: Plots of the models averaged sum of coherence and F-score vs model parameters



(a) Number of seed words x

(b) Seed confidence

Figure 6.4: Plots of the models averaged sum of coherence and F-score vs seed parameters

The model performances and the hyperparameters were plotted. The plots in Figures 6.3 and 6.4 show the values for the hyperparameters on the y-axis and the averaged sum of the coherence score and F-score on the x-axis. The red line in the plots is the **threshold value of 0.42**. The best models are on the right side of it. The average values of the parameters of the best models were chosen as final parameter set. That is for **k=3, α=0.03, β=0.03, seed_confidence=0.98 and x=20**. To get the best list of seed terms, the 20 most frequent terms in the best models were used. Those are *"logistics", "peak", "demand", "capacity", "shortage", "supply chain", "goods", "stock", "product", "reduction", "deficiency", "resource", "unavailability", "market", "supply", "scarcity", "manufacturing", "lack", "request", "price"*.

Influence of the Model Parameters on the Model Performance

This section analyzes the influence of the model parameters on the model performance by looking at their correlation. Figure 6.5 shows the correlation between the hyperparameters and the evaluation measures. **Both evaluations are reported, the one based on the topic terms and the one based on articles**. The scores that contain an "_eval" ending in the figure are calculated with the article evaluation. Precision, recall, and F-score without an ending are calculated based on the top terms in the topic and the list of shortage indicators. The "_eval"-scores are not used for optimization, only to show how the parameters influence the model performance.

Number of topics k. Some interesting correlations can be seen. k is negatively correlating with the coherence and the recall_eval and F-score_eval. That means, **the higher the number of topics, the lower the model performance**. Thus, a low k should be chosen. Nonetheless, the precision_eval rises with an increasing k.

α and β. α and β are less influential on the evaluation scores. However, there is a weak negative correlation between them and the evaluation scores with the "_eval"-ending. Thus, **α and β should be kept low.** Moreover, β and k are positively correlating. So a low k means a low β.

Number of seed terms x. Interestingly, there is also a weak negative correlation between α and the number of seed terms x. For a **low α, the number of seed terms x should be relatively high**.

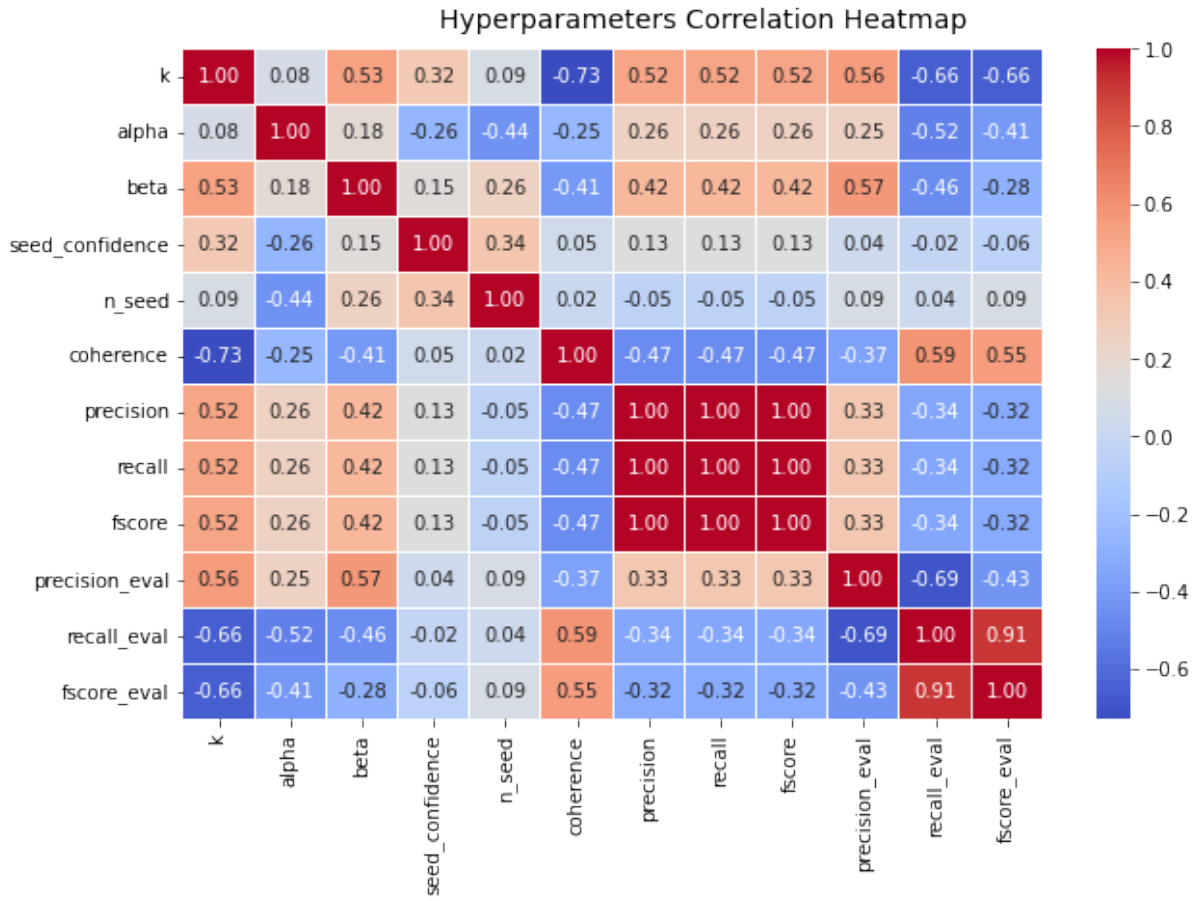


Figure 6.5: Correlation of the hyperparameters and evaluation measures

Derived rules. The following rules can be derived from that which should be true to obtain high evaluation scores based on the articles:

- high F-score_eval and coherence $\rightarrow$ low k, relatively low α and β
- low $k \rightarrow$ low β
- low $k \rightarrow$ relatively high x

The **rules are valid for the final parameter set**. k, α, and β are low and the number of seed terms is relatively high.

Coherence and evaluation measures. The coherence and the F-score_eval have a positive correlation. That shows that coherence is a good value to choose a parameter set. The higher the coherence, the higher the evaluation F-score. The recall_eval and precision_eval have an inverse relationship which shows that the models are not just random.

F-score based on topic terms. A positive relationship cannot be seen between the F-score based on the shortage indicators and the F-score based on the shortage articles. There is even a weak negative correlation between them. Thus, **the F-score based on shortage indicators and keywords is not a good estimation of the model performance.** This might be due to the shortage indicators not actually indicating shortages. Another explanation could be that the topic terms are not representative of the way how the model sorts the articles into topics.

It is also imaginable that the calculation is incorrect. The precision is based on the number of retrieved terms. Since all topics contain all terms with a different probability, the top 1000 terms per topic were retrieved. Thus, the number of top terms is constant which is usually not the case. This approximation to the actual precision might be the reason for the F-score being off. This can also be seen by the correlation behavior of precision and recall. They should be negatively correlating but they are perfectly positively correlating.

6.2.3 TM vs. Keyword Article selection

The article selection based on TM was evaluated against the keyword baseline as described in the methodology. Since the algorithm is not stable, the evaluation should not be based on just one LDA model. It could be a coincidence that the model performs better or worse than the baseline. To get a more stable estimate of the performance of the TM selection, the **average scores of several models** were taken. 100 LDA models based on the final parameter set were created and trained on the entire CORD-19 data.
Each model was evaluated individually. First, the entire dataset was classified. Only the articles were kept that were part of the shortage topic (topic 0, the seeded topic). The resulting dataset of relevant articles was evaluated with the precision, recall, and F-score based on the shortage articles as described in the methodology. Finally, the mean and standard deviation of the values was taken.

	Precision	**Recall**	**F-Score**	**Size**
Baseline	0.33	0.39	0.36	134 250
Mean TM	0.51	0.38	**0.44**	88 450
Standard Deviation TM	0.00004	0	0.00002	38

Table 6.3: Baseline and TM performance

Table 6.3 shows the results of that and the ones of the baseline. The baseline was created by selecting all articles where the abstract contains a shortage indicator. It can be seen that the **TM performs better at selecting the relevant articles than the baseline** keyword selection. On average, the F-score of the TM selection is 0.44 which is 0.08 more than the baseline. Since the standard deviation is so small, the **TM algorithm seems to be more stable than expected**. This might be due to the seeding. This positively answers the research question if TM improves the selection of relevant articles in comparison to keyword-based search. The topic model which was closest to the average performance was chosen to select the relevant articles for further processing.

6.2.4 Identifying Potential Shortages

We executed the same experiment as described in 5.2 on the reduced dataset. Table 6.4 shows the cumulative results of the shortage identification for the entire dataset and the reduced one. Almost the same proportions between the F-score, precision, and recall as for the entire dataset can be observed. Again, the **highest F-score of 0.065 was reached at 100 terms**. In addition, the recall increases with a decreasing precision for more terms. A difference is that the F-score is higher for 1000 terms than for 500 terms because the recall still heavily increases while the precision only slightly degrades.

Top Terms	Precision	Recall	F-Score	Number of Types	Time in seconds	Origin	Results of
100	0.073	0.22	0.109	6	1383	entire dataset	RQ1, ch. 5
500	0.051	0.50	0.092	12	1366		
1000	0.043	0.63	0.08	13	1173		
100	0.043	0.13	0.065	8	295	reduced dataset	RQ2, ch. 6
500	0.030	0.36	0.056	11	290		
1000	0.030	0.53	0.057	13	290		

Table 6.4: Evaluation measures for all methods together on the entire and the reduced dataset

Table 6.5 shows the loss in performance by reducing the dataset and the gain in time. The last row shows the mean relative loss. It is calculated by dividing the difference averaged for different numbers of top terms by the values of the entire dataset. **Precision, recall, F-score, and time decreased**. The F-score degraded by 36% on average. The reason for that is that some relevant terms are not part of the dataset anymore or are just not found by the methods. The processing time decreased by 78% on average. This **suggests that the selection is not really domain-related**.

Top Terms	Precision	Recall	F-Score	Time in seconds
100	0.03	0.08	0.044	1087
500	0.02	0.14	0.036	1075
1000	0.013	0.1	0.023	883
mean relative loss	37%	27%	36%	78%

Table 6.5: Loss of Evaluation measures of the entire dataset and the reduced dataset

The **dataset was reduced from around 360 000 to around 88 000 articles**. That is a loss of about 76%. If a linear relationship is assumed then the evaluation measures should reduce by the same factor. This is true for the processing time. However, the relative loss in precision, recall, and F-score is significantly smaller. That suggests, that the TM actually selects relevant articles. More likely is that the relationship is not linear. The number of unique terms probably does not decrease in the same proportion as the number of articles.
Even though the evaluation measures decreased, **the number of different types of shortages stayed similar**. This means that the resulting list of suggested terms stayed diverse.
The results of each individual method are in appendix C.2. **Some of the individual methods improved** in performance by selecting only relevant articles. All of them are frequency-based. The fact that there are more relevant terms in the most frequent terms of the reduced dataset than the entire dataset shows again that the **TM produced a dataset which is somewhat focused on the domain of shortages in SCs**. However, most of the methods still decreased also some of the frequency-based ones.
Although a large loss could be observed, the **reduction of the dataset is still helpful for several reasons**. First, the processing time for the shortage identification decreased by a lot. Second, the reason for selecting only relevant articles was to create a domain-related dataset. The domain dataset is necessary to build a domain-specific KG. By discarding irrelevant articles, less noise is included in the KG and it stays focused on shortages in SCs. Third, when creating

and using the KG, a smaller dataset saves process time and memory. Moreover, the KG does not become unreasonably large which is an advantage for future usage of it. Furthermore, this **saves a lot of human effort** if the alternative is an expert manually selecting the articles. Finally, the loss in performance for the shortage identification might be due to the relatively poor performance of the TM algorithm in selecting relevant articles. Assumably, a perfect selection of the articles would improve the performance since noisy articles are left out completely.

6.3 Summary of the Chapter

This chapter explained how relevant articles are selected with TM. A guided LDA model using the shortage indicators was used. Two experiments were executed to tune the model. Two more experiments were executed to answer the research question and evaluate the selection of the articles. The result of the first experiment was that training the model on the abstracts rather than other parts of the documents works best for the dataset at hand. The second experiment found a good set of model parameters using optimization. The analysis of the correlation between the model parameters and the evaluation measures showed that the estimation of the model performance using the F-score on the topic terms is not a good approach. Another experiment concluded that the article selection with the TM outperforms the keyword baseline which positively answers that part of the research question. To answer the second part of the question, the shortage-identification method was repeated on the reduced dataset. There was a loss in performance but the dataset is still shortage-related. For that and other reasons, the approach is useful

7 CREATION OF A DOMAIN KNOWLEDGE GRAPH TO IMPROVE SHORTAGE IDENTIFICATION

The methods described so far are effective, as they identify some of the shortages within the pandemic. However, there are two drawbacks to these methods. First, they **do not find all potential shortages**. Second, they find many **terms that cannot be in shortage.** Solving the first problem is difficult because it is hard to say when the list of shortages is complete. There can always be additional shortages or synonyms that were not part of the ground truth. The second problem originates from the fact that solely statistical measures, such as occurrences and co-occurrences, and time were used. These methods also retrieve terms that are not a product or an object and can thus not be a shortage. Including semantics could mitigate this problem. Therefore, this research proposes to **encode the lexical knowledge of the text in a KG and make use of it when identifying potential shortages**.

The KG can help to filter the predictions for **only terms which can be a shortage**, i.e. products or tangible objects. For example, the term “patients” appears very often and the TWS might retrieve it. The KG, however, knows that patients are a group of people and not an object. Therefore, it would not be retrieved anymore.

Moreover, the KG only contains entities. There are many noun phrases that are not an entity and **only entities can be in shortage**. Sorting out these noun phrases already improves the prediction performance. In addition, the relations in the KG can be helpful. If two entities are directly related, and one of them is identified as a shortage (indicator), the other entity might be relevant as well.

Another problem solved by the KG is **entity resolution**. For instance, if one article is referring to masks as “face masks” and another one as “protective masks” the TWS would weigh them as separate terms but the KG would resolve them to the same entity. The occurrences and co-occurrences are thus combined and closer to their actual frequency. All in all, the **KG helps to remove noise by including semantics in the prediction.**

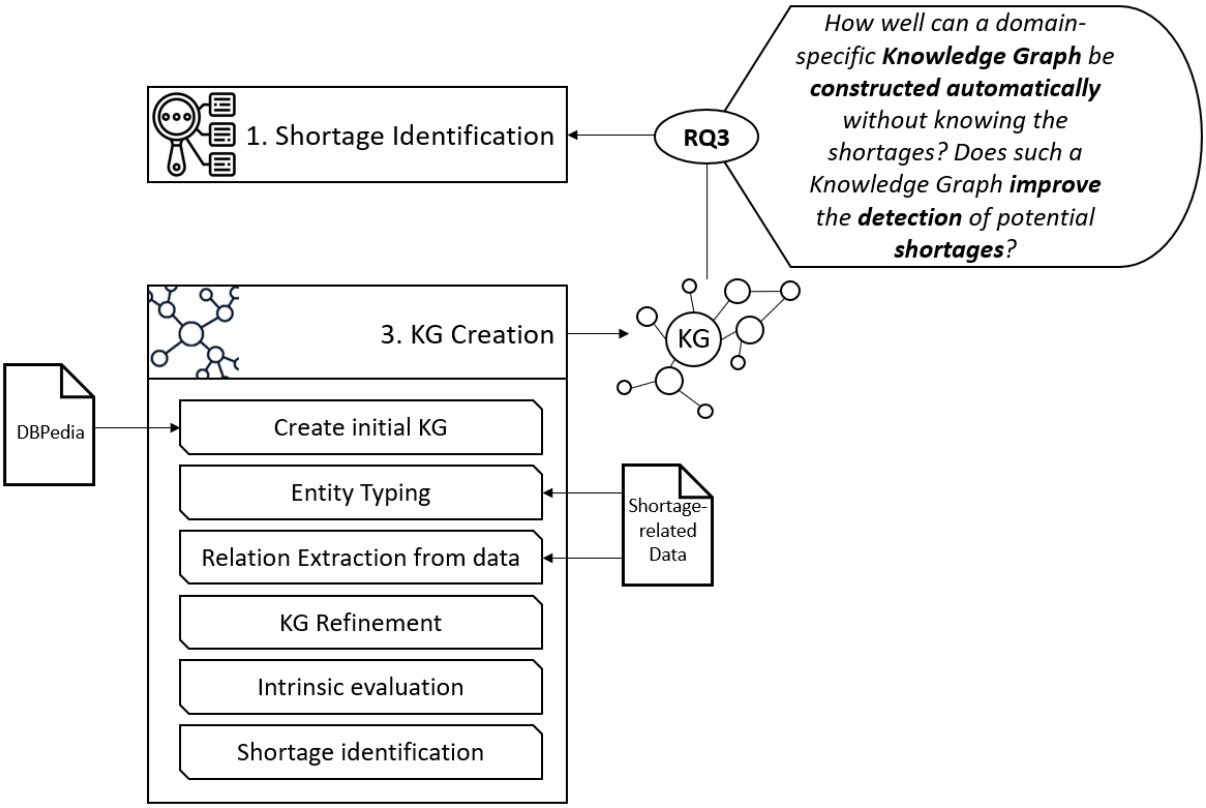


Figure 7.1: Overview of the methodology and experiments of this chapter

Figure 7.1 gives an overview of the chapter. The creation of the KG is described in the methodology. That includes creating an initial KG, entity typing, relation extraction, and refining the KG. The experiment section can be split into three main parts. First, there are some experiments on the individual creation steps. Second, the KG will be evaluated intrinsically. The intrinsic evaluation should answer the first part of the last research question of *How well can a domain-specific Knowledge Graph be constructed automatically without knowing the shortages?* The final experiment will repeat the shortage-identification method but this time with the help of the KG. The performance will be compared to the previous two results of the same experiment to answer the second part of the question of *Does a domain-specific Knowledge Graph improve the detection of potential shortages?* Moreover, the best ensemble of methods is identified. Finally, the list of retrieved shortage suggestions is analyzed.

7.1 Methodology

When reviewing the literature, no domain-specific KG was found about Covid-19 shortages or similar. It is thus very difficult to find an overlap between the triples extracted from text with another KG. Therefore, a new KG has to be constructed. The steps are the following. First, an initial KG is extracted from DBpedia. Then entities in the data are typed and added to the KG. Thereafter, relations are extracted from the data and added to the KG. That KG is refined with different approaches. Finally, the TWS are adjusted to make use of the KG and extended with some more methods.

7.1.1 Initial KG

Starting from an initial KG and enhancing it has the advantage that the relations are already verified and can be assumed correct. Additionally, it can be seen as a seed for a specific

domain. The easiest would be to start from an existing domain-specific KG. As the Covid-19 KGs focus on other parts of the pandemic, they are not very helpful for SCs. Thus, as already concluded in chapter 3, we extract a **domain-specific subgraph from DBpedia as initial KG**. In order to extract a domain-specific subgraph from DBpedia, these steps are taken:

1. Link a list of domain keywords to DBpedia
2. Extract and combine all triples to the linked entities to form the initial KG
3. Add synonyms to the KG
4. Clean the initial KG (see section 7.1.5)

Entity Linking

In this thesis, the **domain keywords are the shortage indicators**. They are used instead of all shortage terms because the goal was to automatically identify shortages and manually inserting known shortages in the initial KG will bias the results. Some of the indicators might not be an entity in DBpedia and cannot be linked.

Linking method. We tested two ways to link the list of terms to DBpedia. One uses the annotation tool DBpedia Spotlight which automatically finds and links DBpedia resources in a text [102]. The other one looks up the terms directly by inserting the term in the DBpedia URI. Out of the 46 shortage indicators, only three terms were linked using Spotlight and 42 terms were linked using direct matching. We chose **direct matching** as it linked a lot more terms. This might be because the list is human-made which means that most of the entries are already the proper term for it. The Spotlight linking does not work well for single words, as the context is used to annotate the entities.

Pre-pandemic DBpedia. The KG is supposed to be built **without any knowledge about Covid-19 shortages**, so that it can support the shortage identification without being biased. For that reason, the triples cannot be extracted from the current version of DBpedia. DBpedia is based on Wikipedia and throughout the pandemic, many Wikipedia entries about Covid-19 have been created. For example, the extensive entry “Shortages related to the COVID-19 pandemic” [90] makes up many triples within DBpedia. The ground truth is largely built upon that. To avoid this, a **DBpedia version before the pandemic** should be used. DBpedia is constantly updating itself, so it is not possible to store every version of it. There are automatic monthly dumps but they are difficult to access. de Sompel et al. [103] created a tool called “Memento” which offers some static DBpedia dumps. The version is from **April 2016**.

Implementation. The implementation of the entity linking is split into two parts. First, it has to be checked if the term is an entity in DBpedia. For that, a URI with the capitalized term is created and looked up in the current DBpedia (“http://dbpedia.org/resource/” + “ENTITY_NAME”). The current DBpedia is used because it only returns an “okay” status to a request if the entity exists. The memento version always returns an “okay” status even if there is no entry for a term. The second step is the triple extraction from the 2016 DBpedia. The URI of the linked entities is simply adjusted by changing “resource” to “page” and adding the prefix “http://dbpedia.mementodepot.org/memento/20160415000000/”. The content of the website is requested and returned in HTML format. The triples of a DBpedia resource are all stored in one table. The triples can easily be extracted from the returned HTML table.

Adding Synonyms

The initial KG was further enhanced with synonyms. This was done because synonyms that are used in text for an entity should not be missed. Wordnet [104] is a large lexical database which contains semantic relations of many common words. For each entity in the KG, the synonyms in Wordnet were looked up. They were added to the KG with the relation “same_as” connecting the original term to each of its synonyms.

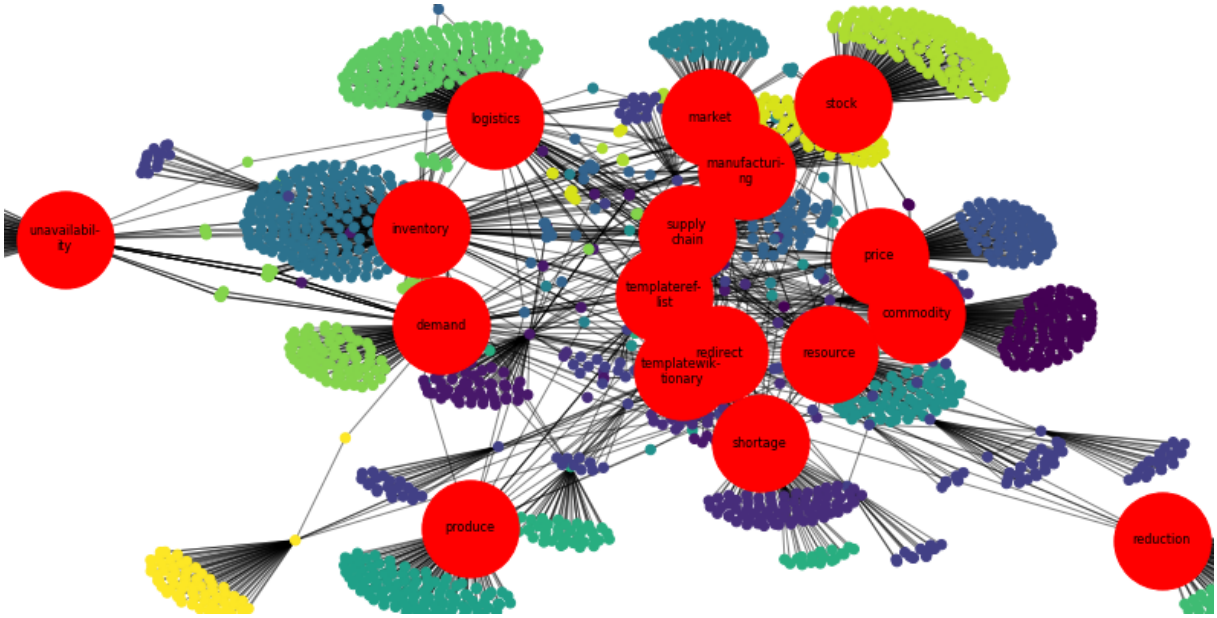


Figure 7.2: Initial KG derived from DBpedia 2016 highlighting the most connected nodes

Figure 7.2 displays the initial KG. The large, red entities with a label are the most connected nodes. A lot of them are part of the shortage indicators. At a first glance, the graph is about the SC domain and, thus, the method to derive the initial KG seems to be somewhat effective.

7.1.2 Labeling Entities

There are two parts to identifying something as a shortage. First, it should be a **tangible object or product which can be in shortage**. That is necessary because there are other terms that are often mentioned in the context of shortages, such as “demand” or “missing” which can be distinguished from the thing in shortage by not being a tangible object or a product for sale. The second task is finding out that it is actually scarce at a certain point in time. That will be done in the shortage-identification experiment. Next to “object” other types can be useful knowledge for the graph too because it categorizes entities. For instance, reasoning can make use of it. Therefore, other types are added to the KG in addition. The following will describe how the entities are labeled, which forms the next step in the construction of the KG.

Entity Typing

We use entity typing to assign a type to noun phrases in text. The background chapter 2 compared it to the more common approach NER. Whereas any term can be classified using entity typing, only the terms that the NER model discovers are typed by NER. Since unknown or uncommon entities should be typed as well, entity typing was chosen.
There are several pre-trained models that type entities. Some models also include the types “object” or “product” which can be used in shortage detection. Comparing different models to

label the type “object”, the **Luke entity classifier** [37] seemed to be the best fit. It is a pretrained model that does not need to be finetuned to the data. The types are *entity, event, group, location, object, organization, person, place, and time*.
The Luke classifier **needs context around a term to assign a type**. Therefore, it is not possible to simply classify all entities present in the KG. Since most entities in the KG will originate from the reduced CORD-19 corpus, it is used as context information around the entities.
The most intuitive approach would be to simply extract all sentences that mention an entity, apply the classifier to each sentence and keep the most common type. However, that would need to be repeated for each new entity found in text. To avoid this, each possible entity candidate in the data is typed. We consider **all noun phrases in text as entity candidates**. The result is **a dictionary with a type to each noun phrase in text**. For each new entity in the KG, the type can be looked up in that dictionary.

Implementation

The following steps are taken to create the entity type dictionary:

1. Split each document in the reduced dataset into sentences
2. **Apply the Luke classifier to each noun phrase within a sentence**:
   (a) Extract the start and end index of each noun phrase in the sentence
   (b) Insert the sentence and the indices into the tokenizer
   (c) Insert the output of the tokenizer to the entity classifier which assigns a probability to each of the nine types
3. Determine the **type with the highest probability**
4. Only keep the type if the probability is **above a threshold** *
5. Group all noun phrases and their types (can be different depending on the context)
6. Only keep the **most common type** per noun phrase **
7. Only add a noun phrase to the dictionary if the most common type was predicted more than five times and more often than all the other types together ***

* The threshold is determined in an optimization experiment, see 7.2.1. A threshold is set because the model always predicts something even if the model is uncertain to what class a noun phrase belongs. In that case, it is better to not add a label rather than to add false information.
** For each entity, only one type is kept because different types for the same entity can be contradicting. Furthermore, not all of the predicted types are necessarily correct or they are but only in a certain context. Thus, a majority vote is taken to strengthen the prediction and reduce the error.
*** These conditions should ensure that only likely types for entities are added to the KG and only little false information.
Finally, for each entity appearing in the KG, the **type is added with the relationship “luke_type”**. That happens twice, once after the initial KG was created and once after the relations were extracted from text.

7.1.3 Relation Extraction from Text

At this point, an initial KG has been created via entity linking to DBpedia. It was enriched by synonyms and entity types from the Luke classifier. The next step is enriching the KG with triples from textual data. For that, RE is applied to the selected articles of the CORD-19, and the newly found triples are added to the KG.
We use **open RE** to extract the relations. As already concluded in the background chapter (see 2), it has some advantages over its alternative "relation prediction". Relation prediction predicts only certain predefined relation types. This over-generalizes the data by trying to fit each entity pair into one of these categories. Open RE does the opposite by extracting the relations directly from the data. This has the disadvantage of becoming too detailed and losing the chance to connect knowledge by generalizing it. To tackle that problem, the triples are reduced and summarized after RE.

Implementation

Open RE is implemented with the help of the **CoreNLP** library [105] and verb extraction. The CoreNLP pipeline is structured in the following way. First, the text is tokenized, split into sentences and Part-of-Speech (POS) tagged. Second, dependency parsing splits the sentences into segments. Finally, each clause is converted to an open RE triple.
CoreNLP requires some **parameters**. The annotator is set to only "openie", which refers to open information extraction (the other necessary annotators are included automatically). The number of entailments determines the variants of triples. The CoreNLP documentation recommends it to be set between 100 and 1000. On a small sample of the data, we extracted the relations for different numbers of entailments. The minimum number of entailments was chosen where no relevant triples were missed in comparison to the triples obtained with 1000 entailments. The relevant triples are the ones where one of the entities contains a shortage term or indicator. That was reached at 600 entailments. The other parameters are left at their default values because they do not seem to improve the results. The only processing applied on the triple is that only the noun phrases and prepositions are kept within the entities. The KG is enriched by the resulting triples.
CoreNLP does not always return a triple for a sentence. In that case, **verb extraction** is applied to not miss any information. That is implemented as follows. For all pairs of noun phrases within a sentence, the verbs in between the two candidate entities are extracted via POS tagging. The number of characters in between the two noun phrases needs to be lower than 50 to increase the likelihood that the noun phrases are directly related. For each discovered verb, a triple is added to the KG. A downside of this approach is that, unlike CoreNLP, it can create triples that are grammatically or syntactically incorrect.

Removing Triple Variations

CoreNLP creates a large number of triples where several are semantic duplicates. This happens because for every sentence fragment, several options are extracted which only differ in a few words. That can mean that two triples have the same meaning but one contains more details than the other (e.g. "patient $\rightarrow$ is_in $\rightarrow$ hospital" and "covid19_patient $\rightarrow$ is_in $\rightarrow$ hospital"). It can also mean that two triples contain the exact same words but, for example, once a word is part of the relation and once part of the object (e.g. "patient $\rightarrow$ is_in $\rightarrow$ hospital" and "patient $\rightarrow$ is $\rightarrow$ in_hospital"). Another variation is the same head and tail but a different relation.

Entity variations. To solve this problem, the triples are grouped by head (or tail) and relation, and only one tail (or head) is kept. Two options were considered: keeping the longest variation and keeping the shortest variation that is still a noun phrase. The longest variation has the

most detail which prevents oversimplification and false facts. Nevertheless, more details also mean more irrelevant information. The shortest variation has the advantage that more entities can be resolved to one. The shorter version found fewer shortages and sometimes parts in the middle were left out, e.g. “levels family violence China” became “levels China”. Considering the total number of heads and tails together, the longer and the shorter version had a difference of less than 1%. This shows that the expected advantage of reducing the KG’s complexity when using the shortest noun phrase was not really present while the expected disadvantage of losing relevant information could be found in the data. Thus, we decided to only keep the **longest entity variation**. This is first applied on the tail and after that on the head.

Relation variations. After reducing the triples with the same head (or tail) and relation to only one triple, there were still some semantic duplicates. Many triples had the same head and tail but a different relation. It could be observed that longer relations contain more details but those are often irrelevant. In many cases, there was an adverb with the verb which did not change the meaning of the relation, e.g. “captures” and “successfully captures”. Moreover, shorter relations can mean fewer relation types. For these reasons, the **shortest relation variation** was kept for the same head and tail.

Initial KG as Seed

To add the final triples to the KG, the initial KG enhanced by the entity types is used as a seed. This means that the extracted **triples are only added to the KG if at least one entity is already in the KG**. Partial matches are considered as well. This way, the KG is enriched instead of just extracting unconnected bits of information. Moreover, the KG remains focused on shortages in SCs which contributes towards the creation of a domain-specific KG. However, important information might be missed if it is not directly related to the initial KG. In an experiment, other approaches which connect the extracted triples to the initial KG are compared to connecting it on entity level (see 7.2.1).

7.1.4 KG Completion: Enhancing the KG

RE is the last part of the knowledge creation. After that, the KG is refined by error detection and completion. In this thesis, only KG completion methods are applied because error detection is quite difficult to implement. The KG completion methods are split into two parts: enhancements and reductions. Section 7.1.5 describes the reduction methods to remove noise and irrelevant information. This section focuses on the enhancements to add missing information.

Enhancing with Super-classes

There are many synonymous entities in the KG but also many which are a more **specific version of the same object**. For example, “N95 mask” and “FFP3 mask” are both a mask but specific versions of it. Thus, the two versions should not be resolved to one entity. Instead, they can be **grouped into classes** so that they can be connected without deleting specific types of it. This can, for example, be helpful when detecting potential shortages with frequency-based methods. Instead of counting the specific objects, the super-classes of all objects can be counted. As a result, differences in frequency will be more clear because the counts of objects with many sub-types would increase. This method can be seen as a kind of entity resolution.

Extracting the classes. The first step is identifying the classes within the KG. We tested different approaches to creating the super-classes automatically and decided on the extraction of the **root of the entities**. The root makes up the meaning of a noun phrase and leaves out the

words which are only describing it. For example, in “face mask”, “mask” is the root, and “face” is only making the root more specific. When extracting noun phrases with Spacy, it is possible to extract the root from each noun phrase.

Adding the classes. Not every extracted root should be considered a class as some roots might only be used in one or a few entities. Since this should enhance the KG and not create more noise, adding classes with only a few elements is not very helpful. Moreover, adding too many super-classes might generalize the KG too much. Therefore, the **number of super-classes should be reduced**. A root is only kept as a class if it is the root to at least eleven different entities and if it has more than two letters. In the end, the super-classes are added to the KG as triples. The head of the new triple is the respective originating entity, the relation is **“subclass_of”** and the tail is the name of the root.

Repeating Enhancements

The enrichments which were used to build the KG can be repeated. Those are entity typing, entity linking, and RE. An experiment evaluated the gain of repeating each of the enhancements, see 7.2.2. Entity linking extracted too many triples and is not kept. The other two methods are used to complete the KG.
After the KG was constructed, the **entity types** from Luke are added again using the developed dictionary of entities and types. Even though the dictionary remained the same, this step still adds new entities to the KG because there are more entities in the KG now than before.
The KG can be extended by **entity linking** to DBpedia, the same way as it was done for the initial KG. The only difference is that this time all the entities in the KG are considered instead of only the shortage indicator list. This helps to enrich and link the new entities found in text with the DBpedia knowledge. Nonetheless, this could potentially expand the KG too much and take it off from the intended domain.
The **relations can be extracted** again from the same dataset. This extracts more information than the first time because entities that are not directly related to the initial KG but are still relevant to the domain, may not have been found. Seeding with the enriched KG and extracting the relations again, results in entities that are related to the initial KG in second degree. Again, the new entities have to contain a KG entity. Moreover, additional relations between already found entities could be found. Nevertheless, relevant entities can still be missed if they are not related directly or in second degree in text to the initial KG. In addition, the KG could become too large and would not be domain-specific anymore when including almost all the information given in the dataset.

7.1.5 KG Completion: Reducing the KG

The second part of KG Completion is reducing the KG. To make the content more concise, two KG cleaning methods are applied. One is the cleaning method which mostly cleans the text in the KG with common NLP preprocessing methods and sorts out unwanted triples via string matching. The second more advanced method searches and removes semantic duplicates in the KG.

Cleaning the KG

To clean the KG, a function was created which is applied to the KG after every addition. That contains several steps:

1. Remove characters from each part of the triple that is not a letter, a digit, or a space
2. Remove entities which are only a number or only one letter
3. Replace spaces by underscores
4. Remove entities longer than six words
5. Remove direct matches to the list of paper terms and Covid-19 synonyms (see appendix B.1) *
6. Lemmatize each word within an entity
7. Lemmatize the relations using the “verb” POS tag
8. Remove duplicate triples
9. Remove triples with the same subject and object (self-reference)
10. Remove certain DBpedia relations[1] **
11. Remove DBpedia triples where the object contained a common file ending [2] **

* The Covid-19 synonyms and paper terms are removed to avoid enlarging the KG with triples where almost every entity is related to one of these terms. However, only complete matches are dropped because partial matches can also contain valuable information.
** The DBpedia relations and triples are removed because they return triples that should not be part of the KG. Those are selected because they appear quite often and either refer to a long text, or the same entity in another language or an image name, or a numeric value like an id. Some of the objects are just names of files.

Removing Semantic Duplicates

The cleaning method and also the removal of triple variations with a common part already reduced the duplicate triples quite a lot. However, semantic duplicates of triples were not yet considered. They have the same meaning but synonyms or different structures are used. To do this, two libraries are leveraged, namely AmpliGraph [86] and Dedupe [50]. AmpliGraph is based on a KG embedding and Dedupe needs to be finetuned with very few human annotations.

KG embedding. AmpliGraph uses an embedding for deduplication. For that, we replace all relations with “related_to” because the relations do not carry much information. Considering all relations as the same can give an advantage for the embedding because less information needs to be encoded. This way the chance of finding duplicates is higher. Moreover, any term related to another term can be relevant no matter the relation. Next, the KG is split into a train and a test set. The embedding algorithm is HolE with its default parameters. The algorithm was chosen because it performed best in most of the performance experiments which AmpliGraph published in their documentation [106].

[1] owl:sameAs, dbp:wikiPageUsesTemplate, dbo:abstract, rdfs:comment, prov:wasDerivedFrom, dbo:wikiPageID, dbo:wikiPageExternalLink, dbo:wikiPageRevisionID, dbo:Thumbnail, dbo:Depiction, dbo:Image, dbo:Genre, dbo:wikiPageLength, dbo:wikiPageInterLanguageLink

[2] pdf, jpg, svg, ppt

AmpliGraph. The “find_duplicates” function can be applied to the trained embedding. The parameters are mode (entity, relation, or triple), a distance metric, the expected fraction of duplicates, and a tolerance threshold. Since the relations are all the same, it does not make sense to look for duplicate relations. For duplicate entities, only very few were returned with a low tolerance, and with a higher tolerance entities were returned which were no duplicates. Thus, **only duplicate triples** are looked for. The duplicates are removed from the KG and the longer triple of the set of duplicates is kept.

Dedupe. A second deduplication method is applied after that. The library pandas_dedupe [107] is a wrapper around the Dedupe library to make it easily applicable for pandas data frames. Before it performs the deduplication, it asks for annotations. Dedupe gives examples of possible duplicate triples which have to be annotated as correct, incorrect or unsure. The minimum for that is ten positive and ten negative examples. Based on that, the algorithm adapts before it is applied on the entire KG.

7.1.6 Intrinsic Evaluation

Evaluating a KG is quite difficult for several reasons. Some of them are: it is very large, it is difficult to measure if a triple is relevant, it is difficult to determine if a triple is a true fact and it is hard to say if a KG is complete within its domain. The best method is to let a human expert evaluate the correctness of triples for a part of the KG and calculate the accuracy from that. This could be done in future work. Here, the KG is evaluated in four ways. After each KG creation step, the **domain affiliation** is calculated by the overlap with the ground truth shortage lists. For the final KG, the **most common entities and relations** are analyzed. In addition **k-fold cross validation** is done. Those are intrinsic evaluations. The extrinsic evaluation of it is the evaluation with the use case which will be described at the end of this chapter. Thus, the results of the **shortage-identification experiment can show the usefulness of the KG** for its dedicated case. The details of the evaluation and the results of it can be found in the experiments section.

7.1.7 Identification of Potential Shortages with the Knowledge Graph

With the help of the developed KG, the TWS over time are executed again. They are still based on the selected articles of the CORD-19 dataset. The only difference is that the methods are altered to make use of the KG and some extra methods are added. The evaluation works the same as before. The following will describe how the TWS are adjusted.

Term Weighting Schemes with the KG

The previous version uses the noun phrases per abstract. When using the KG, instead of that, the **entities within an abstract** can be used. So only the noun phrases are kept which are an entity in the KG. Optionally, the filtering can be further narrowed down by only considering **objects within an abstract**, i.e. entities which are typed as an object in the KG. Anything counts as an object which is directly related to the entities: *object, resource, product, component, or commodity*. Additionally, **entity resolution** can be applied by replacing an entity with its superclass if there is one. For instance, all types of masks would be resolved to the entity “mask”. Each of the shortage-identification methods includes extra parameters that enable these changes. One parameter decides whether to use all noun phrases, only noun phrases in the or only objects. Another one decides whether the terms are replaced by their subclass or not.

Word Embedding

The word embedding is created the same way as before only now **based on the entities** in the KG. It is not based on the objects or the replaced by super-classes. Moreover, it is **based on the entire dataset** instead of the reduced dataset. The reason for these things is that there is more diversity in the entire dataset and when including more terms. That makes finding similar terms with the embedding easier because the semantic differences between terms are larger. Furthermore, a bigger training set makes the word embedding more robust because per term, more sample usages are given.

KG Neighbor Occurrences

One of the extra shortage prediction methods is the KG neighbor occurrence. It is based on the fact that two entities are related, no matter what the relationship is. If an entity is a shortage term or indicator, it could mean that all entities directly related to it, are also shortage-related as they must have some semantic connection. In principle, this method works the same way as the context occurrence method. The difference is that instead of the neighboring words in text, **the neighboring entities to keywords in the KG are selected**.
For each abstract, only the noun phrases are kept that appear in a list of keywords, for example, the shortage indicators. The list of the remaining noun phrases is extended by their direct neighbors in the KG if they appear in the graph. The result is a dataset of lists of entities per abstract where each list contains the entities that occurred within the list of keywords and their KG neighbors. Finally, to get the top n terms, the term frequency or TF-IDF is applied on the filtered and enhanced dataset of lists of terms.

Link Prediction

The next new method uses the KG embedding again (see section 7.1.5 for implementation). It performs link prediction, also called reasoning. On the **trained embedding, the method "query_topn"** can be called to perform link prediction. A relation and a head/tail need to be given as input. The relation is "related_to" for all triples within the embedding which simplifies this task. The method returns the top n entities as tail/head of the triple with a rank. These triples can already be existing in the KG but also new relations are returned where the predicted entity is always part of the KG.
For each input keyword, e.g. the shortage indicator, the top n subjects are retrieved in this way. Since the subjects of a triple are more often specific and the objects more general, it makes sense to query for the head of the triples. For each predicted head, the ranks are summed up if they were predicted more than once. That means if two different terms predict the same head, the ranks of the two are summed up and the respective term ranks higher. Finally, the top n entities sorted by the summed ranks are returned.

7.2 Experiments

The KG was created as described in the methodology. Some experiments were done to fine-tune some of the methods, evaluate the approaches and the KG and answer the last research question: *How well can a domain-specific Knowledge Graph be constructed automatically without knowing the shortages? Does such a Knowledge Graph improve the detection of potential shortages?* First, the experiments regarding the creation of the KG will be explained. Then, the intrinsic evaluation of the KG is described. The final part is about the shortage identification with the KG.

7.2.1 KG Creation

We executed some experiments to strengthen the design choices taken to create the KG. First, the initial KG based on DBpedia from 2016 is compared against the one from 2022. Second, the threshold for entity linking is determined empirically. Finally, the different options to use the initial KG as a seed for RE are explored and compared.

Initial KG based on DBpedia 2016 compared to DBpedia 2022

The initial KG was built based on the 2016 DBpedia version because the 2022 version contains shortage information which could bias the KG. The following will show that the 2022 version of DBpedia actually contains shortages, which supports that claim. To compare that, a **second initial KG was built based on the 2022 DBpedia**. It is also based on the *shortage indicators* which were first linked to DBpedia and then all triples were extracted.
The biggest difference is the **size of the graphs**. Whereas the 2016 KG only contains 2794 triples, the 2022 graph contains more than ten times as many (32 964 triples). Both graphs contain similar topics, which is probably due to the fact that 38 shortage indicators are part of each of the graphs. However, the 2022 version contains a lot more details and links to other subjects.

Term	2016 triple counts	2022 triple counts
producer	38	7282
manufacturer	7	6327
peak	86	2898
product	78	1013
scarcity	49	260
shortage	103	218
hoarding	42	132

Table 7.1: Counts of some of the shortage indicators within the 2016 and 2022 initial KG

Table 7.1 shows **how many times some of the shortage indicators appeared** in each of the KGs. The big changes in ”manufacturer” and ”producer” could be explained by the addition of specific names of companies. Similarly for the increase in ”product”, a lot of specific product names were added to DBpedia. For the term ”peak”, many names of mountain peaks in the world appear in the 2022 DBpedia. But also the terms indicating a shortage like ”shortage”, ”scarcity” and ”hoarding” increased, which could mean that specific shortages related to Covid-19 were added.
This might suggest that the 2022 KG is more useful because it contains more details. Nevertheless, the concern that it contains **too much information about the Covid-19 shortages** could be confirmed with the data. From the shortage term list, twelve terms are part of the 2016 KG: *hamilton, johnson, pfizer, conductor, semiconductor, copper, coal, petroleum, drug, sugar, caffeine, and coffee*. The 2022 version contains in addition to that *abbott, shield, cdc, naat, mask, n95, phillips, honeywell, roche, antigen, reagent, propane, steel, jigsaw, bleach, vaccine, vaccination, astrazeneca, container, gasoline, oxygen, microchip, cleanser, plastic, and dairy*. Those are 37 terms out of 164. Some of the terms are just products or suppliers of them and they could be linked in another context than being a shortage within the Covid-19 pandemic. This can be guaranteed for the 2016 KG because the pandemic did not exist at that time but not for the 2022 one. Moreover, some of the terms like “n95” or “vaccination” are most

likely connected to the Covid-19 pandemic. Thus, the **2022 version should not be used** if the created KG should help to identify shortages without knowing them.

Entity Typing Parameter Tuning

In this study, Entity typing uses the Luke classifier. To avoid labeling entities based on uncertain predictions, a label was only kept if the model attributed a probability over a certain threshold to it. An experiment determined that threshold. A sample of abstracts was classified with different thresholds and a human annotator labeled the result as correct or incorrect. The **threshold with the highest F-score based on the annotation** was kept.

Experimental setup. We took a sample of 10 000 abstracts from the reduced dataset. The Luke classifier predicted a label for all noun phrases per sentence. Subsequently, all the noun phrases, their types, and the prediction probability were put together in one table. Thereafter, a sub-sample of ten abstracts was taken from the bigger sample. The reason for the bigger sample is that the method is based on majority vote classification which works better with a bigger sample because there are more mentions of the term. The reason for the sub-sample is to reduce the workload of the human annotator. For each threshold (ranging from 0-1 in 0.1 steps), the entity typing method was applied as described in the methodology. The human annotator labeled all noun phrases and their predicted types within the sub-sample as correct or incorrect for all thresholds.

Evaluation. From that, the precision, recall, and F-score were calculated. The relevant terms are all the noun phrases in the sub-sample which were classified correctly according to the human annotator. The retrieved terms are the terms with a higher probability than the threshold. The true positives are the number of retrieved terms which were classified correctly. Figure 7.3 displays the result of the threshold tuning. The red line marks the maximum F-score which is reached at a **threshold of 0.3.** In general, it can be seen that the F-score is quite low. This means that the labeling is not very accurate. To avoid adding false labels to the KG, the majority vote method described in section 7.1.2 was used.

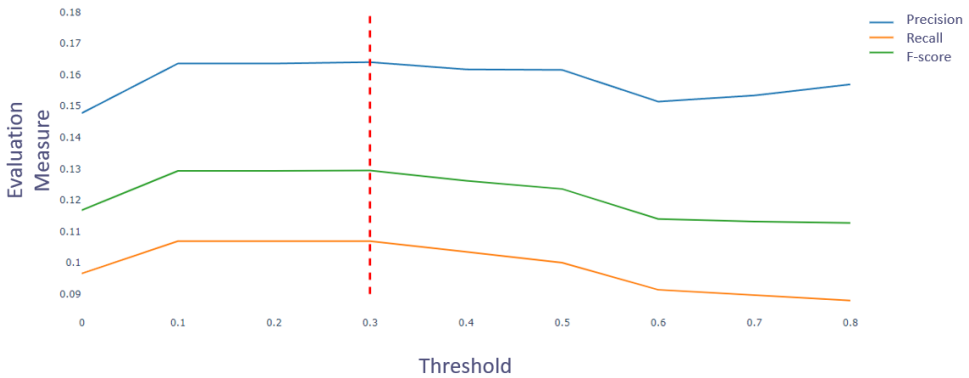


Figure 7.3: The precision, recall, and F-score for different thresholds of correct labels by the entity typing method

Seeding with the Initial KG

In this experiment, we tested different ways of **connecting the initial KG enhanced by the entity types to the triples from RE**. The extracted triples are only added to the KG if they originate from a text fragment that is related to an entity in the initial KG. This way the KG is as

condensed as possible and mostly focused on shortages in SCs. The matching can be done on different text fragments: abstract, sentence, 3-sentence-window, and entity. That means an entity of the initial KG has to be part of that text fragment. Also, partial matches were accepted. On a sample of 10 000 articles from the shortages-related dataset, the **matching to the different sizes of text fragments** was applied. Precision and recall were calculated in the same way as for the KG (see section 7.2.2). Table 7.2 shows the results.

Text Fragment	Precision	Recall	Number of Triples
Abstracts	0.054	0.576	118 265
Sentences	0.054	0.576	118 169
3-sentence-window	0.054	0.576	118 251
Entity	0.055	0.571	108 684

Table 7.2: Performance of the triples resulting from matching the initial KG to different sizes of text fragments

The results show that the **size of the text fragment that contains an entity of the seed KG is not very important**. The scores almost did not change. However, the entity connection had a slightly higher precision and fewer triples than the rest. Therefore, that method was used in this research.

7.2.2 Intrinsic Evaluation of the KG

We evaluated the KG intrinsically. First, the domain affiliation was measured for each step. Second, some general graph statistics were calculated. Finally, 10-fold-cross-validation was performed to measure how self-contained the KG is. This will answer the first part of the last research question of how to construct a domain-specific KG to help identify Covid-19 shortages.

Domain Affiliation

This part evaluates how much the developed domain-specific **KG complies with the domain of Covid-19 shortages in SCs**. It is difficult to evaluate the number of triples because it is hard to say whether two entities being related are relevant or not. Thus, only the unique entities within the KG were evaluated. Since the targeted domain is shortages in SCs, it can be measured how many **shortage-related entities are in the KG**. This includes the shortage indicators because they are part of the domain as well.

Precision and recall. Precision and recall are calculated based on the shortage-related entities. The retrieved terms are the unique entities in the KG. The number of relevant terms is calculated differently for precision and recall. That is because the list of shortages is far from complete. There are many synonyms used in the text, a lot of times a shortage term is part of an entity and also some additional shortages are mentioned. Since precision and recall are calculated with different relevant terms, they should not be combined and the **F-score is not calculated.**
To calculate the **recall**, the shortage terms are the relevant terms. The precision is not very interesting in this case. It will always be low because the list of shortage terms is only 210 terms long and the number of entities is around 100 000. Therefore, the maximum precision would be 210/100 000 $\approx$ 0.0021.
To calculate the **precision**, all entities are considered relevant that contain a shortage-related term. In that case, the recall does not make sense because the number of unique entities that

contain a shortage term is added to the denominator (number of shortage terms). The equation for the recall is $\frac{\text{number of relevant entities}}{\text{number of relevant entities}+210}$. The larger the KG, the closer to 1 this fraction becomes (given that there are some relevant entities). Thus, it does not really measure the quality of the KG.

Method	**Number of Triples**	**Precision**	**Recall**
1. Entity linking	2684	0.274	0.252
2. Adding synonyms (initial KG)	2794	0.269	0.262
3. Adding entity types (& cleaning)	2995	0.269	0.262
4. RE	162 978	0.059	0.429
5. Cleaning	126 162	0.063	0.691
6. Repeating RE (& cleaning)	135 767	0.062	0.695
Repeating entity linking (& cleaning)	1 0425 66	0.032	0.814
7. Adding subclasses (& cleaning)	222 991	0.062	0.695
8. Adding entity types again (& cleaning)	**230 291**	**0.062**	**0.695**
Deduplication AmpliGraph	-	-	-
Deduplication Dedupe (& cleaning)	213 292	0.062	0.690

Table 7.3: Evaluation measures for each step of the KG creation process

Results per method. Each step of the KG creation process was evaluated with the domain affiliation method. Table 7.3 shows the results per step. The enumeration corresponds to the order of how the KG was created. The steps without a number were not included to build the final KG. It can be seen that **each step either increases the recall or does not change it** (excluding the discarded methods).
The **precision is reduced with all steps** besides the cleaning step after RE. It is mentioned separately because it made such a big difference after RE. The biggest drop happens at RE. We expected that because a lot of entities are added and thus also irrelevant information. However, the **precision is underestimated** because more entities could be relevant but they are not part of the ground truth lists. Furthermore, the methods also add information that helps in the shortage identification, e.g. the type “object”, but is not itself relevant and thus not counted towards precision but the information is also not noise.

Removed methods. **Repeating Entity linking performed badly**. While the recall increased and around 25 additional relevant terms are now part of the KG, the precision halved and the KG expanded by factor 8. Moreover, only a few entities were linked (around 13%). That means that most of the added triples are related to only a small fraction of the KG. Most likely the **KG is not domain-specific anymore**. In addition, the largest part is probably not based on the dataset anymore but it is just a subgraph from DBpedia. Especially since most of the entities could not be linked. Since this approach introduces more noise than it enhances the KG, it was discarded.
Deduplication was discarded as well. The deduplication using AmpliGraph only found five sets of duplicate triples when the tolerance was set to 0.5. However, these were no duplicates. A lower tolerance did not return any duplicates. A higher tolerance was not tested because already at 0.5 only false duplicates were returned. For the deduplication using dedupe, the size of the KG was reduced by around 17 000 triples but the precision remained the same and the recall decreased. That means that also relevant entities were removed.

Final results. Out of 210 shortages and shortage indicators, 146 are part of the KG. This means that **around 70% of the searched terms are part of the KG** (including terms that are only part of an entity). **6352 out of 102 467 entities are related to shortages in supply chains** according to the shortage lists. This is a fraction of around 6%. This can mean that the rest of the KG is irrelevant. However, the shortage term lists are not complete and the precision is underestimated. Furthermore, the rest of the entities could be context around these terms. It can thus still be assumed that it is a domain-specific KG. Unfortunately, that is not evaluated at this point.

Analysis of the KG

This section is not an experiment, it is an analysis of the final KG. The entire KG is too large to be plotted. To get an idea of what the graph looks like, the most occurring relations and entities are reported. Furthermore, the sub-community of the term “shortage” is displayed.

Most occurring entities. Figure 7.4 shows the 20 most common entities in the KG. *Object* appears very often because of entity typing. This is good because it helps to find things that can be in shortage. Virus-related terms such as *infection, virus, cell, protein, and pathogen* appear also quite often which makes sense because the dataset contains mostly medical publications. Moreover, pandemic related-terms such as *patient, treatment, drug, world, outbreak, drug, and vaccine* appear frequently. That shows that the KG is actually domain-specific, at least to the domain of Covid-19. In addition, “drug” and “vaccine” are even shortages. Thus, it can be said that **based on the most common entities the KG is relevant for the task of identifying Covid-19 shortages**.

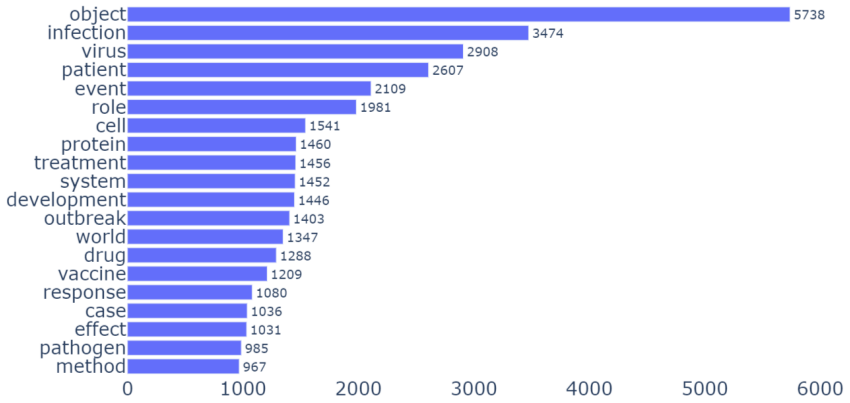


Figure 7.4: 20 most occurring entities in the KG

Most occurring relations. Figure 7.5 shows the 20 most common relations in the KG. It can be seen that these relations have high counts and already make up a large part of the KG. In fact, around 67% of the triples in the KG contain one of these 20 relations. The ***subclass_of* relation is by far the most occurring one**. The total number of super-classes is only 1528. However, each class appears around 60 times on average. There are several very frequent classes, e.g. “infection” and “virus”, but no class which is disproportionally large.

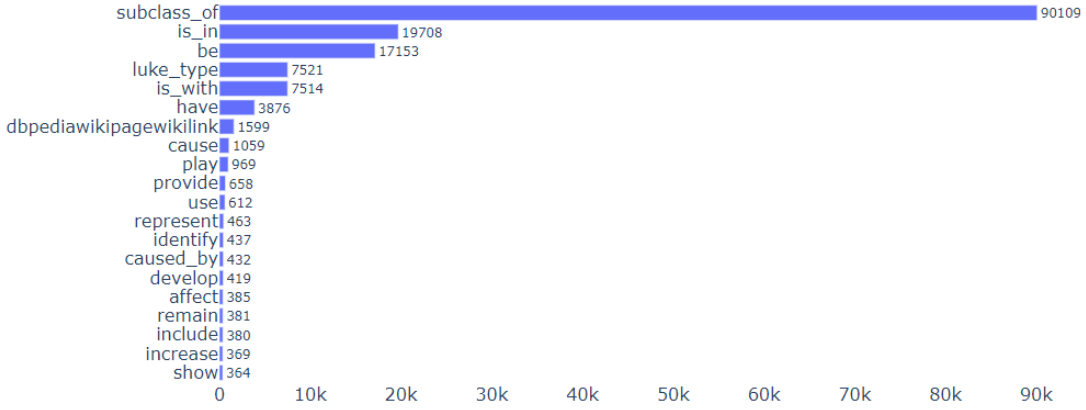


Figure 7.5: 20 most occurring relations in the KG

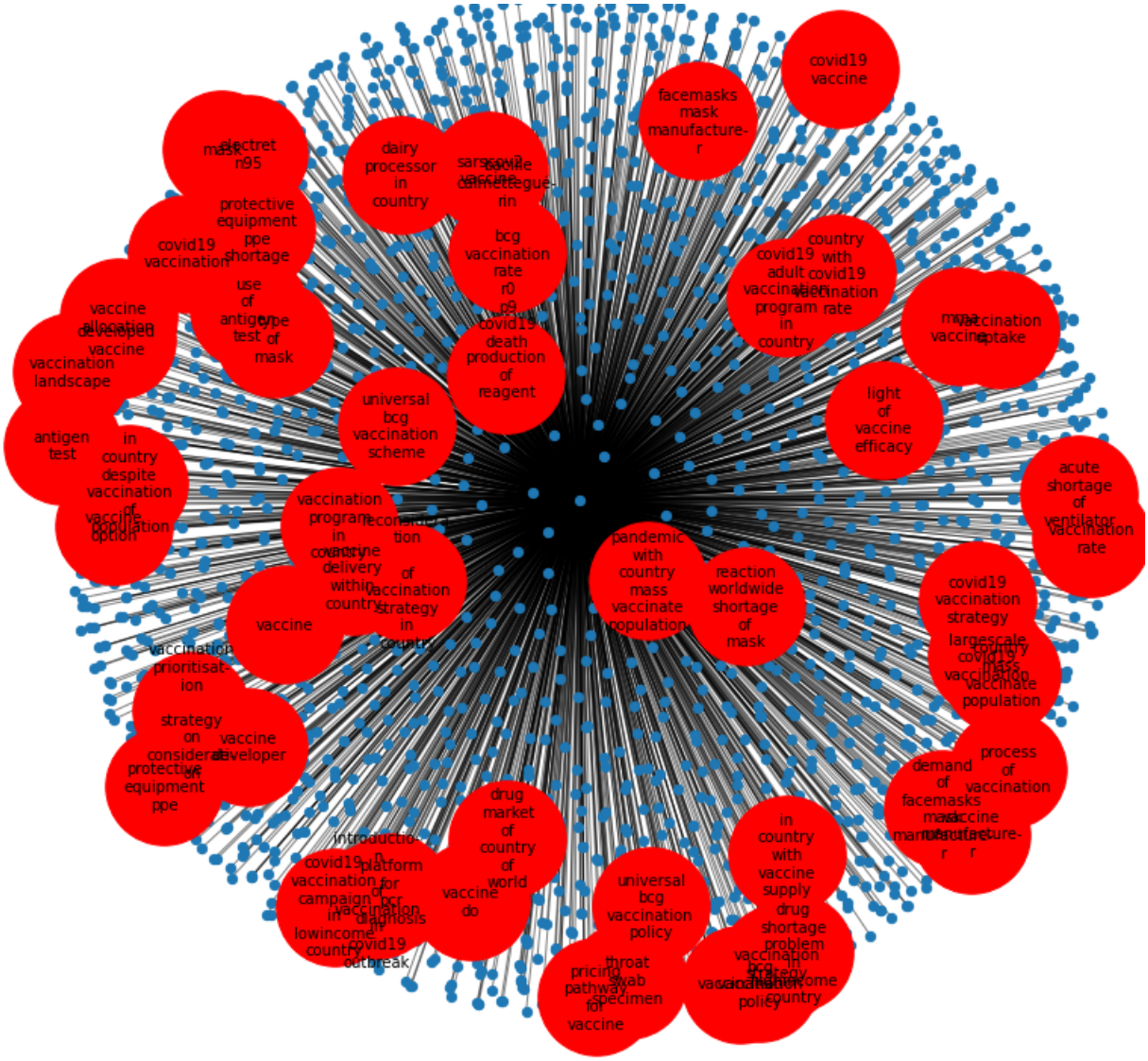


Figure 7.6: Community around “Shortage”

Community around “shortage”. Figure 7.6 shows the community around the term “shortage” to get an idea of how the KG looks. The red nodes are shortages that were found in the community. The smaller nodes are also entities but do not contain a shortage term. The com-

munity was extracted on a reduced version of the KG. Only triples were considered where the subject or object is part of the top 5 top entities measured by centrality. That already results in a KG of 14 360 triples. To extract the community, the libraries Community [108] and NetworkX [109] were used. **Different types of shortages can be seen**, e.g. the categories, *PPE, vaccine, tests, and ventilation* are shown.
When looking at the triples in the displayed community of the KG, it can be seen that the **KG contains a lot of irrelevant details**. Many of the heads and tails are no actual entities. Synonyms contain details that could be left out in order to be resolved to one entity (e.g. “acute shortage of ventilator” and “ventilator shortage”). Some also do not make sense on their own (e.g. “light of vaccine efficacy”) or make no sense whatsoever (e.g. “vaccination do”). Nevertheless, many of the extracted terms are actually entities (e.g. “antigen test”). The same could be observed for the relations. There are many synonyms and meaningless stop words. Heavier cleaning of the entities and relations based on semantics would improve the quality of the KG.

K-fold Cross-Validation

A KG embedding can score triples within the KG with the “predict”-method. The score of a triple can be seen as the probability of this triple belonging to the KG. If the embedding is built on the entire KG, the score is almost always 1 for each triple. Therefore, we used k-fold cross-validation.

Implementation. For that, the KG was split into ten parts. An embedding was trained on nine parts and the tenth one was scored. That was repeated ten times leaving out each part once to score the entire KG. Entities in the part to be predicted that were not part of the other nine parts had to be discarded because the embedding can only score triples of known entities. That is a limitation of the AmpliGraph implementation. Per part, the mean and standard deviation of the scores per triple were calculated. The resulting ten mean scores were averaged again to obtain one validation score for the entire KG. Additionally, the distribution of all probabilities for all triples was plotted.

Results. The 10-fold cross validation lead to a **mean probability of 0.5** for all triples with a **mean standard deviation of 0.3**. Figure 7.7 shows the probability distribution for all triples. It shows that some parts of the KG fit very well together. It also shows that there are many outliers in the KG. Therefore, **no definite statement about the quality of the KG** can be derived from this.

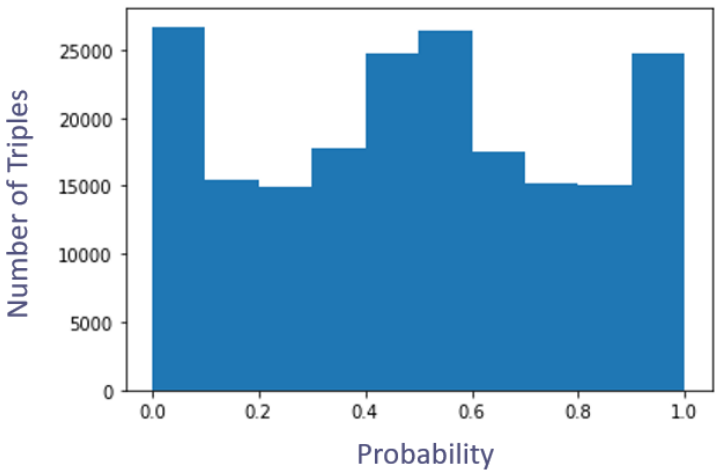


Figure 7.7: Probability distribution for all triples

When looking at the **triples with a low score**, many of them are the sub-classes. It might be a more coherent KG when sorting those out but they contain valuable information for the shortage identification. To see if the low-scoring triples might all be noise, the domain affiliation evaluation was applied on the part above 0.5 and on the part below 0.5. The triples with the lower probability performed better. Therefore, the **probability of a triple fitting well to the KG seems to have no connection with the domain affiliation**.
The results should be considered with caution as this method is **dependent on the quality of the KG embedding**. Furthermore, the KG embedding was not tuned. Thus, the performance of the embedding in predicting the triples might be imprecise.

7.2.3 Identifying Potential Shortages

This part answers the second part of the last research question of *Does such a Knowledge Graph improve the detection of potential shortages?* The same experiment as described in 5.2 was executed a third time with the KG-related adaptions described in the methodology. Thereafter, the results were compared to the previous results. Moreover, we analyzed the methods to identify shortages to see if there might be a better ensemble of them. The methodology of this research is an answer to the overall research question of *How to automatically find unknown potential shortages in supply chains within the Covid-19 pandemic in text?* To evaluate how well the potential shortages were identified, we analyzed the the final list of suggested shortage terms.

Experimental Setup

The shortage-identification methods were executed as described in the methodology with certain settings. There are some hyperparameters that were changed per run and some parameters per method which stayed the same for all runs. We applied all methods to the **reduced dataset** containing only shortage-related articles. The **hyperparameter search space** is:

- **Input entities** (type of terms): entities in the KG, only objects in the KG *
- **Top n terms** (the number of terms to return): 100, 500, 1000
- **Replace sub-classes** (replacing terms by their super-class before applying the methods): True, False

* The input entities are the list of terms per abstract on which the methods are applied. One option is to consider only objects. For that, all entities are extracted from the KG that are related to any of these entities: 'object', 'resource', 'product', 'component', 'commodity'. These entities are referred to as objects.
Table 7.4 shows the values of the parameters for each method. The last column indicates whether the method needs a KG. The parameters are:

- The **Measure** used to weigh the terms
- List of **Seed terms** as an input for the respective method
- **Context size** in words

Method	Measure	Seed terms	Context size	KG required
Term frequency				*No*
TF-IDF				*No*
Context occurrences	Term frequency	shortage indicators	30	*No*
Context difference				*No*
Context occurrences	TF-IDF	shortage indicators	30	*No*
Context difference				*No*
Word embedding		shortage indicators		*No*
Word embedding		“mask”, “shortage”		*No*
KG neighbor occurrences	Term frequency	shortage indicators		*Yes*
KG neighbor occurrences	TF-IDF	shortage indicators		*Yes*
Link prediction		shortage indicators		*Yes*

Table 7.4: Prediction methods and their parameters

Results

Table 7.5 shows the results of the different settings for all methods combined for 500 top terms. We chose 500 terms because that returned the best value for all settings. The results for 100 terms and 1000 terms can be found in appendix C.4. The settings are ordered by F-score.

Input Entities	Precision	Recall	F-Score	Number of Types
Objects	**0.038**	0.461	**0.07**	11
KG entities	0.037	**0.477**	0.069	11
Objects replaced by super-classes	0.037	0.452	0.069	11
KG entities replaced by super-classes	0.036	0.465	0.067	11

Table 7.5: Evaluation measures for all methods together on the reduced dataset for 500 terms with the KG for different input entities

It can be seen, that only considering **objects**, i.e the terms of the abstracts that are related to an object-entity in the KG yield the **best results with an F-score of 0.07** and a precision of 0.038. The reason for that is probably that the possible terms to be returned are not only KG entities but also an object. Thus, irrelevant terms that are not an object re filtered out. This proves the usefulness of the KG. However, the **difference in all performance measures is not significant**. As it seems, it does not matter much rather all entities are considered or only objects, and replacing by sub-classes also did not change much.
The methods where the terms were **replaced by the super-classes perform worse** than without replacing. This is unexpected as the frequency-based methods were supposed to work better when resolving synonyms or specific types of something to one. The reason for that could be that there are too many classes or the classes are too general. Maybe not all shortage-

identification methods should work with the replaced classes but only the frequency-based ones. **11 out of the 18 defined shortage types** (see 4.3) could be found in the list for the reported methods. This means, that the method not only returns shortages of one type but the suggested shortages are quite diverse.
The results of each individual method are in appendix C.3. Looking at the performance of the **individual methods**, the ***word embedding similar to the terms "shortage" and "mask"*** **performed best with an F-score of 0.14**. This is followed by the *KG neighbor occurrences with TF-IDF* with an F-score of 0.12. The context methods and TF-IDF generally scored quite low.

Comparison to previous results

Top Terms	Precision	Recall	F-Score	Number of Types	Origin	Results of
100	0.073	0.217	0.11	6	entire dataset	RQ1, ch. 5
500	0.051	0.5	0.092	12		
1000	0.043	0.632	0.08	13		
100	0.043	0.133	0.065	8	reduced dataset	RQ2, ch. 6
500	0.031	0.36	0.056	11		
1000	0.03	0.528	0.057	13		
100	0.038	0.144	0.06	6	objects, no subclass, reduced dataset and KG	RQ3, ch. 7
500	0.038	0.461	0.07	11		
1000	0.03	0.574	0.058	13		

Table 7.6: Evaluation measures for all methods together on the entire and the reduced dataset and KG

Table 7.6 displays the results of the previous shortage-identification experiments as well as the **best performing setting of this experiment**. Only the best setting was taken from this experiment for easier comparability. Time was not considered in this comparison because it is not really comparable since the shortage identification was adjusted. There are more methods in total and the individual methods were adapted to use the KG.
For the methods without the KG, an anti-proportional relation between the number of terms and precision can be observed while that cannot be seen for the KG method. For all methods using the KG, the F-score increases when increasing the number of top terms from 100 to 500. A large **increase in recall can be observed with the increase in top terms**. Again, the number of different shortage types was similar.

Top Terms	Precision	Recall	F-Score	Origin	Results of
100	-0.0051	0.01	-0.0051	reduced data	RQ2 vs. RQ3
500	0.0075	0.101	0.0139		
1000	0.0003	0.045	0.0008		
mean relative gain	5%	15%	6%		
mean relative loss	-34%	-17%	-32%	entire data	RQ1 vs. RQ3

Table 7.7: Difference of Evaluation measures of with and without the KG

Table 7.7 displays the gain in performance by using the KG on the reduced dataset. It can be seen that the KG method outperforms the method on the reduced dataset for 1000 and 500 terms in F-score, precision, and recall. On average, the F-score is 6% higher and the gain in recall is even 15%. Only for 100 terms the method without the KG has a higher precision and F-score. This shows that the usage of the **KG actually improved the identification of potential shortages**.
However, applying the method on the **reduced dataset and the KG does not perform better than applying the method on the entire dataset without the KG**. A loss in F-score of 32% on average can be observed. This could be because the reduced, shortage-related dataset obtained from TM missed some relevant articles. Another reason could be that some relevant entities are not part of the KG.

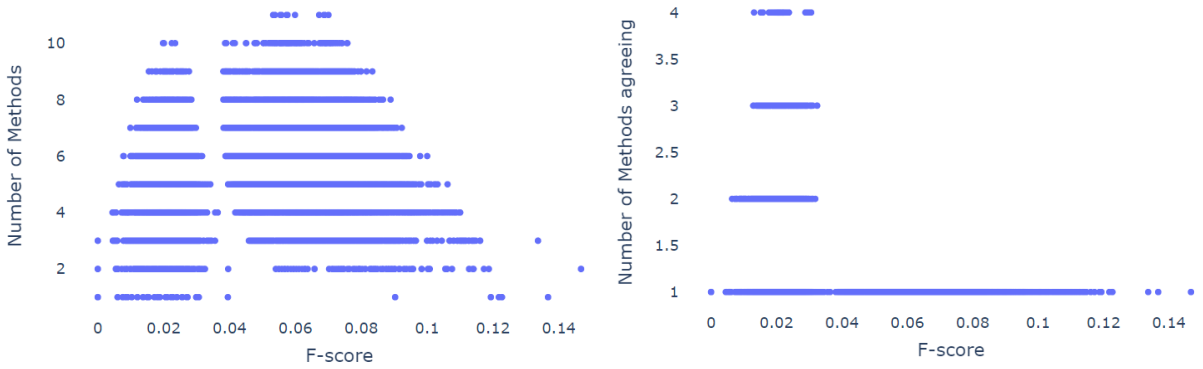
Tuning the Ensemble

The different methods work as an ensemble to find as many potential shortages as possible. So far, all methods were combined. That was also the case here to compare the results to the previous ones and answer the research question. However, this leads to a large list of retrieved terms and thus a high recall but it also leads to many false positives and therefore a low precision. Therefore, we tested a more sophisticated approach to combine the term lists.

Experimental setup. To **strengthen the prediction accuracy**, more than one method should retrieve a term. That means, the **methods need to agree** with each other. A term is only part of retrieved if a number of different methods retrieved the term. This would drastically reduce the list of retrieved terms and it would make the predictions more robust by increasing precision. It is also possible that some methods are not performing very well at the task. Therefore, we test if the performance increases if only the **predictions of a few methods are considered**. All combinations of methods are tried. The following steps are taken:

1. Combine the results of the methods in each possible way from two up until all methods
2. For each combination of retrieved terms, do
    (a) For each i in range 1 to number_of_methods (but max 4)
        i. Calculate the precision, recall, and F-score if i methods agree with the shortage terms
    (b) Keep the maximum F-score of the different i's with the subset of methods and the i
3. Return for each combination of methods the maximum F-score of the different levels of agreement

This tests all combinations of methods. It does not consider single methods because that was already calculated for each method. It also lets up to four methods agree. More than four were left out because the recall was assumed to be too low. However, the result is biased because the combination is found by the highest F-score with the shortage terms. Therefore, this only **evaluates the methods in general and not the prediction performance**.



(a) Number of methods in the ensemble

(b) Number of methods agreeing

Figure 7.8: Plots of ensemble alternatives F-score vs number of methods

Results. Figure 7.8 shows the results of the ensemble. The left plot shows the F-score for different numbers of methods that were combined in the ensemble. It seems **the less methods are combined, the better the performance**. The right part of the figure shows the F-score for different numbers of methods that have to agree. It can be seen that the performance is **best when no methods have to agree** and all terms that are retrieved are combined.
For all settings, **the combination of only two methods performed best with an F-score of 0.15** at 500 terms. These methods were the *word embedding similar to the terms "shortage" and "mask"* and the *KG neighbor occurrences with TF-IDF*. The best recall of 0.58 had all methods together besides *context difference TF-IDF* for 1000 terms and the entities.
For each method, it was checked if the **performance improves or lowers when it is left out**. Only the methods *word embedding similar to the terms "shortage" and "mask"*, *KG neighbor occurrences with TF-IDF* and the *link prediction* improved the performance. For the rest of the methods, the F-score increased when they were left out.
From this, it can be concluded that the ***KG neighbor occurrences with TF-IDF and the word embedding similar to the terms "shortage" and "mask"* are the best ensemble** for the task at hand. They performed best as an ensemble, as individual methods and performance decreased when they were left out. Since the shortages are unknown, only using those two methods for shortage identification would bias the results. In future work, the usage of ensemble methods should be further explored.

Analysis of the suggested Shortages

The ground truth list of shortages is incomplete. Therefore, the performance measures of the methods, in general, are not very high. It is **possible that the rest of the suggested shortages is actually valid but not part of the ground truth**. To find out if this is true, we looked at the list of suggested shortages retrieved by the best method.
The list of retrieved terms is quite long, therefore we only looked at a random sample of the retrieved terms of 100 terms. As it turns out, there are some **additional shortages** that are not part of the list. For example, "meat shortage" or "remdesivir". Moreover, there were more **synonyms to known shortages** which are part of the ground truth list, like "rdt" (rapid diagnostic tests) or "healthcare equipment". Terms that are shortages but were left out of the ground truth because they are **not relevant for the use case** as they are no products were also found. For example, "shortage of expert neurologist" and "shortage of bed" were returned. Another example is "peptide" deficiency which is a **health condition**. That is due to the dataset being about medical publications. Another group of **terms is related to SCs**, e.g. "supply chain network". This shows a limitation of the algorithm as it does not distinguish between the types of shortages.

Even though some new shortages and synonyms were found, a big part of the retrieved terms is still irrelevant. Nonetheless, the list is only 3112 terms long. If looking at only a small sample already uncovered some new shortages, then looking at the entire list is definitely worthwhile. Further, for a **procurement expert, this list of suggested shortages is a support** since it takes less time to go through 3112 terms than having to read numerous articles to find shortages. It might even reveal shortages that would not have been found otherwise.

7.3 Summary of the Chapter

This chapter described how the **KG was built**. For that, an initial KG was extracted from DBpedia based on the shortage indicators. It was enhanced by entity types, such as "object". Next, open RE was applied on the CORD-19 data using the initial KG as a seed. The refinement methods further enhanced the KG: adding super-classes from the entities, repeating entity typing, and repeating RE. The KG was cleaned using some NLP processing methods. Automatic deduplication methods and repeating entity linking to DBpedia proved to be unsuccessful. There were some experiments done on the steps of the KG creation.
The KG was **evaluated intrinsically**. That included the domain affiliation, which evaluates how well the KG fits the domain of shortages in SCs by measuring the number of domain-related entities. Moreover, the KG's most common entities and relations were discussed and a subgraph was presented. Finally, cross-validation was done with the help of a KG embedding. The intrinsic evaluation gave mixed results. While more than half of the searched terms are part of the KG, only a small part is related to the domain. However, the reason for that could be that the ground truth used for evaluation is incomplete. The cross-validation gave an average probability of 0.5 of a triple belonging to the KG which does not say anything.
The **shortage-identification experiment was repeated** with the KG and some additional methods. The KG improved performance on the shortage-related dataset with the highest F-score of 0.07. The best methods are *KG neighbor occurrences with TF-IDF* and the *word embedding similar to the terms "shortage" and "mask"*. They performed best as an ensemble, as individual methods and the performance decreased when they were left out. Looking at the final list of suggested shortages, it can be concluded that there are many more relevant terms suggested that were not part of the ground truth list but also a lot of noise.

8 DISCUSSION

This research developed a method to automatically identify Covid-19 shortages in SCs using statistical NLP methods and a dedicated KG. The main source of data was the CORD-19 dataset of research publications about Covid-19. An ensemble of TWS based on term frequencies and co-occurrences was applied to the dataset. The method suggested many terms as potential shortages. Some of them were part of the ground truth list and some of the remaining terms might be a shortage but are not part of the predefined list. However, many of the suggested terms cannot be in shortage because they do not refer to an object or even an entity. To mitigate this problem, a KG was developed based on a SC-related sub-graph from DBpedia, which was enhanced with shortage-related articles of the CORD-19 dataset and refined. The articles were selected using TM, which showed a significant improvement over the standard keyword-based method. The KG contains a lot of noise but it improved the identification of shortages using TWS.

8.1 Interpretations

In the following, the research questions will be answered. For each question, the most important results are reiterated, discussed, and interpreted. Finally, the respective question will be answered.

8.1.1 Statistical NLP Approaches to Identify Potential Shortages

In the first part of this research, we applied statistical NLP methods to identify Covid-19 shortages in supply chains. An ensemble of different TWS was used. The combination of context knowledge and frequency proved to be useful. The F-score, precision, and recall were calculated based on different numbers of top terms retrieved by the TWS to answer the first research question: *How well can statistical NLP methods alone identify shortages in text?*
The **best overall F-score is 0.11** with a precision of approximately 0.07 was reached at 100 top terms. The best recall of around 0.63 was reached at 1000 terms. That means that some of the shortages were covered. The precision decreases relatively quicker than the recall, which means that **retrieving more terms relatively leads to more noise than additional shortages**. There can be several **possible reasons for the low precision** and thus the low F-score:

- **Semantics not included**: Terms that cannot be in shortage, like verbs or abstract concepts were predicted as well. To avoid that, the KG was built
- **Incomplete ground truth list**: The predefined list of shortages is incomplete. This means that some of the predicted terms might be a shortage but they are not on the list.
- **Methods not well tuned**: The methods could perform better if they were tuned better. That includes the individual methods as well as the way the methods were combined. Some methods might perform poorly and can be left out. To increase the precision, the ensemble can be tuned.

- **Medical dataset**: The dataset is about medical publications. Possibly, the shortages which are looked for are not mentioned a lot. That makes it difficult for the methods to detect them.
- **False assumptions**: the hypothesis of shortages being mentioned more frequently in certain time frames and in the context of certain keywords could just be wrong. Other terms which are not a shortage could also be especially interesting in a certain time frame. For example, if a product enters the market the first time, it would probably score high on TF-IDF.

Despite the many other reasons why a term could be selected by the proposed methods, many of the terms which are selected are actually shortages. The low precision was expected. Therefore, the KG was built.
Considering that a SC expert would like to know the potential shortages, the **recall should be rated higher than the precision**. It is more valuable to return a larger list of potential shortages that covers more terms than a small list. This has two reasons. The first one is that the list of shortage terms is incomplete. The additional terms which are now seen as irrelevant might be relevant but not part of the ground truth shortage list. The second one is that a human can easily discard irrelevant terms. It is more difficult to find additional terms which are not retrieved by the system.
This shows that the ensemble of TWS over time is a way to identify shortages but not very precisely. This answers the first research question of how well statistical NLP methods alone can identify shortages. This is one solution, there can be many others that might perform better at the task.

8.1.2 Topic Modeling

To improve the performance of the shortage identification, a KG was built. To help the KG be focused on the domain of shortages in SCs, a reduced dataset of only relevant articles was created. In the SC-domain, usually keywords are used to find relevant articles. That is dependent on direct matches of the keywords in text. Furthermore, that can select some articles that contain a keyword but are not about shortages. Therefore, we hypothesized that relating texts via TM will outperform the keyword search. TM selects documents based on their entirety instead of just one term. A term can be used in many different contexts, an entire text being related to other texts makes the relation less coincidental. The research question derived from that is: *Does Topic Modeling improve the selection of relevant articles in comparison to keyword-based search?* The second part of the research compares the results of the shortage-identification experiment on all data to the results on the reduced data: *What is the difference in performance when applying the shortage-identification method to the reduced dataset?*

Topic Modeling vs. Keyword-Based Selection

The best parameter setting had an **F-score of 0.44** on average measured with all articles which contain a shortage term as ground truth. That **outperformed the keyword-based search by 8 percentage points**. This means that the TM selection is better. That is mostly due to the selected dataset of around **88 000 articles** being a lot smaller (around 46 000 articles less). Thus, fewer irrelevant articles were selected.
Nonetheless, the F-score is still not very high. Better tuning of the hyperparameters or another method for TM than LDA could improve the selection. Furthermore, the evaluation method might not be perfect. The creation of the ground truth is also keyword-based. An article mentioning a shortage term might not be about shortages. However, those articles create the context around the relevant keywords which can also be valuable.

Model parameters. A big drawback of TM is the instability of the algorithm. Research suggests doing hyperparameter tuning to overcome that problem [25]. For the task at hand, Guided TM with **three topics, $\alpha = 0.03$, $\beta = 0.03$ a seed_confidence of 0.98 and 20 seed terms** led to the highest performance. The analysis of the correlation between the hyperparameters and the performance measures (coherence and F-score) confirmed the results. It found that k, α and β should be low and the number of seed terms should be relatively high. Different than what research suggested, the **TM algorithm seemed to be quite stable**. This might be due to the number of seed terms being quite high, which might limit the variability of the algorithm.
A surprising result was that within the top models, either seeding was given a high value and α and β a low one or seeding was given a very low value. This suggests that the **LDA algorithm might not combine well with seeding**. It seems that the model's decisions were either controlled by the seed or by α and β. In the second case, the models are just based on LDA and are not guided.
Another unexpected finding was that there was a negative correlation between the F-score based on the articles and the F-score based on the shortage indicators. From that, it can be concluded that **the F-score based on shortage indicators is not a good estimation of the model performance**.

TM success. The success of the TM supports the findings of other research, for example Bansal et al. [10]. SC management is just one example. However, in all domains where relevant articles are selected by keywords, TM can be considered as an alternative. Especially, if the articles have to be read by a human, like in the field of SC management. This is due to the low precision of the keyword-based search. That means that many irrelevant articles are selected which would then manually have to be sorted out.

Difference in shortage-Identification Performance

The second part of this research question asks what the difference in performance of the shortage identification on the reduced dataset is. For that, the shortage-identification experiment was repeated and the loss in performance was calculated.
The **process time** was the only measure that **improved** with a mean relative loss of around 77%. The best **F-score was 0.065**. The three **evaluation measures all decreased** with a mean relative loss in F-score of around 36%. However, the loss in precision was higher than the loss in recall. The lower change in recall is good because that means only little relevant data was lost.
The **loss in precision was unexpected** because focusing the dataset on shortages in SCs should remove noise and lead to more relevant terms in general. This could be explained by the nature of the methods. For example, TF-IDF uses the information of a term being frequent in some articles and not in others. If the others are removed, the term becomes less significant. Another example is the context method. It is only considering articles that mention the term. Thus the performance is independent of how many irrelevant articles are in the dataset. Term frequency, on the other hand, should profit from a domain-specific dataset. Evidence supports that claim, as **some of the frequency-based methods improved** using the reduced dataset.
Another explanation for the loss in precision is that the **TM removed relevant articles as well**. That can be further supported by the fact that only around 38% of the articles mentioning a shortage term were selected. Nevertheless, without knowing the shortages, TM is still a better approach than keyword search based on shortage indicators.

Even though the performance decreased, **the reduced dataset was still kept because**:

1. **Processing time decreased** for shortage identification and for the KG creation.
2. A domain-specific dataset is necessary to **create a domain-specific KG**.
3. The **size of the KG** stays manageable which saves memory.
4. As a **proof of concept methodology for future work**: a better topic model would lead to a dataset that fits the domain better and the prediction performance might not decrease.
5. In general, it **saves a lot of human effort**.

8.1.3 Knowledge Graph

In the last part of this research, a domain-specific KG was created to improve the shortage-identification method. The first part of the last research question was *How well can a domain-specific Knowledge Graph be constructed automatically without knowing the shortages?* The methodology of the corresponding chapter describes as a possible solution for that. To get an answer to the question, the developed KG was evaluated intrinsically. The second part of the question is *Does such a Knowledge Graph improve the detection of potential shortages?* For that, the shortage-identification experiment was repeated including the KG, and compared to the previous results.

Intrinsic Evaluation

Domain Affiliation. While around **70% of the ground truth shortage lists** are part of the KG, only about **6% of the entities in the KG are about shortages in SCs**. This suggests, that a big part of the relevant information is included in the KG but there is also a lot of irrelevant information. To improve the domain affiliation of the KG, more KG refinement methods should be applied. The methods so far **did not consider the refinement towards the domain**. Another explanation for the low precision could be that it is underestimated. Some of the other entities in the KG are **relevant but not part of the ground truth lists**. Furthermore, it could be that they are not relevant on their own but are related to relevant entities and give context knowledge around those.

Relations and entities. The final KG has 27 002 different relations but **67% of the KG is made up of the top 20 relations**. It might be possible to group the rest of the relations to one of these relations to create more general relation types as customary in KGs. An example advantage is that queries can be done specifically on the relation. The **most connected entities**, e.g. “object” and “vaccine”, seem **relevant to the domain of Covid-19 shortages**. The KG might not contain the information that “vaccine” is in shortage but the fact that it is part of the KG is already helping the shortage-identification method.

Cleaning. When looking inside the KG, it became evident that more cleaning needs to be done. The values of the entities and relations contain irrelevant details and some do not make sense. This is probably due to the triples being extracted from raw text. Some cleaning was done on the extracted text fragments but it **did not include the semantic meaning of the terms**. That could remove irrelevant information without changing the meaning of an entity. Nevertheless, many of the entities are already quite clean.
This is further visible in the results of the cross-validation with the KG embedding. The mean probability of a triple belonging to the KG is 0.5. This suggests that **although a large part of the KG is consistent in itself, there is still a lot of noise**. The comparison of the lower scoring

part of the KG to the higher scoring part showed that there seems to be no connection between this probability and the domain affiliation.

Shortage Identification

The results of the shortage-identification experiment on the reduced dataset using the KG show that the **KG actually raised the performance**. The best setting was considering only objects in the KG with an **F-score of 0.07**. However, the different settings with the KG, e.g. considering only entities or objects, all performed quite similarly and the difference was not significant.

Recall. The **relative gain in F-score is 6% on average** for the best setting in comparison to the experiment on the reduced dataset. When looking at the absolute difference in F-score, it seems quite small. This is due to the precision being so low in general. However, the **absolute change in recall is quite large**. As already discussed, the recall is more important than the precision because it is easier to discard irrelevant terms from a list than to add unknown terms. Therefore, the recall might have to be weighted higher in the F-score.
The performance of the KG on the reduced dataset was worse than the one without the KG on the entire data. This is probably still because the TM did not do such a good job at selecting the relevant articles. The low precision, in general, can be explained with the same reasons mentioned in 8.1.1.

Individual methods. An analysis of the individual methods and how to best combine them showed that the **word embedding similar to the terms ”shortage” and ”mask” and the KG neighbor occurrences with TF-IDF produced the best results** as an ensemble and individually. For the word embedding, this might be due to the fact that it was seeded with one known shortage. Moreover, the term “shortage” is very often used in combination with a shortage, e.g. “ppe shortage”. The idea of the context method was to capture those co-occurrences. As it seems, the embedding-based method is better at that task. The fact that the neighbors of shortage indicators in the KG are relevant, shows that the KG actually contains relevant terms.

Analysis of the suggested shortages. The low precision, in general, suggests that most of the retrieved terms are irrelevant. However, when looking at a sample of the retrieved term, it turned out that this was not the case. While there are still many irrelevant terms on the list, there are also relevant terms retrieved which are not part of the ground truth. Therefore, the **actual precision can be assumed to be higher**. This could be evaluated by a human expert in future work.
With this, also the overall research question of **How to automatically find unknown potential Covid-19 related shortages in supply chains in text?** is answered. This thesis showed a method for Covid-19 shortage identification with an automatically constructed KG which has shown limited success.

8.2 Contributions and Implications

This section lists the contributions of this work and relates them to existing research. It also gives some practical implications if someone would like to use some parts of the method again. Table 8.1 gives an overview of the contributions, what challenges were faced and what was achieved.

	Contribution	What does it do?	Challenge	Achievement
TWS (RQ1), Ch. 5	**1. Shortage-identification method**	It identifies shortages in SCs within the Covid-19 pandemic using an ensemble of TWS over time.	Let a system identify shortages without knowing them. Which TWS should be used? How to implement them? What is the best way to combine them?	Seven TWS over time were implemented. The method identified some of the expected shortages but it also retrieved a lot of noise. The best ensemble was only combining two of the methods. The method can be used on any other dataset given some keywords.
TM (RQ2), Ch. 6	**2a) TM to reduce the dataset**	It selects relevant articles in a dataset based on keywords.	Automatically finding articles about shortages without knowing the shortages. How to tune TM properly?	The developed TM algorithm reduced the dataset by 76%. More than a third of the relevant articles were selected and more than half of the reduced dataset is domain-related. The algorithm can be used for other data and domains.
	2b) TM evaluation method	It compares the TM selection against keyword selection.	How to evaluate if the selection of articles is relevant? What makes an article domain-related?	An evaluation method was developed based on the number of articles containing a domain keyword. TM outperformed keyword search.
KG (RQ3), Ch. 7	**3a) Method to create a domain-specific KG automatically**	It creates a domain-specific KG automatically from text without bias towards the dedicated usage. (not including shortages)	How to avoid including bias in the creation process? How to type entities as objects? How to extract triples from raw text without including too much noise but including everything relevant?	A KG about Covid-19 shortages in SCs was created. It contains 70% of the expected domain keywords. At least 6% of the entities in KG are domain-related. The method can be used in other use cases.
	3b) Initial KG creation method	It extracts a sub-graph from DBpedia based on keywords.	There was no initial KG given. How to extract a domain-specific sub-graph from DBpedia without bias?	A method was created, which can be used to quickly obtain a domain-specific KG based on a set of keywords.
	3c) method for KG refinement	It removes noise from the KG and enriches it.	What methods are there? How to ensure no relevant information is removed? What is relevant to add?	Two methods were developed and combined with existing ones. 1. Retrieving super-classes by extracting the roots of terms. 2. A cleaning method consisting of some text processing methods.
	3d) Automatic evaluation of a KG	It automatically evaluates how well the KG fits the domain.	How to measure the domain affiliation of a KG without a human expert? When is an entity/link relevant? How to evaluate the refinement steps?	An evaluation method was developed to measure the domain affiliation of a KG based on a set of domain keywords.
	3e) Use case of the KG (shortage identification)	It uses the KG and TWS over time to identify shortages.	How does the shortage-identification method need to be adjusted? How to enrich the method with the knowledge in the graph? What additional TWS can be included?	The KG slightly improved the performance by considering only entities/objects as potential shortages. Two new TWS were added. The KG neighbor occurrences method is novel and performed well. This KG creation method can be used in other scenarios where relevant entities should be identified from text.

Table 8.1: Main contributions including their challenges and achievements

8.2.1 Supply Chain Analysis

In the related work section, the **application of advanced NLP methods for SC analysis** was identified as a research gap, see for example [55, 56]. Comparable approaches, like demand forecasting models or simulations, are usually based on numerical data, especially before the pandemic. This thesis contributes to that research gap with an example of how to apply NLP methods to business studies, like SC management. This is in line with Chowdhury et al. [8] and Queiroz et al. [9] that call for technology to support responsive sc for pandemic situations. A responsive SC must react quickly to disruptions, the developed method can help with that by suggesting potential shortages. As a consequence, those products can be monitored closely, which makes a quick reaction easier. Another contribution is the list of shortages retrieved by the method of which some might be unknown to procurement experts.

8.2.2 Shortage identification

Shortage warning system. The problem to be solved was to automatically identify Covid-19 shortages. The motivation behind that is that shortages can lead to serious consequences. For instance, the PPE shortage worsened the pandemic [3]. As Iyengar et al. [7] write, being informed about upcoming shortages can help to reduce these consequences. The contributions of this thesis can be placed in "early detection" of the SC disruption stages of Ivanov [52]. Furthermore, the contributions directly build on Iyengar et al. [7], since they utter the need for a global early warning system for shortages. The developed shortage prediction **method is a proposal for a warning system of shortages**. However, the method is no actual *early* warning system yet, as it identifies the shortages based on the entire dataset, so in retrospective. Additionally, it is not a *global* system because the location of a shortage was not considered.

Ethics. On the one hand, a shortage warning system might help to make the consequences less severe. On the other hand, if not everyone has access to that system, it **can give certain parties a competitive advantage**. For example, if one company had access to such a system and its competitor would not, they would react earlier, for example by stockpiling. That could make the shortage even more severe. The idea of this shortage warning system is thus to make it freely accessible to everyone. Unfortunately, that will be nearly impossible because not all actors can know about something at the same time and unbalanced information access can still cause problems.

shortage-identification method. The method to identify shortages is also a contribution (see table 8.1 1. and 3e)). There are other studies that use TWS to predict shortages, e.g. [58, 13]. However, using an ensemble of schemes to predict shortages is new and some of the developed TWS over time might be a novelty. For example, the KG neighbor occurrence method performed quite well and does not exist yet to the best of my knowledge. As for the other schemes, they are mostly self-implemented but similar ones might already exist. Moreover, it was found that the fewer schemes are combined, the better the prediction. The proposed method can also be used in other scenarios to find relevant terms at certain points in time in a dataset given a set of keywords. For example, if someone wants to know more keywords around a certain term, the context occurrence method could be used.
An implication arising from this approach is that the **list of terms to predict, in this case, products in shortage, is never complete**. There will always be more synonyms used in text. Therefore, also partial matches should be considered. Nonetheless, the precision will always be underestimated and the recall should be weighted higher. Another reason for weighting the recall higher is that filtering a term list is easier than finding terms that are missing in a list.

8.2.3 TM

TM to select relevant articles performed better than keyword-based selection which aligns with Bansal et al. [10] who stress the use of textual data and TM to study SCs. Other studies used (seeded) TM to study disease outbreaks [61, 62], such as Covid-19 [63]. They used it to analyze topic trends and to identify keywords. They did not use it to reduce the data. Therefore, the developed **approach to creating a domain-related dataset using guided LDA and its evaluation method are a contribution** to this field (see table 8.1 2a) and b)). They can be used in any other scenario where relevant articles need to be selected based on a set of keywords. However, for the evaluation method, it is good to have a ground truth list of terms that should appear in the dataset.
Some practical implications can be drawn from the results. TM on the entire articles performed better than on the abstracts, thus, **the more text is given, the better the model**. Furthermore, looking at the correlation between the evaluation measures and hyperparameters can be useful to **derive some rules** for how the parameters should be set. This can be helpful, as the parameters need to be tuned every time since the algorithm is unstable [25]. Moreover, the **number of seed terms is an interesting parameter to tune**, since it is usually not considered as a parameter and it made the algorithm more stable. The evaluation based on the seed terms was not a good approach.

8.2.4 KG

KG creation method. Chen et al. [64] call for methods to **automatically built KGs without domain-specific labeling**. This study is a contribution to that research gap (see table 8.1 3a)). Reese et al. [74] also created a framework to automatically construct a Covid-19 KG. However, the input data needs to be in a structured format. This thesis provides a method starting from a set of keywords and raw text.
Some new methods were developed to construct the KG. One is the cleaning process based on NLP processing methods. The other one is the extraction of roots from noun phrases to create super-classes to add to the graph (see table 8.1 3c)). Moreover, the creation of the initial KG from keywords can also be seen as a contribution (see table 8.1 3b)). In any scenario where a domain-specific KG needs to be created when resources are scarce, this provides an easy way to do that. There are some lessons learned from the construction of the KG. For example, open RE is the preferred method over relation prediction when the relations are unknown which is often the case when a KG is built from raw text. Another one is that DBpedia contains a lot of noise, thus the relations to extract should be chosen carefully. Measuring the domain affiliation of the KG based on domain terms can be seen as another contribution (see table 8.1 3d)). Normally, KGs are evaluated using human evaluation or a hold-out prediction.
The methods to automatically construct and evaluate a KG are all unsupervised. They would not have to be adjusted much to be applied to **another domain with a new list of keywords and another dataset**. In addition, this is an example of how a KG can be combined with other methods. The improvement in prediction performance confirms the usefulness of KGs. It shows that the semantic knowledge within KGs can actually improve NLP methods.

KG to summarize multi-disciplinary knowledge. The KG is an example of connecting knowledge from different scientific disciplines. Many problems are multi-disciplinary. Therefore, it is helpful to combine knowledge of different areas and derive new conclusions from them. There are more and more publications about specific types of problems. For a human, it is almost impossible to get an overview of all research made within a domain. Let alone, make connections between interdisciplinary scientific findings. This KG is an example of how scientific knowledge can be combined, summarized to its core, and connected. Furthermore, new conclusions can be derived from it by humans but also by automatic reasoning. Since new conclusions made

by reasoning techniques are traceable on the KG and thus human-comprehensible, a KG can be seen as a **contribution to explainable artificial intelligence** [110].

New Covid-19 KG. There are several KGs around Covid-19. Many of them are biomedical, e.g. [75, 76]. There is also another KG based on the CORD-19 data. Nonetheless, this graph only considers the meta-data of the publications, such as authors. To the best of my knowledge, there is no Covid-19 KG revolving around the shortages in SC during the pandemic. The developed KG is thus a **contribution to the field of Covid-19 KGs**.
Identifying shortages with the help of the developed KG is just an example use case. There can be **several other applications**. For instance, trying to understand how a shortage occurred. The KG models how certain products are connected. This can be useful to find out where a shortage came from. For example, the shortage of video game consoles was due to the shortage of computer chips. Another idea would be to use it as a database and query it for certain knowledge. For example, if someone would like to know the synonyms to "face masks" or specific types of a certain product.

8.3 Limitations

Dataset. There are some limitations to this study. The method for predicting shortages and the KG construction is only based on CORD-19 which is mostly about medical data. This limits the findings within the SC domain because the articles are not about that domain. This also makes a large part of the dataset irrelevant to SCs.

Evaluation method. The ground truth shortage list is not complete. Moreover, not everything on the list is a shortage. This distorts precision and recall. The method does not consider the time and place of a shortage which are some important aspects. When the method suggests something as a shortage, that might only be valid for a certain country or a certain timeframe. Moreover, the current method does not distinguish between different types of shortages, such as "PPE" or "vaccine".

TM. We only applied the guided LDA algorithm. There are more TM algorithms that might yield better results. Moreover, the final parameter set might not be the optimal one. The F-score based on the seed terms used to optimize the algorithm was not performing well. Another limitation is that TM algorithms always need to be tuned. Thus, the developed method is no off-the-shelf algorithm to simply apply to another dataset. Furthermore, the fact that a shortage topic was identified with this dataset might be by chance. In another dataset that also talks about shortages, there might not be a shortage topic. However, it is also imaginable that the shortage topic will be found in another dataset if a sufficient amount of articles talks about it and the right keywords are given.

Shortage-identification method. The shortage-identification method is limited in some aspects. Other TWS might be a better fit. Also, the parameters of the current methods were not tuned, which could improve the performance. Furthermore, using TWS is one way of solving the problem, there are other ways to do this. The TWS were built on the assumption that frequency and co-occurrence of keywords are an indicator for shortages. That could be wrong or only partially true. In this study, that hypothesis was not checked. There could be other reasons why the shortage terms are weighted high by the algorithm. In addition, the identified shortages were not verified to be true at the time that they were identified. And again, the time and place of a shortage were not considered. In this research, only product shortages were considered. However, there are other kinds of Covid-19 shortages like health care workers or

hospital beds. Another restriction of the results is that the method identified the shortages retrospectively. Thus, the products in shortage were identified after they were already a known shortage. The frequency of a term and the context might only be an indicator of a shortage after it occurred.

Initial KG. DBpedia was used as an initial KG. It does not have a unified structure. That means that the labels and relations are not very consistent for different entities. Furthermore, in this research all triples to an entity were extracted, only a few relations were discarded. So already the initial KG contained irrelevant information. The choice of a large KG should be seen with caution. A domain-specific, curated KG would have been the better choice, only that it did not exist.

Entity typing and RE. The models for entity typing and RE were not explored thoroughly in this study. There might be a better parameter set for them. Additionally, there are other models for the tasks which might be more accurate than the current ones. The open RE method is limited as it extracts very detailed entities and relations. Not all of them make sense and synonymous entities and relations are not resolved to one. This expands the KG unnecessarily. Especially the number of relations could be reduced greatly by generalizing them to create relation types. Adding on to that, the method of how the triple variations are reduced is simply by the longest or shortest entity or relation. This will not choose the best triple out of the possible ones every time. Moreover, the seeding of the initial KG via string matching might not be the best fit, relevant triples could be missed.

KG refinement. Regarding the refinement of the KG, only a few methods were applied. There are many more which could result in a more complete and more clean KG. The current method only looks for duplicate triples. It does not look for duplicate relations or entities. A limitation of the deduplication method is that it did not find any duplicate triples. The cleaning method only filters out noise from the entities and relations by preprocessing methods that do not consider semantics. Furthermore, no fact-checking was done which can mean that there is false information in the KG. Another restriction from the refinement is that no error detection methods were applied. Moreover, no refinement towards the domain was done.

KG evaluation. The intrinsic evaluation of the KG also has some limitations. The KG is not evaluated as a whole. The relations were not evaluated at all and also not how much it adds that two entities are related. Neither the semantic nor the factual correctness within the SC domain was evaluated. It was only evaluated how much the unique entities fit the domain. Nonetheless, also that result has to be considered with caution because the precision and recall cannot be combined reasonably to one F-score. This is due to the noise in the KG, the incompleteness of the shortage lists, and the underestimated precision.

Defense. Despite all the limitations of the findings, the study still fulfilled its purpose. The aim of this thesis was not to make an accurate prediction of shortages but rather to suggest potential shortages. Furthermore, this study only suggests one way to create the KG. The goal of this study was to show that a KG can improve the identification of shortages and not the creation of a complete and correct KG.

8.4 Future Work

Dataset. In future work, more data sources could be used in addition to CORD-19. For example, a news dataset. The method was tested on the news and it showed quite promising results. The news are focusing on different aspects than the CORD-19 data. Therefore, other shortages could be found which were not found in the CORD-19 data. Nevertheless, the style of writing and the language used within the two datasets is different. A technical term might have an easier synonym in the news. This should be considered when creating a KG on several datasets.

Evaluation. The evaluation methods can be improved. First of all, the list of shortages should be enhanced and verified by a SC expert. Second, the TWS are time-dependent and that should be evaluated as well. For that, the TWS also need to return the month when the time series of the predicted product peaked. For evaluation, the list of shortages should be enhanced by the time frame when it occurred. In future work, the KG should be evaluated completely. That means, also the relations, the semantics, and the facts should be evaluated.

Shortage identification. In the future, the shortage-identification method can be improved. The parameters of the single method and of the ensemble should be tuned. Furthermore, other methods for shortage identification should be considered. They can be other TWS or of a completely different nature. For example, a dataset can be annotated with the known shortages at a certain time. That annotation should also contain the context of the same products when they were not in shortage. It can also contain annotations of other products which were never in shortage. Subsequently, a classifier can be trained on the time series of the context around those products. It could learn how the context of a product changes before it becomes a shortage and with that information actually predict a shortage before it occurs. Furthermore, the location should be included in the prediction which is a difficult task. It could be extracted from text or derived from where the article was published. Finally, the system can be extended to other kinds of shortages than products.

TM selection. To improve the TM, other algorithms can be explored and the hyperparameters should be tuned more extensively. For that, an unbiased evaluation method to tune the model needs to be created as the current one did not perform well. It could also be tried to seed the model with some known shortages, instead of only shortage indicators. Moreover, a human expert should validate if the articles selected by the model are actually relevant to the domain. To validate the approach, it should be applied on another dataset.

Cleaning of the KG. In the future, a less noisy initial KG should be used. That can be an existing one or a self-made one. If the source stays DBpedia, the relations to be extracted should be selected more carefully. Furthermore, missing relations for certain entities should be dealt with. This way a more structured, consistent, and domain-focused initial KG can be built. Entity typing and RE should be researched more to contribute to a cleaner KG. The selection of the triple variations should be based on a semantic approach that makes case-by-case decisions instead of one approach for all. The relation types should be generalized and only relevant ones could be selected instead of including all. Furthermore, the RE should be targeted more towards the domain by improving the seeding with the initial KG. More cleaning should be applied in the refinement phase to increase precision and make the KG more concise. That means, entity and relation resolution should be performed. That can be done with a KG embedding which should also be tuned. Moreover, meaningless triples and irrelevant details

should be discarded. In addition, fact-checking and error detection should be applied to make sure that there are no contradictions.

Alterations of the KG To create a better KG as a source for Covid-19 shortages, the KG can also be constructed by **including the known shortages in the construction**. This would change the objective to mapping the knowledge about Covid-19 shortages as detailed and accurate as possible. For shortage prediction, this KG is not a good choice because it contains a lot of bias. It would be a good fit for other applications which are more focused on understanding the shortages and finding additional ones. As already stated, time is an important aspect of shortages. Therefore, it would be interesting to create a **temporal KG**. This can model that a fact is only valid for a certain time and when it appeared. This information can be used in shortage prediction.

Visualization. Another problem for KGs is to make it accessible for other people. In this case, SC experts might be interested in that. For example, to use it as a graph database and make queries on it. For that, it should be explored in future work how to best visualize the graph. We tried the visualization with Neo4J [111] and it is useful since graph queries can be made easily.

Other crises. In further research, another example of a crisis can be chosen and the approach can be adjusted and applied to it. This would show the generalizability of the shortage-identification system. Moreover, the KG creation method can be adapted and applied to another dataset and a new set of keywords. That would generalize the creation method of a domain-specific KG.

9 CONCLUSION

This research aimed at automatically finding unknown potential Covid-19 shortages in SCs. Based on an ensemble of term weighting schemes, a TM-selected dataset, and an automatically created KG, such a method was developed. The following will summarize the main findings of this research. After that, the main contributions are restated and some recommendations are given.

9.1 Shortage Identification

The first part of this research focused on creating and combining term weighting schemes over time to find potential shortages in text. In the second part, TM selected only relevant articles, and the shortage-identification experiment of the first part was repeated. It was hypothesized that the semantics encoded in a KG improve the performance of the method. Therefore, the last part created a KG and included it in the shortage identification and the experiment was repeated a third time. The following summarizes and compares the best results for each of the three parts. The method was applied to:

Applied to:	**Top Terms**	**Precision**	**Recall**	**F-Score**
1. All data:	100	0.073	0.217	**0.11**
2. TM-selected articles:	100	0.043	0.133	0.065
3. TM-selected articles supported by the KG:	500	0.038	0.461	0.07

The best method according to the F-score is 1., followed by 3. and then 2. That means, applying the method on the entire dataset without the KG performs better than applying the method on the TM-selected dataset with the KG but that still performs better than on the TM-selected dataset without the KG. This gives a possible solution to the overall research questions of *how to automatically find unknown potential Covid-19 related shortages in supply chains in text*. Moreover, it also answers parts of the three sub-questions: *How well can statistical NLP methods alone identify shortages in text? What is the difference in performance when applying the shortage-identification method to the reduced dataset? Does the KG improve the detection of potential shortages?*
Several conclusions can be drawn from the results:

1. **The proposed method actually finds shortages but with low precision.**

   The low precision can be explained with the following reasons: the precision is underestimated because the ground truth list is incomplete, the methods and the ensemble were not tuned, the dataset is biomedical and not SC-related, and finally, the assumption of finding shortages based on frequency and co-occurrence methods could be wrong. When taking a look at the suggested shortages, additional shortages were found. Therefore, the actual precision can be assumed higher.

2. **Topic Modeling did not select a very good dataset.**

   An explanation for the loss in precision from all data to the reduced data is that the TM removed relevant articles as well. That is supported by the fact that only around 38% of the relevant articles in the entire dataset were selected. Even though the performance decreased, the reduced dataset was still kept because: the processing time decreased, it is needed to create a domain-specific KG, to keep the size of the KG manageable saving memory, as a proof of concept methodology for future work, and it saves a lot of human effort.

3. **The recall is significantly higher than the precision for all runs.**

   This shows that a significant amount of the expected shortages were retrieved but also a lot of irrelevant terms. This can be generalized to retrieving more terms relatively leads to more noise than additional shortages. The recall should be given higher importance than the precision because it is easier to filter out irrelevant terms than to add unknown shortages. Moreover, some retrieved terms might still be relevant but are missing in the ground truth list. Therefore, the precision is inaccurate and should not be weighted the same as recall.

4. **The KG improves performance on the same dataset.**

   When comparing the results of the method on the TM-selected dataset without the KG to the results with the KG, a relative average gain of 6% in the F-score can be observed. While the absolute difference in F-score might seem low, the absolute change in recall is quite large. This can again be justified with the low precision in general.

5. **The most similar terms to "mask" and "shortage", and the KG neighbor occurrences of the shortage indicators were the best TWS.**

   These two schemes performed best individually, and as an ensemble. Furthermore, the performance decreased when they were left out. It makes sense that the word embedding performed well because it is based on semantic relatedness and was seeded with a product in shortage. The KG neighbor occurrences performing so well shows the usefulness of the KG. The fact that products in shortage are related to terms that indicate a shortage suggests that relevant triples are part of the KG.

9.2 Topic Modeling to select relevant articles

A KG was constructed to improve the performance of the shortage identification. It should be domain-specific, thus, a domain-specific dataset had to be created. For that, a guided TM approach was used because it was hypothesized that relating texts via TM will outperform keyword searches.
The main findings were:

1. **Topic modeling outperformed the keyword baseline** with an F-score of 0.44 (0.08 higher than the baseline).

   Keyword-search is dependent on direct matches of the keywords in text, which can select some irrelevant articles and miss some relevant ones. TM, on the contrary, selects documents based on their entirety instead of just one term.

2. **The more text is given, the better the topic model**.

   TM on the entire text performed better than on the abstracts. Therefore, in future research, all available information should be used. It was not done in this research because a lot of the entire articles were missing in the dataset.

3. **The estimation of the model performance using the F-score on the seed terms is not a good approach.**

   For hyperparameter tuning, this score was used. However, it correlated negatively with the evaluation based on relevant articles. Therefore, it should not be used in future work.

The answer to the second research question is that TM performs better than keyword selection. In all domains where relevant articles are selected by keywords, TM can be considered as an alternative. Especially, if the articles have to be read by a human. This is due to the low precision of the keyword-based search. That means that many irrelevant articles are selected by keywords which would have to be filtered out manually.

9.3 Automated Creation of a KG

The final part of this thesis gives a methodology to automatically create a domain-specific KG from text. An initial KG was extracted from DBpedia and enhanced by entity types, open RE from text, super-classes, and text processing to cleaning. This methodology is an answer to the final research question of *how such a KG can be created*. The main findings from the creation process are:

1. **The KG contains relevant entities of the domain.**

   Around 70% of the ground truth shortage lists are part of the KG and the most occurring entities are relevant to the domain of Covid-19 shortages.

2. **The KG contains a lot of noise.**

   At least 6% of the entities in the KG are about shortages in SCs. This means that there are a lot of irrelevant details. However, the measure to evaluate the domain affiliation is based on the ground truth list, which is incomplete. Thus, also here the precision is underestimated. Moreover, the entities within the KG which are not directly domain-related might give context around the domain entities. Nonetheless, a quick inspection of the triples showed that some of them are meaningless, incorrect, or contain irrelevant details. Thus, cleaning based on semantics should be done in future work.

3. **Automatic deduplication methods and repeating entity linking to DBpedia were unsuccessful.**

   Repeating the entity linking retrieved a KG which was a large sub-graph from DBpedia containing a large number of irrelevant triples. This makes sense because DBpedia is very large and not well structured. It was a surprise that the deduplication did not find any duplicates. This could mean that the cleaning method already did quite a good job at removing duplicates. However, there are probably still many semantic duplicates in the KG. Therefore, another approach for deduplication should be tested in the future.

4. **67% of the KG is made up of the top 20 relations.**

   The reason for that is that by far the most frequent relation type is “subclass_of” which was added by the super-classes and there are many super-classes. Open RE was used which led to 27 002 different types of relations. Since a large part of the graph is made up of only 20 relation types, it might be possible to use these to form more general predefined relation types.

9.4 Contributions

The following summarizes the main contributions of this research:

- **A technology-driven shortage-identification method**

  Iyengar et al. [7] call for a global early warning system for shortages. The method developed in this research is an approach towards reaching that goal. Furthermore, some studies found that technology-driven methods in SC analysis are scarce and should be explored more, especially for the study of SC disruptions during the Covid-19 pandemic [8, 9]. This thesis is a contribution to that field. Moreover, the method can be generalized to other problems. If the most important terms in certain months of another dataset are looked for, an adjusted version of this method could be applied.

- **A method to select relevant articles automatically**

  SC experts usually use a keyword search to select relevant articles. The developed TM algorithm is an automated approach to do that which saves human effort along with some other advantages. It can be used as a stand-alone approach to reduce a dataset, independently from the rest of the methodology, for other data and domains. Moreover, Bansal et al. [10] stress using TM to study SCs.

- **A method to automatically construct a KG from text**

  Chen et al. [64] identified methods to automatically build KGs without domain-specific labeling as a scarcely researched field. This study is a contribution to that. The proposed method is about the domain of shortages in SCs. It is based on a dataset including the domain and a set of keywords. With some minor adjustments, this method is applicable to any other domain to create a domain-specific KG.

- **A Covid-19 KG of product shortages**

  The review of existing Covid-19 KGs showed that there are other Covid-19 KGs but they are about other domains, e.g. bio-medicine [75, 76]. The developed Covid-19 KG about shortages in SC during the pandemic is a novel KG and can be used in other applications.

9.5 Recommendations

The following will summarize some recommendations for possible future directions:

- **Improving the evaluation methods.**

  The list of ground truth shortages should be largely extended and human evaluated. Alternatively, another approach for the automated evaluation of the predicted terms can be developed. Possibly, the timeframe of the shortages can be included in the evaluation. Moreover, the KG, the list of suggested shortages, and the TM-selected articles should be human evaluated.

- **Extensive tuning of the methods.**

  This work proposes a proof-of-concept methodology. All proposed methods suffer from not being tuned well. For future work, the parameters of the individual methods should be tuned and also the hyperparameters of the ensemble. For the ensemble of TWS and for the KG construction methods that can also mean removing some ineffective ones and adding some new ones. Also, the TM approach should be tuned.

- **Completing the KG.**

  We recommend cleaning the developed KG by methods considering semantics. The aim should be to filter out all noise and irrelevant details and retrieve a compact version of the KG with a high information density. Furthermore, the correctness of the graph should be validated using methods, such as fact-checking. This would not only improve automated methods but also humans that query the KG can profit from a concise and correct representation of the sought information.

- **Adapting the KG.**

  There are two possible adaptions of the KG that might be interesting to the use case. One is to include time within the graph which is interesting for the Covid-19 pandemic because there was a vast amount of rapidly changing information. This means that some facts are only valid for a certain timeframe, which can be encoded within the KG. The second idea is to base the KG on the ground truth list of shortages. That would aim at finding additional shortages that were not known or specific types of a known shortage. Another aim could be to create a KG that knows as much about the domain as possible.

REFERENCES

[1] Vincent C. C. Cheng, Susanna K. P. Lau, Patrick C. Y. Woo, and Kwok Yung Yuen. Severe acute respiratory syndrome coronavirus as an agent of emerging and reemerging infection. *Clinical Microbiology Reviews*, 20(4):660–694, 2007.

[2] Vineet D Menachery, Boyd L Yount, Kari Debbink, Sudhakar Agnihothram, Lisa E Gralinski, Jessica A Plante, Rachel L Graham, Trevor Scobey, Xing-Yi Ge, Eric F Donaldson, et al. A sars-like cluster of circulating bat coronaviruses shows potential for human emergence. *Nature medicine*, 21(12):1508–1513, 2015.

[3] ManMohan S Sodhi, Christopher S Tang, and Evan T Willenson. Research opportunities in preparing supply chains of essential goods for future pandemics. *International Journal of Production Research*, pages 1–16, 2021.

[4] Stephen S Morse. Factors in the emergence of infectious diseases. *Plagues and politics*, pages 8–26, 2001.

[5] Stanley C Oaks Jr, Robert E Shope, Joshua Lederberg, et al. Emerging infections: microbial threats to health in the united states. 1992.

[6] Kate E Jones, Nikkita G Patel, Marc A Levy, Adam Storeygard, Deborah Balk, John L Gittleman, and Peter Daszak. Global trends in emerging infectious diseases. *Nature*, 451(7181):990–993, 2008.

[7] Swathi Iyengar, Lisa Hedman, Gilles Forte, and Suzanne Hill. Medicine shortages: a commentary on causes and mitigation strategies. *BMC medicine*, 14(1):1–3, 2016.

[8] Priyabrata Chowdhury, Sanjoy Kumar Paul, Shahriar Kaisar, and Md Abdul Moktadir. Covid-19 pandemic related supply chain studies: A systematic review. *Transportation Research Part E: Logistics and Transportation Review*, page 102271, 2021.

[9] Maciel M Queiroz, Dmitry Ivanov, Alexandre Dolgui, and Samuel Fosso Wamba. Impacts of epidemic outbreaks on supply chains: mapping a research agenda amid the covid-19 pandemic through a structured literature review. *Annals of operations research*, pages 1–38, 2020.

[10] Pratima Bansal, Jury Gualandris, and Nahyun Kim. Theorizing supply chains with qualitative big data and topic modeling. *Journal of Supply Chain Management*, 56(2):7–18, 2020.

[11] Dmitry Ivanov. Predicting the impacts of epidemic outbreaks on global supply chains: A simulation-based analysis on the coronavirus outbreak (covid-19/sars-cov-2) case. *Transportation Research Part E: Logistics and Transportation Review*, 136:101922, 2020.

[12] Forecasting demand: Quantitative and intuitive techniques. *International Journal of Tourism Management*, 1(1):5–12, 1980.

[13] Amelie Meyer, Wiebke Walter, and Stefan Seuring. The impact of the coronavirus crisis on supply chains and their sustainability: A text mining approach. *Frontiers in Sustainability*, 2:3, 2021.

[14] Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, K. Funk, Rodney Michael Kinney, Ziyang Liu, W. Merrill, P. Mooney, D. Murdick, Devvret Rishi, J. Sheehan, Zhihong Shen, Brandon Brandon Stilson Stilson, Alex D Wade, Kuansan Wang, Christopher Wilhelm, Boya Xie, Douglas M. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. Cord-19: The covid-19 open research dataset. *ArXiv*, 2020.

[15]

[16] Lisa Ehrlinger and W. Wöß. Towards a definition of knowledge graphs. In *SEMANTiCS*, 2016.

[17] Introducing the knowledge graph: things, not strings. URL `https://blog.google/products/search/introducing-knowledge-graph-things-not/`.

[18] Ike Vayansky and Sathish AP Kumar. A review of topic modeling methods. *Information Systems*, 94:101582, 2020.

[19] Pooja Kherwa and Poonam Bansal. Topic modeling: A comprehensive review. *EAI Endorsed Transactions on Scalable Information Systems*, 7(24), 7 2019.

[20] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[21] David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120, 2006.

[22] Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 204–213, 2012.

[23] Arjun Mukherjee and Bing Liu. Aspect extraction through semi-supervised modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 339–348, 2012.

[24] Weizhong Zhao, J. Chen, R. Perkins, Zhichao Liu, W. Ge, Yijun Ding, and Wen Zou. A heuristic approach to determine an appropriate number of topics in topic modeling. *BMC Bioinformatics*, 16:S8 – S8, 2015.

[25] What is wrong with topic modeling? and how to fix it using search-based software engineering. 98.

[26] David Newman, Youn Noh, Edmund Talley, Sarvnaz Karimi, and Timothy Baldwin. Evaluating topic models for digital libraries. In *Proceedings of the 10th annual joint conference on Digital libraries*, pages 215–224, 2010.

[27] David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, page 262–272, USA, 2011. Association for Computational Linguistics.

[28] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic evaluation of topic coherence. In *Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics*, pages 100–108, 2010.

[29] Nikolaos Aletras and Mark Stevenson. Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers*, pages 13–22, 2013.

[30] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, page 399–408, New York, NY, USA, 2015. Association for Computing Machinery.

[31] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8:489–508, 12 2016.

[32] Shaoxiong Ji, Shirui Pan, E. Cambria, P. Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition and applications. *ArXiv*, abs/2002.00388, 2020.

[33] Knowledge graphs and covid-19: Opportunities, challenges, and implementation. *Harvard Data Science Review*. https://hdsr.mitpress.mit.edu/pub/xl0yk6ux.

[34] Gleb Gawriljuk, Andreas Harth, Craig A. Knoblock, and Pedro Szekely. A scalable approach to incrementally building knowledge graphs. In Norbert Fuhr, László Kovács, Thomas Risse, and Wolfgang Nejdl, editors, *Research and Advanced Technology for Digital Libraries*, pages 188–199, Cham, 2016. Springer International Publishing.

[35] Archana Goyal, Vishal Gupta, and Manish Kumar. Recent named entity recognition and classification techniques: a systematic review. *Computer Science Review*, 29:21–43, 2018.

[36] Tareq Al-Moslmi, Marc Gallofré Ocaña, Andreas L Opdahl, and Csaba Veres. Named entity extraction for knowledge graphs: A literature overview. *IEEE Access*, 8:32862–32881, 2020.

[37] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. Luke: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057*, 2020.

[38] Mayank Kejriwal. *Domain-Specific Knowledge Graph Construction*. Springer Publishing Company, Incorporated, 1st edition, 2019.

[39] Meiji Cui, Li Li, Zhihong Wang, and Mingyu You. A survey on relation extraction. In *China Conference on Knowledge Graph and Semantic Computing*, pages 50–58. Springer, 2017.

[40] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74, 2008.

[41] Alisa Smirnova and Philippe Cudré-Mauroux. Relation extraction using distant supervision: A survey. 51(5), November 2018.

[42] Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27 (2):443–460, 2014.

[43] Luigi Bellomarini, Emanuel Sallinger, and Sahar Vahdati. *Chapter 6 Reasoning in Knowledge Graphs: An Embeddings Spotlight*, pages 87–101. Springer International Publishing, Cham, 2020.

[44] Xiaojun Chen, Shengbin Jia, and Yang Xiang. A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications*, 141:112948, 2020.

[45] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

[46] K Bougiatiotis, R Fasoulis, F Aisopos, A Nentidis, and G Paliouras. Guiding graph embeddings using path-ranking methods for error detection innoisy knowledge graphs. *arXiv e-prints*, pages arXiv–2002, 2020.

[47] Matteo PALMONARI and Pasquale MINERVINI. Knowledge graph embeddings and explainable ai. *Knowledge Graphs for Explainable Artificial Intelligence: Foundations, Applications and Challenges*, 47:49, 2020.

[48] Baoxu Shi and Tim Weninger. Discriminative predicate path mining for fact checking in knowledge graphs. *Knowledge-based systems*, 104:123–133, 2016.

[49] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Icml*, 2011.

[50] Forest Gregg and Derek Eder. Github - dedupeio/dedupe: A python library for accurate and scalable fuzzy matching, record deduplication and entity-resolution., 2019. URL `https://github.com/dedupeio/dedupe`.

[51] Application of machine learning techniques for supply chain demand forecasting. *European Journal of Operational Research*, 184(3):1140–1154, 2008.

[52] Dmitry Ivanov. Exiting the covid-19 pandemic: after-shock risks and avoidance of disruption tails in supply chains. *Annals of Operations Research*, pages 1–18, 2021.

[53] Sanjoy Kumar Paul and Priyabrata Chowdhury. A production recovery plan in manufacturing supply chains for a high-demand item during covid-19. *International Journal of Physical Distribution & Logistics Management*, 2020.

[54] Sube Singh, Ramesh Kumar, Rohit Panchal, and Manoj Kumar Tiwari. Impact of covid-19 on logistics systems and disruptions in food supply chain. *International Journal of Production Research*, 59(7):1993–2008, 2021.

[55] Angie Nguyen, Samir Lamouri, Robert Pellerin, Simon Tamayo, and Béranger Lekens. Data analytics in pharmaceutical supply chains: state of the art, opportunities, and challenges. *International Journal of Production Research*, 0(0):1–20, 2021.

[56] Henning Schöpper and Wolfgang Kersten. Using natural language processing for supply chain mapping: a systematic review of current approaches. In *5th International Conference on Computational Linguistics and Intelligent Systems (COLINS 2021)*, number 5, pages 71–86. RWTH Aachen, 2021.

[57] Thanos Papadopoulos, Angappa Gunasekaran, Rameshwar Dubey, Nezih Altay, Stephen J Childe, and Samuel Fosso-Wamba. The role of big data in explaining disaster resilience in supply chains for sustainability. *Journal of Cleaner Production*, 142: 1108–1118, 2017.

[58] Abhinav Khare, Qing He, and Rajan Batta. Predicting gasoline shortage during disasters using social media. *Or Spectrum*, 42(3):693–726, 2020.

[59] Nasser Alsaedi, Pete Burnap, and Omer Rana. Temporal tf-idf: A high performance approach for event summarization in twitter. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 515–521. IEEE, 2016.

[60] Fabian Stephany, Niklas Stoehr, Philipp Darius, Leonie Neuhäuser, Ole Teutloff, and Fabian Braesemann. The corisk-index: A data-mining approach to identify industry-specific risk assessments related to covid-19 in real-time. *arXiv preprint arXiv:2003.12432*, 2020.

[61] Ranganathan Chandrasekaran, Vikalp Mehta, Tejali Valkunde, and Evangelos Moustakas. Topics, trends, and sentiments of tweets about the covid-19 pandemic: Temporal infoveillance study. *Journal of medical Internet research*, 22(10):e22624, 2020.

[62] Saurav Ghosh, Prithwish Chakraborty, Elaine O Nsoesie, Emily Cohn, Sumiko R Mekaru, John S Brownstein, and Naren Ramakrishnan. Temporal topic modeling to assess associations between news trends and infectious disease outbreaks. *Scientific reports*, 7(1): 1–12, 2017.

[63] Ashkan Ebadi, Pengcheng Xi, Stéphane Tremblay, Bruce Spencer, Raman Pall, and Alexander Wong. Understanding the temporal evolution of covid-19 research through machine learning and natural language processing. *Scientometrics*, 126(1):725–739, 2021.

[64] Xieling Chen, Haoran Xie, Zongxi Li, and Gary Cheng. Topic analysis and development in knowledge graph research: A bibliometric review on three decades. *Neurocomputing*, 461:497–515, 2021.

[65] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195, 2015.

[66] Home - dbpedia organization. URL `https://www.dbpedia.org/`.

[67] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[68] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*, 2015.

[69] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, page 1535–1545, USA, 2011. Association for Computational Linguistics.

[70] Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. Yago3: A knowledge base from multilingual wikipedias. In *7th biennial conference on innovative data systems research*. CIDR Conference, 2014.

[71] Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. A comparative survey of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web Journal*, 1(1):1–5, 2015.

[72] Nicolas Heist, S. Hertling, Daniel Ringler, and H. Paulheim. Knowledge graphs on the web - an overview. In *Knowledge Graphs for eXplainable Artificial Intelligence*, 2020.

[73] Covidgraph - main page. URL `https://covidgraph.org/`.

[74] Justin T. Reese, Deepak Unni, Tiffany J. Callahan, Luca Cappelletti, Vida Ravanmehr, Seth Carbon, Kent A. Shefchek, Benjamin M. Good, James P. Balhoff, Tommaso Fontana, Hannah Blau, Nicolas Matentzoglu, Nomi L. Harris, Monica C. Munoz-Torres, Melissa A. Haendel, Peter N. Robinson, Marcin P. Joachimiak, and Christopher J. Mungall. Kg-covid-19: A framework to produce customized knowledge graphs for covid-19 response. *Patterns*, 2(1), 2021.

[75] Franck Michel, Fabien Gandon, Valentin Ah-Kane, Anna Bobasheva, Elena Cabrio, Olivier Corby, Raphaël Gazzotti, Alain Giboin, Santiago Marro, Tobias Mayer, et al. Covid-on-the-web: Knowledge graph and services to advance covid-19 research. In *International Semantic Web Conference*, pages 294–310. Springer, 2020.

[76] Colby Wise, Vassilis N Ioannidis, Miguel Romero Calvo, Xiang Song, George Price, Ninad Kulkarni, Ryan Brand, Parminder Bhatia, and George Karypis. Covid-19 knowledge graph: accelerating information retrieval and discovery for scientific literature. *arXiv preprint arXiv:2007.12731*, 2020.

[77] Bram Steenwinckel, Gilles Vandewiele, Ilja Rausch, Pieter Heyvaert, Pieter Colpaert, Pieter Simoens, Anastasia Dimou, Filip De Turkc, and Femke Ongenae. Facilitating covid-19 meta-analysis through a literature knowledge graph. In *Accepted in Proc. of 19th International Semantic Web Conference (ISWC)*, 2020.

[78] Covid-19: Knowledge graph (starter) | kaggle. URL `https://www.kaggle.com/group16/covid-19-knowledge-graph-starter`.

[79] Taejin Kim, Yeoil Yun, and Namgyu Kim. Deep learning-based knowledge graph generation for covid-19. *Sustainability*, 13(4), 2021.

[80] Github - stko-lab/covid-19-kg. URL `https://github.com/stko-lab/covid-19-kg`.

[81] M. Galkin, S. Auer, Maria-Esther Vidal, and S. Scerri. Enterprise knowledge graphs: A semantic approach for knowledge management in the next generation of enterprise information systems. In *ICEIS*, 2017.

[82] Autoknow: Self-driving knowledge collection for products of thousands of types.

[83] Xuedong Li, Yue Wang, Dongwu Wang, Walter Yuan, Dezhong Peng, and Qiaozhu Mei. Improving rare disease classification using imperfect knowledge graph. In *2019 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 1–2, 2019.

[84] Shan Jiang, Chengxiang Zhai, and Qiaozhu Mei. Exploiting knowledge graph to improve text-based prediction. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1407–1416. IEEE, 2018.

[85] Neo4j Maya Natarajan. How knowledge graphs have transformed the supply chain industry during covid. `https://neo4j.com/blog/how-knowledge-graphs-have-transformed-the-supply-chain-industry-during-covid/`, 12 2021. (Accessed on 01/06/2022).

[86] Luca Costabello, Sumit Pai, Chan Le Van, Rory McGrath, Nicholas McCarthy, and Pedro Tabacof. AmpliGraph: a Library for Representation Learning on Knowledge Graphs, March 2019.

[87] Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings. *Journal of Machine Learning Research*, 22(82): 1–6, 2021. URL http://jmlr.org/papers/v22/20-825.html.

[88] Steven Loria. textblob documentation. *Release 0.15*, 2, 2018.

[89]

[90] Shortages related to the covid-19 pandemic, Aug 2021. URL https://en.wikipedia.org/wiki/Shortages_related_to_the_COVID-19_pandemic.

[91] Usa Today Staff. Pandemic shortages: Products we've struggled to find during covid. *USA TODAY*, 4/9/2021. URL https://eu.usatoday.com/picture-gallery/money/2021/04/09/covid-shortages-these-products-were-high-demand-during-pandemic/7127027002/.

[92] Money Talks News. 9 things that are in short supply in august, 2021. URL https://www.moneytalksnews.com/slideshows/products-now-in-short-supply-due-to-the-pandemic/.

[93] Aluminium Insider. Ball corp and molson coors face aluminium can shortages, continue seeking new resources – aluminium insider, 2021. URL https://aluminiuminsider.com/ball-corp-and-molson-coors-face-aluminium-can-shortages-continue-seeking-new-resources

[94] World Health Organization. Vaccine equity. https://www.who.int/campaigns/vaccine-equity, 2021. (Accessed on 01/29/2022).

[95] Get Us PPE. Ppe shortage data | real-time metrics from get us ppe. https://getusppe.org/data/, 2022. (Accessed on 01/29/2022).

[96] Claude Sammut and Geoffrey I. Webb, editors. *TF–IDF*, pages 986–987. Springer US, Boston, MA, 2010.

[97] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[98] Valerio Di Carlo, Federico Bianchi, and Matteo Palmonari. Training temporal word embeddings with a compass. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6326–6334, 2019.

[99] GitHub. Github - vi3k6i5/guidedlda: semi supervised guided topic model with custom guidedlda, 07/09/2021. URL https://github.com/vi3k6i5/GuidedLDA.

[100] A systematic comparison of search-based approaches for lda hyperparameter tuning. *Information and Software Technology*, 130:106411, 2021.

[101] Simon Blanke. Hyperactive: An optimization and data collection toolbox for convenient and fast prototyping of computationally expensive models. https://github.com/SimonBlanke, since 2019.

[102] Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems*, pages 1–8, 2011.

[103] Herbert Van de Sompel, Michael L. Nelson, Robert Sanderson, Lyudmila L. Balakireva, Scott Ainsworth, and Harihar Shankar. Memento: Time travel for the web, 2009.

[104] George A Miller. *WordNet: An electronic lexical database*. MIT press, 1998.

[105] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, 2015.

[106] Luca Costabello, Sumit Pai, Chan Le Van, Rory McGrath, Nicholas McCarthy, and Pedro Tabacof. Performance — ampligraph 1.4.0 documentation. `https://docs.ampligraph.org/en/1.4.0/experiments.html#predictive-performance`, 2022. (Accessed on 02/02/2022).

[107] Keith Lyons. pandas-dedupe · pypi. `https://pypi.org/project/pandas-dedupe/`, 2022. (Accessed on 02/03/2022).

[108] Thomas Aynaud. community api — community detection for networkx 2 documentation. `https://python-louvain.readthedocs.io/en/latest/api.html`, 2010. (Accessed on 02/03/2022).

[109] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[110] Ilaria Tiddi and Stefan Schlobach. Knowledge graphs as tools for explainable machine learning: A survey. *Artificial Intelligence*, 302:103627, 2022.

[111] Neo4J. Graph data platform | graph database management system | neo4j. `https://neo4j.com/`, 2022. (Accessed on 01/09/2022).

[112] Wikipedia. 2020–present global chip shortage - wikipedia. `https://en.wikipedia.org/wiki/2020%E2%80%93present_global_chip_shortage`, January 2022. (Accessed on 02/08/2022).

A FIRST APPENDIX

A.1 Shortage Terms

name	type	source
abbott	company	expert keyword list
commodity	product (shortage indicator)	expert keyword list
resource	product (shortage indicator)	expert keyword list
goods	product (shortage indicator)	expert keyword list
product	product (shortage indicator)	expert keyword list
purchase	procure (shortage indicator)	expert keyword list
inadequate	shortage synonym (shortage indicator)	expert keyword list
scarcity	shortage synonym (shortage indicator)	expert keyword list
scarceness	shortage synonym (shortage indicator)	expert keyword list
gloves	ppe	expert keyword list
shield	ppe	expert keyword list
lack	shortage synonym (shortage indicator)	expert keyword list
kf94	mask	expert keyword list
3m	company	expert keyword list
glasses	ppe	expert keyword list
produce	procure (shortage indicator)	expert keyword list
producer	procure (shortage indicator)	expert keyword list
seller	procure (shortage indicator)	expert keyword list
logistics	procure (shortage indicator)	expert keyword list
sourcing	procure (shortage indicator)	expert keyword list
cdc	company	expert keyword list
naat	test	expert keyword list
grifols	company	expert keyword list
scarce	shortage synonym (shortage indicator)	expert keyword list
distribute	procure (shortage indicator)	expert keyword list
ppe	ppe	expert keyword list
becton dickinson	company	expert keyword list
face mask	mask	expert keyword list

procurement	procure (shortage indicator)	expert keyword list
fisher paykel	company	expert keyword list
diasorin	company	expert keyword list
biontech	company	expert keyword list
bottleneck	shortage synonym (shortage indicator)	expert keyword list
capacity	stock (shortage indicator)	expert keyword list
stock	stock (shortage indicator)	expert keyword list
unavailable	shortage synonym (shortage indicator)	expert keyword list
deficit	shortage synonym (shortage indicator)	expert keyword list
mask	mask	expert keyword list
deficiency	shortage synonym (shortage indicator)	expert keyword list
respirator	ppe	expert keyword list
bd	company	expert keyword list
ffp2	mask	expert keyword list
kn95	mask	expert keyword list
n95	mask	expert keyword list
shortfall	shortage synonym (shortage indicator)	expert keyword list
unavailability	shortage synonym (shortage indicator)	expert keyword list
shortage	shortage synonym (shortage indicator)	expert keyword list
supply chain	procure (shortage indicator)	expert keyword list
evita	company	expert keyword list
gowns	ppe	expert keyword list
supply	procure (shortage indicator)	expert keyword list
manufacturer	procure (shortage indicator)	expert keyword list
manufacture	procure (shortage indicator)	expert keyword list
manufacturing	procure (shortage indicator)	expert keyword list
supplier	procure (shortage indicator)	expert keyword list
price	procure (shortage indicator)	expert keyword list
personal protective equipment	ppe	expert keyword list
protective equipment	ppe	expert keyword list
hamilton	company	expert keyword list
nucleic acid amplification	test	expert keyword list
inventory	stock (shortage indicator)	expert keyword list
blood test	test	expert keyword list
demand	require (shortage indicator)	expert keyword list
phillips	company	expert keyword list
honeywell	company	expert keyword list
insufficiency	shortage synonym (shortage indicator)	expert keyword list

aprons	ppe	expert keyword list
roche	company	expert keyword list
market	procure (shortage indicator)	expert keyword list
buy	procure (shortage indicator)	expert keyword list
rt pcr	test	expert keyword list
luminex	company	expert keyword list
avellino	company	expert keyword list
peak	increase (shortage indicator)	expert keyword list
request	require (shortage indicator)	expert keyword list
draeger	company	expert keyword list
stockpile	stock (shortage indicator)	expert keyword list
respironics	ppe	expert keyword list
bgi	company	expert keyword list
ventilator	ventilation	expert keyword list
antigen	test	expert keyword list
pcr	test	expert keyword list
polymerase chain reaction	test	expert keyword list
diagnostic test	test	expert keyword list
reagent	test	[90]
swab	test	[90]
sanitizing product	sanitize_syn	[90]
hand sanitiser	sanitize_syn	[90]
hand sanitizer	sanitize_syn	[90]
protective gear	ppe	[90]
surgical mask	mask	[90]
goggle	ppe	[90]
visor	ppe	[90]
protective clothing	ppe	[90]
protective gear	ppe	[90]
ffp3	mask	[90]
medical mask	mask	[90]
medical face shield	ppe	[90]
face shield	ppe	[90]
faceshield	ppe	[90]
mechanical ventilation	ventilation	[90]
ventilation	ventilation	[90]
ecmo	ventilation	[90]

extracorporeal membrane oxygenation	ventilation	[90]
oxygenation	ventilation	[90]
cpap	ventilation	[90]
continuous positive airway pressure	ventilation	[90]
propane	raw_material	[90]
lumber	raw_material	[90]
raw material	raw_material	[90]
steel	raw_material	[90]
inhaler	ventilation	[90]
toilet paper	paper	[90]
freezer	consumer_good	[90]
household appliance	consumer_good	[90]
sewing machine	consumer_good	[90]
jigsaw	consumer_good	[90]
blood donation	blood	[90]
blood donor	blood	[90]
donated blood	blood	[90]
chlorine	chlorine	[90]
paper towel	paper	[90]
tissue paper	paper	[90]
diaper	paper	[90]
production shift	shortage synonym (shortage indicator)	[90]
disinfect	sanitize_syn	[92]
bleach	sanitize_syn	[92]
vaccine	vaccine	
vaccination	vaccine	
johnson	company	
pfizer	company	
moderna	company	
astrazeneca	company	
vaccinate	vaccine	
container	container	[92]
oxygen	ventilation	[92]
gasoline	gas	[92]
nitrile	ppe	nitrile gloves
reduction	shortage synonym (shortage indicator)	

hoarding	shortage synonym (shortage indicator)	
stockpiling	stock (shortage indicator)	
short supply	shortage synonym (shortage indicator)	
chip	chip_shortage	[112]
conductor	chip_shortage	[112]
semiconductor	chip_shortage	[112]
microchip	chip_shortage	[112]
intel	chip_shortage	[112]
cotton mask	mask	[14]
oropharyngeal nasopharyngeal swab	swab	[14]
polyester nasal swab	swab	[14]
ventilator circuit	ventilation	[14]
quantitative reverse transcriptase polymerase chain reaction	test	[14]
rtqpcr	test	[14]
pcv13	vaccine	[14]
bnt162b2	vaccine	[14]
heating ventilation system	ventilation	[14]
hvac	ventilation	[14]
taqman	test	[14]
thermofisher inc	test	[14]
stainless steel	raw_material	[14]
conjunctival swab	swab	[14]
immunoreagent	test	[14]
smart mask	mask	[14]
tozinameran	vaccine	[14]
filter mask	mask	[14]
water disinfection system	chlorine	[14]
chadox1s	vaccine	[14]
alinity m	test	[14]
facepiece mask	mask	[14]
adenovirus serotype vaccine	vaccine	[14]
cloth mask	mask	[14]
ad26cov2s	vaccine	[14]
plastic container	raw_material	[14]
hocl	chlorine	[14]
hypochlorous acid	chlorine	[14]

abhs	sanitize_syn	[14]
npop	swab	[14]
protective mask	mask	[14]
cleanser	sanitize_syn	[14]
disinfectant nanospray	sanitize_syn	[14]
rapid antigenic test	test	[14]
clo2	chlorine	[14]
chlorine dioxide	chlorine	[14]
disinfectant sprayer	sanitize_syn	[14]
covishield	vaccine	[14]
molecular diagnostic test	test	[14]
rhinooropharyngeal swab	swab	[14]
throat swab	swab	[14]
beckman coulter	test	[14]
polyester swab	swab	[14]
valve mask	mask	[14]
acetonitrile	ppe	[14]
pharyngeal swab	swab	[14]
frsm	mask	[14]
fluid resistant surgical mask	mask	[14]
fluid repellent surgical mask	mask	[14]
sfm	mask	[14]
surgical face mask	mask	[14]
copper	raw_material	[14]
coal	raw_material	[14]
petroleum	gas	[14]
plastic	raw_material	[14]
drug	medicine	[14]
tea	food	[14]
sugar	food	[14]
rice	food	[14]
caffeine	food	[14]
dairy	food	[14]
natural products	raw_material	[14]
coffee	food	[14]

B SECOND APPENDIX

B.1 Removed Terms in Preprocessing

Paper terms: ”paper”, ”study”, ”studies”, ”abstract”, ”purpose”, ”background”, ”introduction”, ”objective”, ”article”, ”review”, ”research”, ”authors”, ”author”, ”analysis”, ”objectives”, ”present_study”, ”conclusion”, ”conclusions”, ”methods”, ”results”, ”online version”, ”supplementary material”, ”electronic supplementary material”
Covid-19 synonyms: ”covid”, ”corona”, ”sarscov”, ”cov2”, ”covid19_virus”, ”sars”, ”cov”, ”covid19”, ”covid19_pandemic”, ”coronavirus”, ”coronavirus_disease”, ”sarscov2”, ”disease”, ”pandemic”, ”novel_coronavirus”, ”severe_acute_respiratory_syndrome_coronavirus_2”, ”coronavirus_disease_covid19_pandemic”, ”acute_syndrome_coronavirus_sarscov2”, ”acute_respiratory_syndrome_coronavirus_sarscov2”, ”acute_respiratory_syndrome_coronavirus”, ”severe_acute_respiratory_syndrome_coronavirus”, ”coronavirus_disease_covid19”, ”sarscov2_infection”, ”syndrome_coronavirus_sarscov2”, ”respiratory_syndrome_coronavirus_sarscov2”, ”novel_coronavirus_disease_covid19”, ”coronavirus_sarscov2”, ”novel_coronavirus_sarscov2”, ”sarscov2_pandemic”, ”acute_respiratory_syndrome_coronavirus2_sarscov2”, ”coronavirus_pandemic”, ”coronavirus_covid19_pandemic”, ”pandemic_of_coronavirus_disease”, ”acute_syndrome_coronavirus2_sarscov2”, ”syndrome_coronavirus_sarscov2_pandemic”, ”syndrome_coronavirus2_sarscov2”, ”coronaviruses”, ”acute_respiratory_syndrome_coronavirus_sarscov2_pandemic”, ”respiratory_syndrome_coronavirus_sarscov2_pandemic”, ”respiratory_syndrome_coronavirus2_sarscov2”, ”acute_syndrome_coronavirus_sarscov2_pandemic”, ”covid19_disease”, ”coronavirus_covid19”, ”the_coronavirus_disease”, ”coronavirus_disease2019_covid19”, ”the_coronavirus_pandemic”

C THIRD APPENDIX

C.1 Results of all methods for the entire dataset

Method	Top Terms	Precision	Recall	F-score	Time in seconds
Term Frequency	100	0.010	0.006135	0.007605	171.9
Term Frequency	500	0.014	0.042424	0.021053	169.1
Term Frequency	1000	0.012	0.071429	0.020548	148.2
DF-IDF	100	0.050	0.029940	0.037453	138.9
DF-IDF	500	0.034	0.095506	0.050147	139.4
DF-IDF	1000	0.032	0.166667	0.053691	117.2
Context Frequency	100	0.050	0.030120	0.037594	519.4
Context Frequency	500	0.032	0.090395	0.047267	510.3
Context Frequency	1000	0.029	0.153439	0.048780	437.7
Context Difference Frequency	100	0.040	0.024242	0.030189	3.2
Context Difference Frequency	500	0.034	0.095506	0.050147	3.4
Context Difference Frequency	1000	0.030	0.157068	0.050378	2.8
Context DF-IDF	100	0.010	0.006135	0.007605	539.4
Context DF-IDF	500	0.024	0.069767	0.035714	532.8
Context DF-IDF	1000	0.023	0.127778	0.038983	454.8
Context Difference DF-IDF	100	0.000000	0.000000	0.000000	0.8
Context Difference DF-IDF	500	0.016	0.047337	0.023916	0.7
Context Difference DF-IDF	1000	0.017	0.095506	0.028862	0.7
Word Embedding shortage indicators	100	0.020	0.012195	0.015152	8.9

Method	Top Terms	Precision	Recall	F-score	Time in seconds
Word Embedding shortage indicators	500	0.024	0.070588	0.035821	10.0
Word Embedding shortage indicators	1000	0.019	0.107955	0.032313	11.6
Word Embedding ”mask”, ”shortage”	100	0.290	0.155914	0.202797	0.6
Word Embedding ”mask”, ”shortage”	500	0.174	0.356557	0.233871	0.7
Word Embedding ”mask”, ”shortage”	1000	0.139	0.474403	0.215004	0.8

C.2 Results of all methods for the reduced dataset

Method	Top Terms	Precision	Recall	F-score	Time in seconds
Term Frequency	100	0.030	0.018182	0.022642	41.4
DF-IDF	100	0.030	0.018182	0.022642	31.2
Context Frequency	100	0.050	0.029940	0.037453	107.6
Context Difference Frequency	100	0.040	0.024096	0.030075	0.7
Context DF-IDF	100	0.010	0.006135	0.007605	109.0
Context Difference DF-IDF	100	0.000	0.000000	0.000000	0.2
Word Embedding shortage indicators	100	0.000	0.000000	0.000000	5.0
Word Embedding ”mask”, ”shortage”	100	0.130	0.075581	0.095588	0.4
Term Frequency	500	0.016	0.048193	0.024024	40.7
DF-IDF	500	0.024	0.069364	0.035661	32.2
Context Frequency	500	0.034	0.094972	0.050074	103.5
Context Difference Frequency	500	0.034	0.094972	0.050074	0.7
Context DF-IDF	500	0.020	0.058480	0.029806	108.6
Context Difference DF-IDF	500	0.018	0.052632	0.026826	0.2

Method	Top Terms	Precision	Recall	F-score	Time in seconds
Word Embedding shortage indicators	500	0.020	0.058480	0.029806	4.2
Word Embedding ”mask”, ”shortage”	500	0.070	0.183246	0.101302	0.5
Term Frequency	1000	0.016	0.093023	0.027304	40.2
DF-IDF	1000	0.022	0.120219	0.037194	31.8
Context Frequency	1000	0.036	0.182741	0.060150	102.9
Context Difference Frequency	1000	0.036	0.182741	0.060150	0.7
Context DF-IDF	1000	0.018	0.101124	0.030560	109.1
Context Difference DF-IDF	1000	0.017	0.096045	0.028887	0.2
Word Embedding shortage indicators	1000	0.016	0.090909	0.027211	5.3
Word Embedding ”mask”, ”shortage”	1000	0.068	0.307692	0.111384	0.6

C.3 Results of all methods for the reduced dataset and KG

Method	Number of Top Terms	Top Terms	Precision	Recall	F-score
Term Frequency	100	Entities	0.030	0.018	0.023
TF-IDF	100	Entities	0.010	0.006	0.008
Context Frequency	100	Entities	0.000	0.000	0.000
Context Difference Frequency	100	Entities	0.000	0.000	0.000
Context TF-IDF	100	Entities	0.010	0.006	0.008
Context Difference TF-IDF	100	Entities	0.000	0.000	0.000
Word Embedding shortage indicators	100	Entities	0.010	0.006	0.008
Word Embedding ”mask”, ”shortage”	100	Entities	0.170	0.096	0.123
KG Neighbor Term Frequency	100	Entities	0.010	0.006	0.008
KG Neighbor TF-IDF	100	Entities	0.020	0.012	0.015
Link Prediction	100	Entities	0.020	0.012	0.015
Term Frequency	500	Entities	0.018	0.054	0.027
TF-IDF	500	Entities	0.014	0.042	0.021

Context Frequency	500	Entities	0.016	0.048	0.024
Context Difference Frequency	500	Entities	0.016	0.048	0.024
Context TF-IDF	500	Entities	0.010	0.030	0.015
Context Difference TF-IDF	500	Entities	0.008	0.024	0.012
Word Embedding shortage indicators	500	Entities	0.020	0.059	0.030
Word Embedding ”mask”, ”shortage”	500	Entities	0.096	0.238	0.137
KG Neighbor Term Frequency	500	Entities	0.010	0.030	0.015
KG Neighbor TF-IDF	500	Entities	0.086	0.209	0.122
Link Prediction	500	Entities	0.016	0.048	0.024
Term Frequency	1000	Entities	0.011	0.065	0.019
TF-IDF	1000	Entities	0.009	0.055	0.015
Context Frequency	1000	Entities	0.015	0.087	0.026
Context Difference Frequency	1000	Entities	0.015	0.086	0.026
Context TF-IDF	1000	Entities	0.012	0.071	0.021
Context Difference TF-IDF	1000	Entities	0.008	0.048	0.014
Word Embedding shortage indicators	1000	Entities	0.018	0.103	0.031
Word Embedding ”mask”, ”shortage”	1000	Entities	0.073	0.327	0.119
KG Neighbor Term Frequency	1000	Entities	0.015	0.089	0.026
KG Neighbor TF-IDF	1000	Entities	0.055	0.252	0.090
Link Prediction	1000	Entities	0.020	0.012	0.015
Term Frequency	100	Entities, replaced by super class	0.030	0.018	0.023
TF-IDF	100	Entities, replaced by super class	0.010	0.006	0.008
Context Frequency	100	Entities, replaced by super class	0.020	0.012	0.015
Context Difference Frequency	100	Entities, replaced by super class	0.020	0.012	0.015
Context TF-IDF	100	Entities, replaced by super class	0.010	0.006	0.008
Context Difference TF-IDF	100	Entities, replaced by super class	0.000	0.000	0.000
Word Embedding shortage indicators	100	Entities, replaced by super class	0.010	0.006	0.008

Word Embedding ”mask”, ”shortage”	100	Entities, replaced by super class	0.170	0.096	0.123
KG Neighbor Term Frequency	100	Entities, replaced by super class	0.010	0.006	0.008
KG Neighbor TF-IDF	100	Entities, replaced by super class	0.020	0.012	0.015
Link Prediction	100	Entities, replaced by super class	0.020	0.012	0.015
Term Frequency	500	Entities, replaced by super class	0.012	0.037	0.018
TF-IDF	500	Entities, replaced by super class	0.012	0.037	0.018
Context Frequency	500	Entities, replaced by super class	0.014	0.043	0.021
Context Difference Frequency	500	Entities, replaced by super class	0.010	0.030	0.015
Context TF-IDF	500	Entities, replaced by super class	0.010	0.030	0.015
Context Difference TF-IDF	500	Entities, replaced by super class	0.008	0.024	0.012
Word Embedding shortage indicators	500	Entities, replaced by super class	0.020	0.059	0.030
Word Embedding ”mask”, ”shortage”	500	Entities, replaced by super class	0.096	0.238	0.137
KG Neighbor Term Frequency	500	Entities, replaced by super class	0.010	0.030	0.015
KG Neighbor TF-IDF	500	Entities, replaced by super class	0.086	0.209	0.122
Link Prediction	500	Entities, replaced by super class	0.016	0.048	0.024
Term Frequency	1000	Entities, replaced by super class	0.006	0.037	0.010
TF-IDF	1000	Entities, replaced by super class	0.009	0.055	0.015
Context Frequency	1000	Entities, replaced by super class	0.009	0.055	0.015
Context Difference Frequency	1000	Entities, replaced by super class	0.008	0.048	0.014

Context TF-IDF	1000	Entities, replaced by super class	0.011	0.067	0.019
Context Difference TF-IDF	1000	Entities, replaced by super class	0.010	0.061	0.017
Word Embedding shortage indicators	1000	Entities, replaced by super class	0.018	0.103	0.031
Word Embedding ”mask”, ”shortage”	1000	Entities, replaced by super class	0.073	0.327	0.119
KG Neighbor Term Frequency	1000	Entities, replaced by super class	0.015	0.089	0.026
KG Neighbor TF-IDF	1000	Entities, replaced by super class	0.055	0.252	0.090
Link Prediction	1000	Entities, replaced by super class	0.020	0.012	0.015
Term Frequency	100	Objects	0.030	0.018	0.023
TF-IDF	100	Objects	0.000	0.000	0.000
Context Frequency	100	Objects	0.030	0.018	0.023
Context Difference Frequency	100	Objects	0.010	0.006	0.008
Context TF-IDF	100	Objects	0.000	0.000	0.000
Context Difference TF-IDF	100	Objects	0.000	0.000	0.000
Word Embedding shortage indicators	100	Objects	0.010	0.006	0.008
Word Embedding ”mask”, ”shortage”	100	Objects	0.170	0.096	0.123
KG Neighbor Term Frequency	100	Objects	0.010	0.006	0.008
KG Neighbor TF-IDF	100	Objects	0.020	0.012	0.015
Link Prediction	100	Objects	0.036	0.012	0.018
Term Frequency	500	Objects	0.014	0.042	0.021
TF-IDF	500	Objects	0.006	0.018	0.009
Context Frequency	500	Objects	0.014	0.042	0.021
Context Difference Frequency	500	Objects	0.016	0.048	0.024
Context TF-IDF	500	Objects	0.004	0.012	0.006
Context Difference TF-IDF	500	Objects	0.004	0.012	0.006
Word Embedding shortage indicators	500	Objects	0.020	0.059	0.030
Word Embedding ”mask”, ”shortage”	500	Objects	0.096	0.238	0.137
KG Neighbor Term Frequency	500	Objects	0.010	0.030	0.015

KG Neighbor TF-IDF	500	Objects	0.086	0.209	0.122
Link Prediction	500	Objects	0.033	0.048	0.040
Term Frequency	1000	Objects	0.014	0.083	0.024
TF-IDF	1000	Objects	0.007	0.042	0.012
Context Frequency	1000	Objects	0.016	0.092	0.027
Context Difference Frequency	1000	Objects	0.013	0.076	0.022
Context TF-IDF	1000	Objects	0.005	0.030	0.009
Context Difference TF-IDF	1000	Objects	0.005	0.030	0.009
Word Embedding shortage indicators	1000	Objects	0.018	0.103	0.031
Word Embedding ”mask”, ”shortage”	1000	Objects	0.073	0.327	0.119
KG Neighbor Term Frequency	1000	Objects	0.015	0.089	0.026
KG Neighbor TF-IDF	1000	Objects	0.055	0.252	0.090
Link Prediction	1000	Objects	0.036	0.012	0.018
Term Frequency	100	Objects, replaced by super class	0.030	0.018	0.023
TF-IDF	100	Objects, replaced by super class	0.000	0.000	0.000
Context Frequency	100	Objects, replaced by super class	0.020	0.012	0.015
Context Difference Frequency	100	Objects, replaced by super class	0.020	0.012	0.015
Context TF-IDF	100	Objects, replaced by super class	0.000	0.000	0.000
Context Difference TF-IDF	100	Objects, replaced by super class	0.000	0.000	0.000
Word Embedding shortage indicators	100	Objects, replaced by super class	0.010	0.006	0.008
Word Embedding ”mask”, ”shortage”	100	Objects, replaced by super class	0.170	0.096	0.123
KG Neighbor Term Frequency	100	Objects, replaced by super class	0.010	0.006	0.008
KG Neighbor TF-IDF	100	Objects, replaced by super class	0.020	0.012	0.015
Link Prediction	100	Objects, replaced by super class	0.036	0.012	0.018

Term Frequency	500	Objects, replaced by super class	0.010	0.031	0.015
TF-IDF	500	Objects, replaced by super class	0.006	0.018	0.009
Context Frequency	500	Objects, replaced by super class	0.010	0.031	0.015
Context Difference Frequency	500	Objects, replaced by super class	0.008	0.025	0.012
Context TF-IDF	500	Objects, replaced by super class	0.004	0.012	0.006
Context Difference TF-IDF	500	Objects, replaced by super class	0.004	0.012	0.006
Word Embedding shortage indicators	500	Objects, replaced by super class	0.020	0.059	0.030
Word Embedding ”mask”, ”shortage”	500	Objects, replaced by super class	0.096	0.238	0.137
KG Neighbor Term Frequency	500	Objects, replaced by super class	0.010	0.030	0.015
KG Neighbor TF-IDF	500	Objects, replaced by super class	0.086	0.209	0.122
Link Prediction	500	Objects, replaced by super class	0.033	0.048	0.040
Term Frequency	1000	Objects, replaced by super class	0.008	0.049	0.014
TF-IDF	1000	Objects, replaced by super class	0.006	0.037	0.010
Context Frequency	1000	Objects, replaced by super class	0.008	0.049	0.014
Context Difference Frequency	1000	Objects, replaced by super class	0.008	0.049	0.014
Context TF-IDF	1000	Objects, replaced by super class	0.007	0.043	0.012
Context Difference TF-IDF	1000	Objects, replaced by super class	0.006	0.037	0.010
Word Embedding shortage indicators	1000	Objects, replaced by super class	0.018	0.103	0.031
Word Embedding ”mask”, ”shortage”	1000	Objects, replaced by super class	0.073	0.327	0.119

KG Neighbor Term Frequency	1000	Objects, replaced by super class	0.015	0.089	0.026
KG Neighbor TF-IDF	1000	Objects, replaced by super class	0.055	0.252	0.090
Link Prediction	1000	Objects, replaced by super class	0.036	0.012	0.018

C.4 Results of all settings combining all methods on the reduced dataset with the KG

Top Terms	Origin	Precision	Recall	F-score	Number of Types
100	Entities	0.036	0.144	0.058	5
500	Entities	0.037	0.477	0.069	11
1000	Entities	0.030	0.592	0.057	13
100	Entities replaced by super-classes	0.034	0.134	0.054	5
500	Entities replaced by super-classes	0.036	0.465	0.067	11
1000	Entities replaced by super-classes	0.029	0.578	0.055	13
100	Objects	0.038	0.144	0.060	6
500	Objects	0.038	0.461	0.070	11
1000	Objects	0.030	0.574	0.058	13
100	Objects replaced by super-classes	0.034	0.128	0.053	5
500	Objects replaced by super-classes	0.037	0.452	0.069	11
1000	Objects replaced by super-classes	0.029	0.560	0.056	13

Table C.4: Results of the last experiment for all settings combining all methods