GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Master's Thesis
submitted in partial fulfillment of the
requirements for the course "Applied Data Science"

Self-supervised Domain Adaptation of Language Models for the Process Industry

Jonas Lührs

Institute of Computer Science

Master's Thesis
of the Center for Computational Sciences
at the Georg-August-Universität Göttingen

21. October 2024

Georg-August-Universität Göttingen
Institute of Computer Science

Goldschmidtstraße 7
37077 Göttingen
Germany

☎ +49 (551) 39-172000
FAX +49 (551) 39-14403
✉ office@informatik.uni-goettingen.de
🌍 www.informatik.uni-goettingen.de

First Supervisor:	Dr. Terry Ruas
Second Supervisor:	Prof. Dr. Bela Gipp
Thesis Advisor:	Anastasia Zhukova

I hereby declare that I have written this thesis independently without any help from others and without the use of documents or aids other than those stated. I have mentioned all used sources and cited them correctly according to established academic citation rules.

Göttingen, 21. October 2024

J. Lührs

Abstract

Incorporating additional knowledge into pre-trained language models (PLMs) has proven to be highly effective in improving their performance in specialized fields. Graph structures, in particular, allow models to capture domain-specific relationships between documents that would otherwise be missed. In the process industry, where unstructured text logs document critical operational insights, leveraging these relationships becomes essential for improving document representation. This thesis proposes a graph-aware domain adaptation method aimed at enhancing the representations of PLMs for the process industry. Building upon SciNCL, a neighborhood contrastive learning approach, the study constructs a process industry graph comprising functional locations and maintenance text logs to sample document pairs for contrastive learning. The proposed method outperformed baseline models in a semantic search task, with the best-performing model achieving an nDCG@10 score 9 points higher than the best baseline. These findings encourage further exploration of graph-based domain adaptation techniques, particularly in domains characterized by sparse document connections and limited data availability.

Acknowledgements

I would like to extend my heartfelt thanks to the various parties whose collaboration and support were vital to the completion of this thesis.

The idea for this project was provided by Anastasia Zhukova, a doctoral student from the Chair for Scientific Information Analytics of Prof. Bela Gipp at the Georg-August-Universität Göttingen. I am especially grateful for her dedication to my success in writing this thesis and for supporting me in all my research endeavors.

I am indebted to Prof. Akiko Aizawa for inviting me to do research at the National Institute of Informatics (NII) in Tokyo. I thank Prof. Akiko Aizawa, and Prof. Florian Boudin for giving valuable feedback during my stay at NII. I also thank my fellow NII interns for the insightful discussions.

I gratefully acknowledge the German Academic Exchange Service (DAAD) for their scholarship, which allowed me to research at the NII.

Dr. Malte Ostendorff implemented the SciNCL approach, which formed the basis of my research, as part of his PhD thesis.

A company that develops software for the process industry provided the data and computing resources. The models were evaluated on a benchmark provided by the same company.

Finally, and most importantly, I thank my family and friends for supporting me to achieve my goals.

Contents

1 Introduction **1**

1.1 Problem Description . . . 2

1.2 Research Objective . . . 2

1.3 Thesis Outline . . . 3

2 Background and Related Work **4**

2.1 Pre-trained Language Models . . . 4

2.2 Domain Adaptation . . . 7

2.3 Process Industry . . . 10

2.4 Knowledge-enriched Domain Adaptation . . . 13

3 Methodology **16**

3.1 Process Industry Graph . . . 17

3.2 Graph Embeddings . . . 17

3.3 Triplet Sampling . . . 18

3.4 Contrastive Learning Objective . . . 20

3.5 Graph-aware Domain Adaptation . . . 21

4 Implementation **23**

4.1 Data . . . 23

4.2 Data Pre-processing . . . 26

4.3 Graph . . . 27

4.4 Graph Embedding Model . . . 28

4.5 Triplet Sampling . . . 32

4.6 Language Model . . . 33

5 Evaluation **35**

5.1 Process Industry Applications Benchmark . . . 35

5.2 Results and Discussion . . . 38

5.2.1 Influence of Contrastive Fine-tuning . . . 38

5.2.2 Influence of Different Datasets 40
5.2.3 Influence of Domain Model 42
5.2.4 Influence of Data Enrichment 44
5.3 Broader Discussion 46
5.4 Limitation and Future Work 48

6 Conclusion **50**

Bibliography **62**

A Code and Data **63**

B AI Usage Statement **64**

List of Figures

2.1 BERT for text classification 12

3.1 Graph-aware Domain Adaptation 16
3.2 A heterogeneous process industry graph with directed edges 17
3.3 SciNCL: Controlled nearest neighbor sampling over graph embeddings for contrastive learning 19

4.1 Vocabulary overlap between tenants 24

5.1 Vocabulary overlap between tenants and benchmark datasets 42

List of Tables

2.1 Overview of German language models 7
2.2 Overview of German domain-specific language models 9
2.3 Overview of knowledge-enriched fine-tuning methods 15

3.1 Scientific vs process industry domain 21

4.1 Overview of process industry datasets 24
4.2 Text log examples 25
4.3 Data enrichment 26
4.4 Data quality filters 27
4.5 Analysis of process industry graphs before data enrichment 28
4.6 Analysis of process industry graphs after data enrichment 28
4.7 PyTorch BigGraph parameters 29
4.8 PyTorch BigGraph train-test split 30
4.9 Exemplary hyperparameter tuning of PyTorch BigGraph 31
4.10 Evaluation of graph embedding models 31
4.11 Analysis of triplets 33

5.1 Process Industry Applications (PIA) benchmark 36
5.2 Influence of contrastive fine-tuning reported on PIA benchmark 39
5.3 Influence of different datasets reported on PIA benchmark 41
5.4 Influence of domain model reported on PIA benchmark 43
5.5 Influence of data enrichment reported on PIA benchmark 45

List of Abbreviations

ANN	Approximate Nearest Neighbor	32
CP	Continual Pre-training	8
fNER	Funcloc-NER	37
KE	Knowledge Embedding	14
KG	Knowledge Graph	13
KNN	K-Nearest Neighbors	20
MLM	Masked Language Modeling	5
NER	Named Entity Recognition	9
NLG	Natural Language Generation	4
NLI	Natural Language Inference	36
NLP	Natural Language Processing	1
NLU	Natural Language Understanding	4
NSP	Next Sentence Prediction	5
PC	Priority Classification	35
PGB	PyTorch-BigGraph	28
PIA	Process Industry Applications	35
PSC	Problem-Solution Classification	35
PSE	Problem-Solution Extraction	36
PLM	Pre-trained Language Model	1
RL	Record Linking	36
RTD	Replaced Token Detection	5
SemS	Semantic Search	36
SSL	Self-supervised Learning	8
SSP	Sentence Splitting	37
S2ORC	Semantic Scholar Open Research Corpus	26
TLP	Technical Language Processing	11
VA	Vocabulary Adaptation	8
WWM	Whole Word Masking	6

Chapter 1

Introduction

In recent years, we have observed great advances in Natural Language Processing (NLP). At the center of this development are so-called Pre-trained Language Models (PLMs) like BERT [1], RoBERTa [2], or BART [3]. Those models were trained on large datasets using vast computing and time resources. Transfer learning allows us to leverage the knowledge of these models by fine-tuning them for new tasks or on new datasets [4]. This approach can produce well-performing models with smaller datasets and less training time. Thus, fine-tuning PLMs has become the leading practice in NLP [5].

Domain adaptation is a special case of transfer learning and focuses on fine-tuning PLMs on domain-specific data. PLMs are usually pre-trained on a large general-domain language corpus like Wikidata [6]. During the training, they learn universal language representations, which are useful for various downstream tasks. However, when applied to specialized domains, they do not perform as well as models pre-trained on domain data [7]. So why do we not train a domain-specific model from scratch every time?

Large efforts have been put into training domain-specific models from scratch such as BioBERT [8] for biology, SciBERT [9] for the scientific field, and ClinicalBERT [10] for the medical domain, leading to new state-of-the-art on their domain-specific downstream tasks. However, in-domain pre-training is not feasible for every domain, considering the large costs and amounts of data necessary [11]. Especially, since even small specializations can be considered separate domains [12]. Domain adaptation aims at improving the knowledge of a general language PLM by fine-tuning it on domain-specific data by applying methods suitable for the available amount of domain data [5].

1.1 Problem Description

In manufacturing industries such as the process industry, valuable employee knowledge is often described textually [13], PLMs allow to leverage this knowledge for decision support [14] or predictive maintenance [15]. However, no domain-specific PLM exists for the process industry, making the adaptation of general-purpose PLMs crucial for enhancing downstream task performance. Considering the lack of labeled data in real-world scenarios, unsupervised and self-supervised domain adaptation methods achieve the widest applicability. When the amount of unlabeled text data is scarce, self-supervised methods propose to utilize additional data sources, e.g., underlying graph structures, to support the language model training.

Knowledge graphs capture how documents are connected by representing documents as nodes and their relationships as edges [16]. Introducing this information into the training process of a language model enables it to learn inter-document relatedness. Since the pre-training of language models is usually done on token- or sentence-level objectives, learning document-level semantics can improve the model's language representation [17]. Ostendorff et al. [18] improved the vector representation of documents in the scientific domain by combining text data with a citation graph for a contrastive learning method to learn citation similarity. The authors found that even training a general-domain language model this way outperformed baselines pre-trained in-domain.

1.2 Research Objective

This work aims to improve the vector representation of the domain-specific language of the process industry, focusing on German language models. To do so, I will use SciNCL, a contrastive learning approach proposed by Ostendorff et al. [18] to learn the similarity between documents of the process industry. In this project, I will evaluate whether transferring this methodology to a different domain — in this case, the process industry, and specifically in the context of German language data — can enhance performance in domain-specific downstream applications, especially document-level ones like text classification or semantic search.

In chemical plants, a functional location tree describes the machinery the plant consists of. Every machine, and even its parts can be considered an individual functional location. Using a plant management software, employees report their daily operations, e.g. maintenance activities. These unstructured text reports ("text logs") contain information about functional locations involved in these activities. This project will use the assumption that if two logs mention the same functional locations, they are similar. Since text logs have a direct connection to a graph structure, i.e. the machinery graph, they can be used for the graph-based contrastive learning approach proposed by Ostendorff et al.

Motivated by the successful application of Ostendorff et al.'s approach in the scientific domain, the following research question was formulated:

How can we transfer and adapt a graph-based contrastive learning approach for the domain adaptation of PLMs for the process industry?

To answer this research question, the following research tasks were defined:

Task 1: Develop a methodology for transferring and adapting the approach of Ostendorff et al. [18], i.e., graph-based contrastive learning for improving scientific representations, to the process industry

Task 2: Implement the proposed methodology for the process industry

Task 3: Evaluate the methodology on a benchmark of the process industry

1.3 Thesis Outline

In Chapter 1, I highlight the need for domain adaptation and the challenge of applying it to narrow domains like the process industry. Based on this problem setting the research objective was defined.

In Chapter 2, I describe background knowledge and related work for this thesis. Relevant research areas are Pre-trained Language Models with a focus on German models and self-supervised domain adaptation techniques. Next, I define the process industry as the application domain, discussing its language characteristics and relevant NLP use cases. Finally, I explore domain adaptation approaches applicable to the process industry, highlighting the potential of incorporating additional knowledge, such as graph structures, as a promising research direction.

In Chapter 3, I propose a graph-aware domain adaptation methodology for the process industry that uses SciNCL as its core. I present the heterogeneous process industry graph, and the sampling scheme to create positive and negative samples for contrastive learning. Lastly, I summarize the key differences between applying SciNCL to the scientific domain and the process industry.

In Chapter 4, I describe the implementation of the proposed methodology. I give an overview of the training data and parameters for the graph embedding and language models.

In Chapter 5, I evaluate the proposed methodology on a domain benchmark of the process industry. The benchmark covers a wide range of NLP tasks like text classification, semantic search, and token classification. I conduct an ablation study by analyzing the influence of different datasets, fine-tuning an already domain-adapted PLM, and further enriching the training data. I discuss the results and provide an outlook for future work.

Finally, in Chapter 6, I conclude the thesis with a summary.

Chapter 2

Background and Related Work

In this chapter, I provide an overview of the research field of this master's thesis. This includes definitions of the relevant terminology used throughout the work. The project is located at the intersection of two areas within NLP: PLMs, especially German models, and the domain adaptation of PLMs focusing on self-supervised methods.

In Section 2.1, I briefly introduce PLMs and provide an overview of the German foundation models available. Next, in Section 2.2, I describe self-supervised domain adaptation methods and their application in adapting German language models to specific domains. In Section 2.3, I characterize the application domain in this work, the process industry, by giving a definition, introducing the language characteristics, discussing NLP use cases, and how PLMs are applied to them. Lastly, in Section 2.4, I discuss domain adaptation approaches suited to the process industry, emphasizing the use of additional knowledge, such as underlying graph structures of the text, as a promising research direction.

2.1 Pre-trained Language Models

PLMs have significantly advanced NLP by offering general-purpose models that can be fine-tuned for a wide range of tasks. PLMs are deep neural networks that learn universal language representations from large volumes of text data using self-supervised training methods. Central to their success is the transformer architecture [19], which allows efficient training and contextual text understanding. Transformer-based PLMs can be divided into three categories: Encoder-only models [1], decoder-only models [20], and encoder-decoder models [3]. While encoder-only models were developed for Natural Language Understanding (NLU) tasks like text classification, decoder-only models are typically used for Natural Language Generation (NLG) like text generation. In this work, I focus on encoder-only models as the targeted downstream tasks do not contain any NLG tasks. I begin by exploring influential English encoder-only PLMs, which have not only advanced the field of NLP but also laid the groundwork for the German foundation models in use today.

The release of BERT [1] marks the start of the development of several powerful encoder-only PLMs. The key innovation of BERT is to apply the bidirectional training of the transformer architecture to language modeling. This is done by introducing a new pre-training task called Masked Language Modeling (MLM), where the model needs to predict the masked words in a sentence given the left and the right context. Since the relationship between two sentences is not directly captured by the MLM objective, the authors introduce a second task called Next Sentence Prediction (NSP) where the model needs to distinguish whether two input sentences are continuous segments from the training corpus. BERT was the first fine-tuning-based representation model that achieved state-of-the-art performance on a large range of sentence-level and token-level tasks, outperforming many task-specific architectures.

Motivated by the success of BERT, researchers developed similar language models based on it. RoBERTa [2] enhances the training process of BERT by optimizing the training parameters, increasing the training data, and dropping the NSP pre-training objective. It also slightly adopts the MLM objective, by introducing dynamic masking: tokens are masked differently at each epoch, whereas BERT does it once and for all.

ELECTRA [21] introduces a more sample-efficient pre-training task called Replaced Token Detection (RTD), where instead of masking input tokens, some are replaced with plausible alternatives generated by a small generator network. Then, a discriminative model is trained to identify which tokens in the corrupted sequence have been replaced, allowing for gradient updates across all input tokens rather than just a masked subset, as in MLM.

DeBERTa [22] improves the pre-training efficiency and performance of BERT and RoBERTa models by using two novel techniques: disentangled attention and an enhanced mask decoder. The disentangled attention mechanism represents each input word using two vectors that encode its content and position, respectively, and the attention weights among words are computed using disentangled matrices on their contents and relative positions, respectively. The enhanced mask decoder incorporates absolute positions in the decoding layer to predict the masked tokens in model pre-training.

Although the models were initially trained for the English language, research efforts have focused on applying them to other languages, including German. In the following section, I will examine German encoder-only models in more detail, with an overview provided in Table 2.1.

In 2019, the first German BERT models were published by deepset (deepset-BERT[1]) and the MDZ Digital Library team (DBMDZ) at the Bavarian State Library (dbmdz-BERT[2]) outperforming mBERT [1], a multilingual BERT variant, on various German NLP tasks.

In 2020, deepset and DBMDZ published together two new German models GBERT and GELECTRA, based on the BERT and ELECTRA models [23]. Compared to the previous two German models

[1]https://deepset.ai/german-bert
[2]https://github.com/dbmdz/berts#german-bert

which were trained on around 15 GB, the new models were trained on 163 GB of data and were published in a base and a large version. GBERT is trained by replacing the MLM objective by Whole Word Masking (WWM) which masks all subword tokens corresponding to a word at once, instead of just randomly masking a single subword, making the prediction of subword tokens of a word, that had been split into multiple subword tokens, more challenging. The GBERT and GELECTRA models were found to outperform the two previous German BERT models on all of the three evaluated NLP tasks.

In the same year, Scheible et al. [24] published GottBERT, a German RoBERTa model trained on 143 GB of data. GottBERT outperforms deepset-BERT and dbmdz-BERT on three of five downstream tasks.

In 2022, German WECHSEL-RoBERTa was published by Minixhofer et al. [25]. Instead of training a German model from scratch, they transferred the English RoBERTa [2] to the German language by effectively initializing subword embeddings for cross-lingual transfer. The model was evaluated together with GBERT on two downstream tasks, where they found that their model was outperforming GBERT in both cases.

In 2023, Dada et al. published GeBERTa [26], a German model based on DeBERTa-v2, together with a comprehensive benchmark of eight downstream datasets. They evaluated GeBERTa together with previous German models such as GBERT, GELECTRA, and GottBert. They found GeBERTa outperforming those models on all downstream tasks.

Recently, a German language understanding evaluation benchmark known as SuperGLEBer [27] was published, comprising 29 tasks across various categories, including document classification, sequence tagging, sentence similarity, and question answering. The authors evaluate four previously introduced models: deepset-BERT, GBERT-base, GBERT-large, and GottBERT. The findings revealed that while GBERT-large generally achieved the highest performance, it was occasionally matched or surpassed by smaller models, with GBERT-base emerging as the best-performing base-size model overall.

Alongside the development of pure German language models, there has also been a rise in multilingual models that incorporate German data among other languages. A prominent example is mBERT, developed by the original BERT authors, along with other significant models like XLM-R [28] and MBART [29]. However, exploring multilingual models is outside the scope of this thesis and is reserved for future work.

Model	Year	Architecture	Data Source	Data Size
deepset-BERT	2019	BERT	German Wikipedia, OpenLegal-Data [30] and News	12 GB
dbmdz-BERT	2019	BERT	German Wikipedia, EU Book-shop corpus, Open Subtitles, CommonCrawl, ParaCrawl and News Crawl	16 GB
GottBERT [24]	2020	RoBERTa	German OSCAR corpus [31]	145 GB
GBERT* [23]	2020	BERT	German OSCAR corpus [31], German Wikipedia, OPUS [32], and OpenLegalData [30]	163 GB
GELECTRA* [23]	2020	ELECTRA	*Same as GBERT*	163 GB
German WECHSEL-RoBERTa [25]	2022	RoBERTa	German OSCAR corpus [31]	4 GB
GeBERTa* [26]	2023	DeBERTa-v2	Formal (Wikipedia, News), informal, legal (OpenLegalData [30]), medical, and literature data	167 GB

Table 2.1: Overview of German language models. Models marked with an asterisk (*) also have a large version.

Summary The introduced PLMs achieve state-of-the-art performance on various NLP tasks. While these models were initially published for English, substantial efforts have been made to leverage these advancements for German NLP by training pure German models. Nonetheless, the language barrier is just one factor affecting the applicability of PLMs. Another important issue is a possible domain shift between the task domain and the pre-training domain which can lead to a performance drop of the PLM when applied to the downstream tasks of the task domain [5].

2.2 Domain Adaptation

Different approaches have emerged to fine-tune general-purpose PLMs for specific domains, categorized by the amount of labeled data required [5]. Supervised and semi-supervised methods rely on fully or partially labeled datasets, while unsupervised and self-supervised approaches require no labels, making them more broadly applicable where labeled data is scarce. In the following section, I introduce self-supervised domain adaptation methods.

Continual Pre-training (CP) The most common approach for domain adaptation of PLMs is to continue the pre-training on an unlabeled domain corpus [4]. This second phase of pre-training allows the model to learn about the characteristics of the domain language, resulting in better domain-specific text representations. By carefully choosing the right self-supervised pre-training task, different learning goals can be emphasized. Self-supervised Learning (SSL) can be classified into generative, contrastive, and adversarial approaches [33].

Generative SSL allows the model to learn by decoding the encoded input. An example is the MLM [1] task where a language model predicts the masked tokens based on the unmasked tokens. Another example is TSDAE [34] which encodes corrupted sentences into fixed-sized vectors and requires the decoder to reconstruct the original sentences from these sentence embeddings.

Contrastive SSL allows the model to learn by making comparisons. It is commonly used to further improve the model by introducing sentence-level or document-level semantics. The language model is trained to distinguish between pairs of positive and negative texts [35]. NSP [1] requires the model to identify if the given sentence pair includes consecutive sentences or not. SimCSE [36] applies different dropout masks to the same sentence to create positive pairs and uses other sentences of the same batch as negatives.

Adversarial SSL allows the model to learn by identifying whether the tokens in the input sentence are replaced, shuffled, or randomly substituted. An example is the RTD task of ELECTRA [21].

Hybrid SSL combines more than one type of SSL. For example, the BERT model uses generative (MLM) and contrastive (NSP) SSL to learn word-level as well as sentence-level semantics.

Vocabulary Adaptation (VA) The vocabulary of general-purpose PLMs may not capture the diverse terminology of a specific domain. Because of this, researchers have created domain-specific vocabularies when training a PLM on domain data from scratch [9, 37], added additional domain-specific words to the existing vocabulary before continuing the pre-training on a domain corpus [38, 39], or updated the vocabulary during fine-tuning on downstream data [40].

Knowlegde-enrichment Another line of research aims to improve the domain knowledge of PLMs by adding more context through additional data sources. K-BERT [41] adds triplets from a domain knowledge graph to the training data, providing more information about entities within the text. SciNCL [18] uses citation information extracted from a citation graph to identify similar and dissimilar papers for contrastive learning, improving document-level representations.

After introducing several domain adaptation approaches, I will now discuss studies that adapted German PLMs to specialized domains. Table 2.2 shows an overview of German domain-specific language models, including the domain, the training approach, and the training data.

Domain	Year	Model	Training	Data source	Data size
Clinical	2022	BioGottBERT [42]	CP GottBERT	A biomedical corpus	0.8 GB
Clinical	2022	BioELECTRA [42]	PT ELECTRA	A biomedical corpus	0.8 GB
Clinical	2024	GBERT-Clinical* [43]	CP GBERT	Clinical documents	25M docs
Clinical	2024	GeBERTa-Clinical* [43]	CP GeBERTa	Clinical documents	25M docs
Medical	2024	GBERT-BioM-Translation* [43]	CP GBERT	Translated medical texts from PubMed	45M docs
Medical	2023	medBERT.de [37]	PT BERT + VA	Medical texts from a hospital and scientific texts	10 GB
JobsAds	2022	jobGBERT [39]	CP of GBERT	German job advertisements	0.23 GB
JobsAds	2022	jobBERTde [39]	CP + VA dmbdz-BERT	German job advertisements	0.23 GB
Financial	2024	Financial GBERT [44]	CP GBERT	FinCorpus-DE10k [45]	0.9 GB
Parlamentary debates	2022	ParlBERT [46]	CP deepset-BERT	DeuParl corpus [47]	2.55 GB

Table 2.2: Overview of German domain-specific language models. Abbreviations for training methods: pre-training (PT), continual pre-training (CP), and vocabulary adaptation (VA). Models marked with an asterisk (*) also have a large version.

I find that most attempts to train German domain-specific models were made for the clinical domain. Lentzen et al. [42] experimented with adapting GottBERT [24] to the biomedical and clinical domain by applying continual pre-training on a small biomedical corpus (BioGottBERT). They trained an ELECTRA model [21] from scratch (BioELECTRA) on the same data, but found it was underperforming because the amount of pre-training data was insufficient. They compared these domain models with general-purpose models like GBERT or GELECTRA [23] on biomedical Named Entity Recognition (NER) and document classification tasks, identifying GBERT as the best-performing model for document classification. Idrissi-Yaghir et al. [43] also applied continual pre-training to train several German medical language models using GBERT and GeBERTa as base models. They evaluated them on various downstream tasks, including NER, multi-label classification, and extractive question answering. They found that continuously pre-trained models match or even exceed the performance of clinical models trained from scratch.

Gnehm et al. [39] applied continual pre-training on German job advertisements and could improve over the general-domain language models in all evaluated tasks, namely text classification, token-level sequence labeling, and NER.

Kozaeva et al. [44] trained several models for the financial domain using different data configurations and evaluated them on a text classification and an NER task. While continual pre-training of GBERT outperformed the vanilla GBERT in some cases, this finding was not consistent over all configurations.

Klamm et al. [46] continually pre-trained deepset-BERT on German parliamentary debates (ParlBERT). They evaluated both models on a topic classification task. While there was no substantial improvement over the base model, ParlBERT yielded higher results on low-frequency topics.

Summary While domain adaptation of PLMs is an active research field, it was shown that continual pre-training with the MLM objective is the dominant method when domain-adapting German language models. However, the success depends on the amount of domain data available, i.e. the more distant the domain, the more data is required. Since data is often scarce, sample-efficient methods or methods leveraging additional data sources are a promising research direction.

2.3 Process Industry

In this section, I examine NLP applications in the process industry, focusing on the use of PLMs for industrial plant maintenance. I define the process industry, identify the types of text data it generates, review current NLP research, and explore key use cases, especially those leveraging PLMs for improving maintenance processes.

The process industry refers to companies that process raw materials to manufacture finished or semi-finished products through physical, mechanical, or chemical processes [3]. The production involves a continuous flow of materials. Examples of process industries include food, beverages, chemicals, and pharmaceuticals.

An industrial plant produces vast amounts of data, this can be numerical data like sensor data from machines, but also text data like maintenance reports. Maintenance reports are usually free-form text entries recorded by maintenance operators. These log entries contain vast amounts of tacit knowledge [14]. However, until recently this data source was underutilized [48]. The reason is that these reports are characterized by domain-specific technical terms, abbreviations [49] as well as incomplete sentences, typographical errors, and non-standard notations [50] which create

[3] https://www.ipa.fraunhofer.de/en/industry-solutions/process-industry.html

challenges for applying classical NLP methods developed for "standard" language (e.g. newspaper texts).

Brundage et al. [51] proposed the field of Technical Language Processing (TLP) which aims to transfer classical NLP methods to the technical industry setting. Addressing the challenges of technical texts, several TLP pre-processing pipelines were developed [13, 49]. A recent literature review [52] showed that transformer-based models like BERT have not been widely applied to industrial maintenance yet. However, given their advantage of being less dependent on classical pre-processing pipelines, I find several works that explore their application to industrial downstream tasks.

As one of the earliest studies, Usuga-Cadavid et al. [53] used CamemBERT [54], a French BERT version, to predict the criticality and duration of a maintenance issue from the description provided. They compared the model with a smaller range of classical and feature-based approaches and found that CamemBERT outperformed them. In a following study [15], they extended their evaluation to another French BERT model, FlauBERT [55], and compared the two models with a broader range of classic machine learning approaches. Similar to their first study, the transformer-based models achieved the best results with minimum text pre-processing and hyperparameter tuning. In a similar study, Naqvi et al. [48] fine-tuned CamemBERT to predict the problem category of maintenance logs, outperforming traditional NLP approaches.

In a more recent study, Naqvi et al. [14] explored how PLMs like BERT [1] can be adapted to enable semantic search in industrial text. They compared three self-supervised domain-adaptation techniques to fine-tune BERT for the target domain: SimCSE [36], TSDAE [34] and contrastive tension [56]. They found that TSDAE can efficiently identify intricate patterns in the industrial text regardless of associated complexities, outperforming the other fine-tuning techniques.

Figure 2.1 depicts an example of a BERT model fine-tuned to a text classification task, predicting whether the description of a machine problem will stop production or not (severity prediction). BERT tokenizes the input text, adds a special classification token "[CLS]", and processes it through transformer layers to capture contextualized information. The final hidden state of the [CLS] token represents the entire input and is passed to a linear layer for classification. A softmax function produces probabilities for each class, leading to the final prediction.

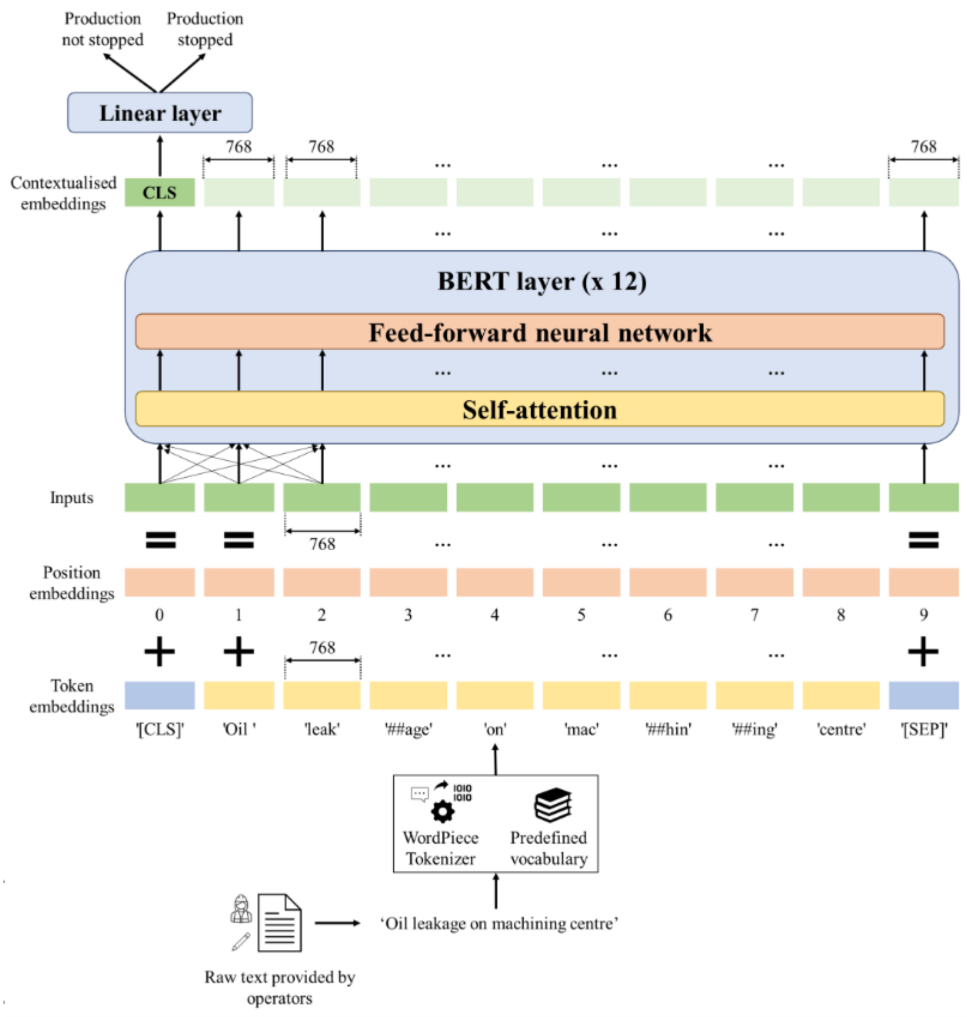


Figure 2.1: BERT for text classification [15].

The presented studies focus on two NLP use cases, predictive maintenance and semantic search. Predictive maintenance aims at predicting machine failures to reduce machine downtime while semantic search allows employees to quickly retrieve critical information from vast amounts of text data. Another important NLP use case is knowledge extraction [57] which focuses on identifying relevant information from unstructured text. By extracting asset names and maintenance reasons from maintenance logs, previously unstructured information can be captured in more easily readable formats, such as knowledge graphs [58].

Compared to the vast number of documents available in domains like science, publicly accessible resources in manufacturing are limited [52]. Companies often withhold internal data to protect privacy or maintain a competitive edge, leading to research being conducted on non-disclosed datasets [53]. The process industry, as a subdomain of manufacturing, is affected by the same constraints.

Most of the works did not adapt the general-purpose model to the target domain before fine-tuning it to the target tasks [15, 48, 53]. This is likely because their focus was on comparing BERT with classical machine learning methods, rather than on improving the performance of BERT itself. While Naqvi et al. [14] explore different domain adaptation techniques, the chosen methods do not consider any other resources besides the given text, only learning sentence-level semantics. However, for narrow domains like the process industry, leveraging additional information like graph structures to integrate further domain knowledge into the model training is a promising research direction.

Summary The process industry offers several NLP use cases like predictive maintenance [15], knowledge extraction [57], and semantic search [14]. To fully leverage PLMs in this technical domain, this work aims at improving the domain representation of German models through domain adaptation. In the next section, I will explore knowledge-enriched domain adaptation techniques well-suited to the process industry.

2.4 Knowledge-enriched Domain Adaptation

Since general-purpose PLMs lack domain-specific knowledge, several approaches have been developed to incorporate additional information from external knowledge bases into PLMs to improve the domain language representation [59]. Widespread sources of knowledge are graphs, e.g., an academic graph gives information on which papers cite each other or were written by the same authors, indicators for document similarity.

Knowledge Graphs (KGs) encapsulate rich structural information, and knowledge embedding techniques enable the transformation of this information into continuous embeddings for entities and relations [60].

K-BERT [41] injects triples extracted from a knowledge graph into sentences as domain knowledge. To prevent a change of the original sentence meaning due to too much incorporated knowledge (known as knowledge noise), K-BERT introduces soft positioning and visible matrix which limits the impact of knowledge.

ERNIE [61] improves text representation by linking entity embeddings from a knowledge graph with the corresponding entity mentioned in the text. KnowBERT [62] follows a similar approach by integrating an entity linker to retrieve relevant entity embeddings and training jointly the entity linker and self-supervised language modeling objective in an end-to-end fashion.

KEPLER [60] integrates factual knowledge into the language presentation by introducing a Knowledge Embedding (KE) objective. By combining the KE objective with the classical MLM objective the goal is to align the factual knowledge and language representation into the same semantic space.

The previously introduced works incorporate knowledge by injecting it directly into the training data in the form of entity embeddings or text. Another research direction leverages the underlying graph structure of documents as a similarity signal to learn document-level semantics, also known as graph-aware language model fine-tuning [63]. For example, papers citing each other are assumed to be semantically similar.

SPECTER [17] applies a contrastive learning approach to learn task-independent representations of scientific documents. To do so, they use the information about the citations between papers to generate citation-informed triplets for training. A positive paper is cited by the query paper, while a negative paper is either not cited by the query paper (easy negative) or cited by a positive paper (hard negative). The authors fine-tune SciBERT [9] with a triplet loss.

LinkBERT [64] uses document hyperlinks for PLM training. The authors propose a document relation prediction objective that aims to classify the relation of two node text pairs as contiguous, random, and linked. The linked pairs are sampled from a document graph. They experimented with hyperlinks of Wikipedia articles (general domain) and citation links of academic articles (biomedical domain).

SciNCL [18] improves on the work from SPECTER. Instead of sampling triplets directly from the citation links, they train a graph embedding model on a citation graph, allowing them to sample positives and negatives on a continuous scale. Positive papers are sampled close to a given query paper in the citation embedding space, while negatives are sampled more distant from the query paper. By introducing a sample-induced margin between positives and negatives, the authors are able to sample hard-to-learn negatives. Like SPECTER they fine-tune SciBERT. They achieved state-of-the-art results on the SciDocs benchmark [17]. They found that even a small amount of triplets leads to an improvement on the benchmark tasks which is interesting for a narrow domain like the process industry. They also found that the fine-tuned model outperforms the model pre-trained in-domain, emphasizing the suitability of their approach for domain adaptation.

TwHIN-BERT [65] introduces a contrastive social loss to train BERT on Twitter data. After constructing a Heterogeneous Information Network [66] from the engagement logs of users and Tweets (TwHIN), they train a graph embedding model to capture co-engagement and map Tweets and users into a vector space. Similar to Ostendorff et al., they mine positive node pairs based on graph embeddings, i.e. a positive tweet is close to the query tweet in the embedding space. However, they do not apply a scheme for sampling hard-to-learn negatives, but just randomly sample negatives.

MICoL [67] applies metadata-induced contrastive learning using a Heterogeneous Information Network. A positive sample is a document that is reachable from the query document within a given meta-path [68] or meta-graph [69]. Negatives are randomly sampled from the whole corpus. Using a discrete path makes their work similar to SPECTER but they do not consider hard-to-learn negatives. Table 2.3 gives an overview of the introduced techniques that use external data for domain adaptation.

Approach	Graph / Domain	Training Objective
K-BERT [41]	CN-DBpedia KG, HowNet KG, MedicalKG	MLM+NSP
ERNIE [61]	Wikidata KG	MLM+NSP+dEA
KnowBERT [62]	Wikipedia KB, Wordnet KG	MLM+NSP+EL
KEPLER [60]	Wikidata5M KG, Wordnet KG	MLM+KE
SPECTER [17]	Citation Graph	CL
LinkBERT [64]	Wikipedia Graph, Citation Graph	MLM+DRP
SciNCL [18]	Citation Graph	CL
TWHIN-BERT [65]	Twitter Graph	MLM+CL
MICoL [67]	Academic Graph	CL

Table 2.3: Overview of knowledge-enriched fine-tuning methods. Abbreviations used: Contrastive Learning (CL), Denoising Entity Auto-Encoder (dEA), Document Relation Prediction (DRP), Entity Linker (EL), Knowledge Embeddings (KE), Knowledge Graph (KG), Knowledge Base (KB), Masked Language Modeling (MLM), and Next Sentence Prediction (NSP).

Summary After reviewing various knowledge-enriched and graph-aware fine-tuning approaches, SciNCL [18] emerges as the most suitable for adapting a pre-trained language model to the process industry. Unlike methods that rely on discrete document links [17, 64, 67], SciNCL leverages a graph embedding space for sampling similar and dissimilar documents. Moreover, compared to approaches like TwHIN-BERT [65], which randomly samples negatives, SciNCL's sophisticated sampling strategy includes hard-to-learn negatives which are known to improve contrastive learning [70, 71].

Chapter 3

Methodology

In this chapter, I propose a methodology for self-supervised domain adaptation of language models for the process industry, by leveraging underlying graph structures of text data. It builds upon SciNCL, a contrastive learning approach introduced by Ostendorff et al. [18] for the scientific domain, which has been reworked and tailored to the process industry. Figure 3.1 shows the four components of the proposed methodology which will be described in the following sections.

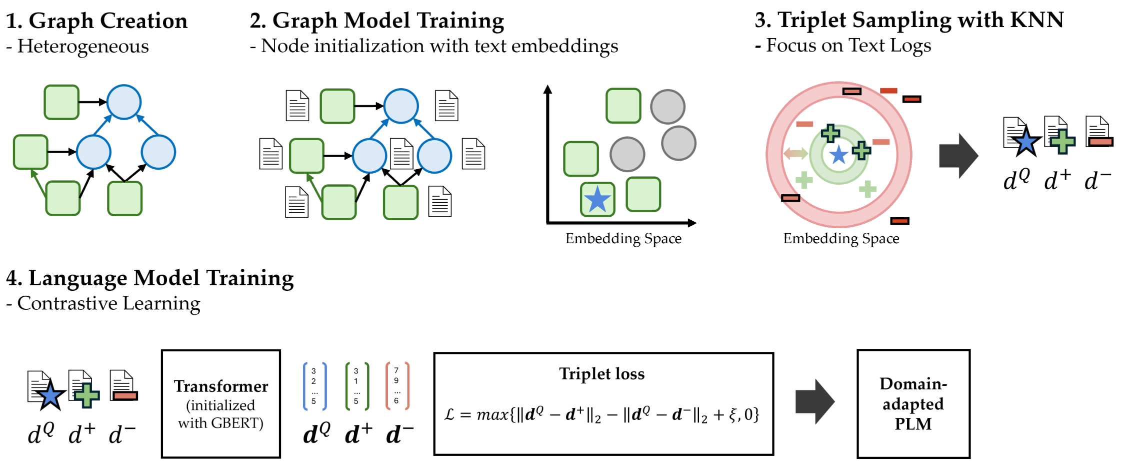

1. Graph Creation
- Heterogeneous
2. Graph Model Training
- Node initialization with text embeddings
Embedding Space
3. Triplet Sampling with KNN
- Focus on Text Logs
Embedding Space
d^Q d^+ d^-
4. Language Model Training
- Contrastive Learning
d^Q d^+ d^-
Transformer (initialized with GBERT)
$\boldsymbol{d}^Q$ $\boldsymbol{d}^+$ $\boldsymbol{d}^-$
Triplet loss
$\mathcal{L} = max\{\|\boldsymbol{d}^Q - \boldsymbol{d}^+\|_2 - \|\boldsymbol{d}^Q - \boldsymbol{d}^-\|_2 + \xi, 0\}$
Domain-adapted PLM

Figure 3.1: Graph-aware Domain Adaptation.

In Section 3.1, I define the heterogeneous process industry graph used in this work. In Section 3.2, I motivate the training of a graph embedding model to capture the domain-specific relationships. Next, in Section 3.3, I outline how the triplets are sampled from the graph embedding space. In Section 3.4, I describe the contrastive learning objective used for fine-tuning the language model on the triplets. Finally in Section 3.5, I summarize the differences between the scientific and process industry domains and the proposed changes to the methodology of Ostendorff et al.

3.1 Process Industry Graph

An industrial plant consists of multiple functional locations (i.e. processing units, machines, and other parts). Thus, the entire plant can be described in the form of a functional location tree. When workers report about their daily operations at the plant, they usually mention functional locations involved in these activities. Besides functional locations, these "text logs" can also refer to older reports that are related to the current one. For example, if an employee reports a solution to an earlier reported problem.

Based on these relationships within the process industry data, I define three edge types. First, a *is_part_of* relation between functional locations (funclocs). Second, a *reports_about* relation between a text log and a funcloc. Lastly, a *follows* relation between text logs. All of these edges are directed, having a source and a target node. I expect that combining all three relation types will lead to a well-connected graph, effectively capturing the domain-specific relationships within the process industry data. A graph consisting of multiple node and edge types is referred to as a heterogeneous graph in graph theory. In comparison, Ostendorff et al. [18] use a unipartite graph, i.e. only consisting of one node type (paper) and one relation type (citation). Figure 3.2 shows an exemplary graph containing the defined node and relationship types.

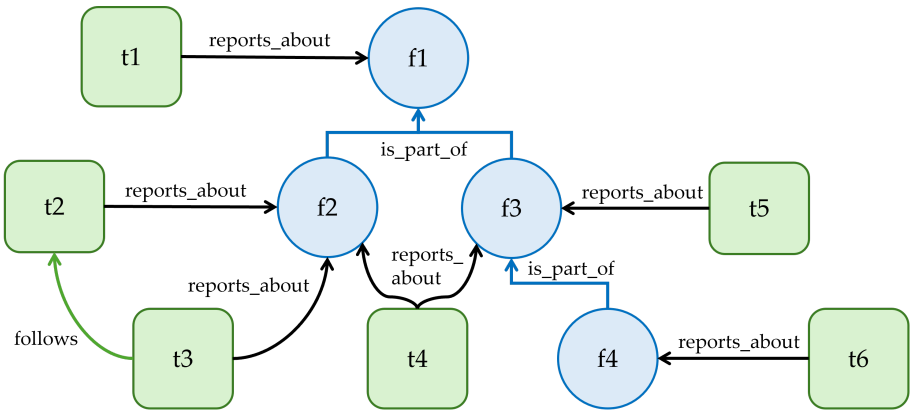


Figure 3.2: A heterogeneous process industry graph with directed edges. Two node types: Text log (t) and funcloc (f) nodes. Three edge types: *follows* (green), *reports_about* (black) and *is_part_of* (blue).

3.2 Graph Embeddings

To capture the domain-specific relationships of the process industry graph, I train a graph embedding model. During training, the model learns to position nodes connected by an edge closer together in the embedding space. Typically, the initial values for node embeddings are generated randomly. However, to enhance the learning process, I initialize the node embeddings with text embeddings. To do so, I use a text encoder model to generate text embeddings from the texts connected to the graph nodes.

This approach is similar to the strategy employed by Asada et al. [72], where the authors use text embeddings to support learning a graph embedding model in areas where the graph's structural information is insufficient. They use PubMedBERT [73], a BERT model pre-trained on biomedical text data, to encode textual information of a heterogeneous pharmaceutical knowledge graph into embeddings.

In the absence of a domain-specific model for the process industry, I will use a general German PLM to encode the text. Although this model may not produce domain-specific embeddings as effectively as Asada et al.'s approach, it will capture the semantics of the text logs and functional location (funcloc) descriptions, providing valuable insights for the graph training process. For instance, initializing funcloc nodes with their descriptions will naturally position similar funcloc types (e.g., pumps) closer together in the embedding space. During training, text logs reporting about different pumps are more likely to be placed in a similar location within the embedding space since the funclocs of the same type were placed close together right from the start. Thus, text logs that do not share the same funclocs but are from a semantically similar context can still end up with similar node embeddings. I expect that combining the domain-specific relationships with semantic similarity will lead to high-quality text log embeddings that can be used for sampling similar and dissimilar text documents.

Upon learning dense representations of nodes in the process industry graph, I utilize the text log representations to sample context-similar text logs.

3.3 Triplet Sampling

In contrastive learning, the effectiveness of the model relies heavily on the quality of triplet sampling. A triplet consists of a query document d^Q, a positive (similar) document d^+, and a negative (dissimilar) document d^-. Ostendorff et al. [18] use the graph embedding neighborhood around a query document vector $\mathbf{d}^Q$ to sample similar (positive) and dissimilar (negative) documents by selecting the k nearest neighbors. Graph embeddings provide a continuous and undirected similarity signal that allows us to find semantically similar nodes, even without direct edges between those nodes in the graph. The continuous scale makes it possible to define hard-to-learn positives as well as hard-to-learn negatives.

In a heterogeneous graph setup, sampling from graph embeddings simplifies identifying similar and dissimilar nodes. The generated node embeddings inherently capture the diverse contexts provided by various relation types, thereby eliminating the necessity for complex sampling schemes that directly build positive samples from graph edges [67], which might work in unipartite setups [17] but becomes impractical in heterogeneous ones.

Figure 3.3 visualizes a graph embedding space, where potential 'query, positive, negative' triplets $(\mathbf{d}^Q, \mathbf{d}^+, \mathbf{d}^-)$ are represented by (★, ✚, ▬).

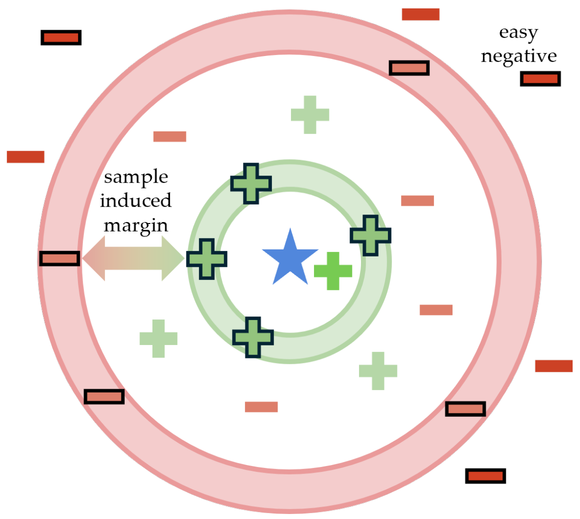


Figure 3.3: SciNCL: Controlled nearest neighbor sampling over graph embeddings for contrastive learning. Starting with a query document ★ in the graph embedding space, hard-to-learn positives ✚ are embeddings from a close context (green band), yet distant enough to prevent gradient collapse. Hard-to-learn negatives ▬ are close to positives within a sample-induced margin (red band), while easy negatives ▬ are very distant from the query document. [18]

Positive Samples d^+ should be semantically similar to the query document d^Q, but not too similar to prevent gradient collapse. Moreover, positives should be sampled from a comparable distance to the query embedding $\mathbf{d}^Q$ [74]. To account for this, Ostendorff et al. sample positive (similar) documents by first defining the distance to the query embedding $\mathbf{d}^Q$ with k^+ and then selecting the c^+ nearest neighbors $(k^+ - c^+, k^+)$ visualized by the green band in Figure 3.3.

Negative Samples d^- should be semantically dissimilar from the query document d^Q, i.e., sampled distant to the query embedding $\mathbf{d}^Q$. The sampling of hard-to-learn negatives, i.e., negatives that are close to potential positives, has been found to improve contrastive learning [70,71]. However, the collision of negative and positive samples can make the learning signal noisy [75]. To avoid collisions, Ostendorff et al. introduce a sample-induced margin using the k parameter. This margin allows to sample hard negatives (red band in Figure 3.3) that do not collide with the sampled positives (green band). To produce a diverse similarity signal for contrastive learning, they recommend sampling a mix of hard and easy negatives. Easy negatives are document embeddings that are even more distant from the query embedding than hard negatives (outside of the red band).

Sampling Strategies Ostendorff et al. [18] propose three strategies for sampling triplets. Sampling close documents from the graph embeddings space (KNN), using random sampling, or by using both strategies (filtered random). For each strategy, c' denotes the number of samples.

K-Nearest Neighbors (KNN) Given a graph embedding model (e.g., PyTorch BigGraph [76]) denoted as f_{gem}, document node embeddings $\mathbf{D}$, and a search index (e.g., FAISS [77]), a k-nearest neighbors search $KNN(f_{gem}(\mathbf{d}^Q), \mathbf{D})$ is performed. From the range of neighbors around the query document d^Q, c' samples are selected using the interval $(k - c', k]$, where the neighbors $N = \{n_1, n_2, n_3, \ldots\}$ include n_i as the i-th nearest neighbor in the graph embedding space. For example, if $c' = 3$ and $k = 10$, the selected samples would be the three neighbors: n_8, n_9, and n_{10}.

Random Sample c' documents from the corpus without replacement.

Filtered Random Similar to random sampling but excluding the documents retrieved by KNN, i.e., all neighbors within the largest k are omitted.

Like Ostendorff et al., I use KNN sampling for positives and hard negatives and filtered random sampling for easy negatives.

3.4 Contrastive Learning Objective

Given the textual content of a document d (text log), the objective is to create a dense vector representation $\mathbf{d}$ that effectively encodes the document's information and is applicable for downstream tasks. A Transformer language model (e.g., GBERT [23]) f_{lm} encodes documents d into vector representations $f_{lm}(d) = \mathbf{d}$.

While Ostendorff et al. use a combination of title and abstract to represent a paper, I use only the text log document as input to the language model. The final layer's hidden state of the [CLS] token is then used to represent the document. Following the approach from Ostendorff et al., I continue training the model using a self-supervised triplet margin loss [78]:

$$\mathcal{L} = \max\left\{\left\|\mathbf{d}^Q - \mathbf{d}^+\right\|_2 - \left\|\mathbf{d}^Q - \mathbf{d}^-\right\|_2 + \xi, 0\right\} \tag{3.1}$$

Here, ξ represents the margin which ensures that $\mathbf{d}^+$ is at least ξ closer to $\mathbf{d}^Q$ than $\mathbf{d}^-$, and $\|\Delta d\|_2$ is the L^2 norm, used as a distance function.

Using a contrastive learning objective, similar documents will be placed close in the embedding space of the language model while dissimilar documents will be further away. Since the training is done on domain data, it will capture domain-specific document-level semantics.

3.5 Graph-aware Domain Adaptation

The goal of this work is to improve the domain language representation of a language model by utilizing additional data, i.e., underlying graph structures of text data. Ostendorff et al. [18] enhance scientific document representations by incorporating inter-document information through citations. My research question (see Section 1.2) investigates how this graph-aware contrastive learning approach can be adapted to the process industry domain. Given the distinct nature of the scientific and process industry domains, successfully bridging these differences will expand the methodology's applicability across various fields, establishing it as a domain adaptation technique.

The previous sections proposed adjustments to the methodology to solve challenges of the process industry domain. The key contribution revolves around creating a heterogeneous graph for the process industry and supporting the training of graph embeddings through text embeddings. Table 3.1 gives an overview of the key distinctions of the data used in the work from Ostendorff et al. and this work. There are distinctions in the length and quality of the documents, in the document connectivity, and in the dataset size.

	Ostendorff et al.	This work
Domain	Scientific	Process Industry
Document type	Papers (title + abstract)	Text logs
Document length (in words)	150-250	8-21
Doc-to-Doc connectivity	High	Low
Data quality	High	Low
Dataset size (in docs)	>52M	23-166K
Graph type	Unipartite	Heterogeneous
Graph size (in edges)	>462M	22-152K

Table 3.1: Scientific vs process industry domain.

Document Type Ostendorff et al. use scientific publications, represented by the title and abstract as a document. The text documents of interest for the process industry are reports about daily operations at an industrial plant (aka "text logs").

Document Length An abstract consists of 150-250 words while the median word count of a text log in the available datasets lies between 8 and 21. While abstracts will always keep a certain length to fulfill their roles as a summary of a scientific paper, there is no such restraint for text logs. This results in text logs below 8 words which are challenging for any language learning objective.

Doc-to-Doc Connectivity Scientific publications typically include a large number of references to acknowledge prior research, provide evidence for claims, and situate the work within the broader scientific context. In contrast, linking text logs into a chain of story-related entries is not a commonly-used functionality.

Data Quality Published papers are carefully reviewed, so they rarely contain typos or grammatical errors. On the other hand, text logs are handwritten notes and often include errors, missing words, inconsistent formatting, and other irregularities.

Dataset Size There are millions of papers available, but only a limited number of records from the process industry.

Graph type Ostendorff et al. used a unipartite graph consisting only of paper nodes and their citations. I employ a heterogeneous graph, consisting of text logs and funclocs, and three edge types.

The proposed methodology expands the applicability of SciNCL to domains with less data, shorter texts, and heterogeneous graph structures.

Chapter 4

Implementation

In this chapter, I describe the implementation of the methodology outlined in the previous chapter. I provide an overview of the training data and the training parameters for the graph embedding models and the language models.

In Sections 4.1 and 4.2, I present an overview of the four datasets utilized in this study and outline the various pre-processing techniques applied. In Section 4.3, I describe the constructed process industry graphs. In Section 4.4, I explain the training process for the graph embedding models. In Section 4.5, I present the sampled triplets. Lastly, in Section 4.6, I describe the language model chosen for the later experiments and its training parameters.

Ostendorff et al. [18] made their code publicly available[1]. This allowed me to modify the code base to meet the specific requirements of my project. SciNCL was implemented using Huggingface Transformers [79], and the KNN strategy was implemented with FAISS [77]. My project code can be found in the submitted ZIP file.

4.1 Data

In this section, I introduce the datasets used in this work by first providing a high-level overview, followed by examples of the text data. Finally, I present the features of the datasets that were utilized for constructing the process industry graphs.

The experiments were performed on four German customer datasets representing different sub-domains of the process industry (chemistry and pharma). The customers, from here on called "tenants", are anonymized by replacing their names with a letter (A, B, C, and D). Each dataset consists of two files: one containing unstructured reports (text logs) documenting daily operations within an industrial plant, and the other describing the plant's functional location tree. Table 4.1 shows an overview of the datasets. Tenant D has the lowest number of text logs and the smallest

[1] `https://github.com/malteos/scincl`

functional location tree, while tenant B shows the lowest median word count, i.e. half of the records consist of eight or fewer words.

Tenant	A	B	C	D
Domain	Chemistry	Chemistry	Chemistry	Pharma
Funclocs	36 824	35 113	62 290	489
Text logs	82 013	112 447	166 866	23 448
- Median word count	21	8	15	18

Table 4.1: Overview of process industry datasets. The term "funclocs" refers to all unique entries of the functional location tree of the given tenant.

To get a better idea of how similar the four datasets are, I calculated the overlap of their respective vocabulary by considering the top 10K most frequent words excluding stopwords (see Figure 4.1). Figure 4.1 shows a slight overlap between tenants A, B, and C, whereas tenant D appears more distinct from the others. This aligns with the distinct domains of the tenants, as tenant D operates in the pharmaceutical industry while the others are part of the chemical sector (see Table 4.1). These findings highlight the need for careful integration of tenant D's data when combining all four datasets to avoid disrupting the language model's learning process.

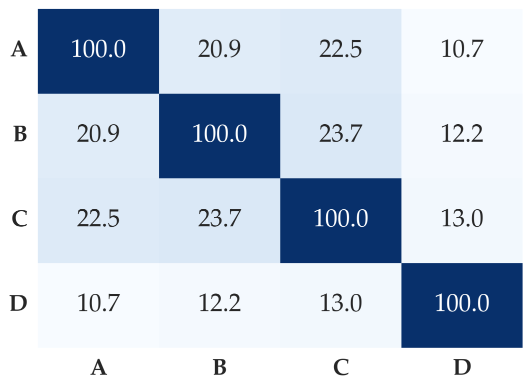


Figure 4.1: Vocabulary overlap (%) between tenants. Vocabularies for each dataset were created by considering the top 10K most frequent words (excluding stopwords).

After giving a more high-level introduction to the different datasets, Table 4.2 shows some anonymized examples of the actual data translated to English. The samples were selected, so that they show the heterogeneity of the data. While some logs cover general information like employee attendance, most focus on maintenance-related activities like routine maintenance tasks, problem identification and resolution, or shift instructions. The different content types are not strictly separated and are often mixed within the data records. The writing style is characterized by short sentences and funcloc codes are often used without any additional description of the machinery behind it. These text characteristics align with the ones described in the maintenance literature [50].

Type	Text
General	Person A and B are on vacation. Person C is sick
Daily Operation	Cleaning carried out, A13 is now pH neutral and empty. The following pipes were flushed, to B345, to K45, and to B455.
Problem/Solution	Due to a shortage in substance A, we reduced the feed into K10 to 500kg/h. The feed was also adjusted for K30.
Instruction	If reactor B is empty, the B12 must be removed and cleaned.

Table 4.2: Text log examples. The examples were translated from German to English and anonymized.

In the following, I introduce the relevant attributes of the datasets. The text log datasets contain the following features:

- Guid: The unique identifier of a text log
- ParentItemGuid: Refers to the id of the text log that occurred earlier but is connected to the event of the current text log
- text: The report about daily maintenance activities
- FunclocGuid: The functional locations that are referred to within the report

The features *FunclocGuid* and *ParentItemGuid* are important for the graph-building step which is described in detail in Section 4.3.

The funcloc datasets contain the following features:

- Guid: The unique identifier of a funcloc
- ParentFunctionalLocationGuid: Refers to the id of a funcloc it is part of
- FuncLoc: A short description of the funcloc

4.2 Data Pre-processing

The pre-processing of the data involved three steps: data enrichment, data cleaning, and data filtering.

Data Enrichment The process industry data was enriched in two ways. First, by extracting funclocs codes mentioned within the text (funcloc retrieval). This method may identify other funclocs besides the already attributed ones. By increasing the number of attributed funclocs, a more interconnected graph can be built. Second, the expansion of the context of a funcloc code within the text by adding the funcloc description, e.g. instead of "A1" it is "Filter A1" (context expansion). This improves the semantic meaning of a text log, while also increasing its length. The functionality for both approaches was provided by the collaborating company. Unfortunately, it was not available for tenant D. Table 4.3 shows a higher number of text logs with attributed funclocs and a higher median word count after data enrichment.

Tenant	A	B	C	D
Text logs	82 013	112 447	166 866	23 448
- with funclocs (before)	77 016	90 202	77 686	21 689
- with funclocs (after)	79 801	96 465	110 046	-
- median word count (before)	21	8	15	18
- median word count (after)	23	10	17	-

Table 4.3: Data enrichment. The number of attributed funclocs before and after funcloc extraction and linking. Median word count before and after context expansion using funcloc descriptions.

Data Cleaning Unlike earlier NLP models, transformer-based models like BERT require minimal pre-processing. Removing stop words and punctuation can reduce performance, as BERT uses these elements for contextual understanding. However, checking the text for any formatting issues or unwanted characters remains important. In my case, this affects newline "\n" and carriage return characters "\r" which I replaced with a period, and tabs "\t" which I replaced with white space. Furthermore, repeated occurrences of white spaces and punctuation were removed.

Data Quality Filtering Ostendorff et al. base their experiments on the Semantic Scholar Open Research Corpus (S2ORC) published by Lo et al. [80]. In the creation process of S2ORC, Lo et al. removed papers with fewer than 100 characters of abstract and body text since they provide little value for text-based analyses. In my work, I exclude log entries with no text, as they do not satisfy the essential criteria for language model training, and no attributed funclocs, which are necessary for constructing the process industry graph. Lo et al. mention a trade-off: papers lacking text may be useful as cite-able nodes in S2ORC but are in general of lower quality. Finally, the authors decided to remove those papers to improve corpus quality. Including other post-processing, the final corpus consists of 81.1M papers. In my case, I only have a small number of records available

(23K-166K, see Table 4.4), removing every short record (<100 chars) would filter out a big part of my records. Thus, I favor the contribution of low-text documents to the graph connectivity, over disregarding them.

Tenant	A	B	C	D
Text logs	82 013	112 447	166 866	23 448
- no text	1	7	53	200
- no funcloc	4996	22 245	89 180	1759
- no parent item	76 940	110 571	154 997	23 100
- < 100 chars of text	50 749	40 467	83 547	13 791
Funclocs	36 824	35 113	62 290	489
- no text	83	123	2839	1

Table 4.4: Data quality filters. Filters displayed in *italic* were not applied.

4.3 Graph

After completing the data pre-processing steps, the process industry graphs were constructed as outlined in Section 3.1. My implementation for creating the graphs can be found in `da_data.py`.

Table 4.5 presents a statistical analysis of the process industry graphs before the data enrichment, providing information on the distribution of the various node and edge types. It shows that the number of direct links between text logs is small (see *follows* edge type). This highlights the importance of indirectly connecting text logs via shared function locations. Since every text log of the graph is associated with at least one funcloc, we observe a high number of *reports_about* edges. The functional location trees are quite big for tenants A, B, and C containing between 35K and 62K funclocs. Since every funcloc is part of another funcloc, we observe a *is_part_of* edge number similar to the number of funclocs nodes, only excluding the root funclocs. The *is_part_of* relation connects more distant text logs over multiple functional locations. Thus, it plays an important role in increasing the connectivity within the graph. All in all, the process industry graphs for tenants A, B, and C contain over 100K nodes and edges, while the one from tenant D is much smaller with around 20K nodes and edges.

Table 4.6 presents a statistical analysis of the process industry graphs after the data enrichment. It shows a higher number of text log nodes, as text logs that previously had no attributed funclocs could now be included in the graph. This increase is especially high for tenant C, gaining 30K new text log nodes. Also, the number of edges increased for all three tenants, multiplying the original number by a factor of 1.35 for tenant A, by 1.55 for tenant B, and by 1.98 for tenant C.

Tenant	A	B	C	D
Nodes	113 833	125 315	139 974	21 983
- textlog	77 016	90 202	77 686	21 494
- funcloc	36 817	35 113	62 288	489
Edges	119 104	137 685	152 478	22 148
- follows	1053	95	950	98
- reports_about	81 242	102 480	89 298	21 563
- is_part_of	36 809	35 110	62 230	487

Table 4.5: Analysis of process industry graphs before data enrichment

Tenant	A	B	C
Nodes	116 611	131 578	172 166
- textlog	79 794	96 465	109 876
- funcloc	36 817	35 113	62 290
Edges	161 106	213 359	303 351
- follows	1074	104	1197
- reports_about	123 223	178 145	239 924
- is_part_of	36 809	35 110	62 230

Table 4.6: Analysis of process industry graphs after data enrichment

4.4 Graph Embedding Model

To evaluate the effect of the data enrichment, I train graph embedding models for both scenarios, before and after enrichment. Given the seven process industry graphs presented in the previous section, this sums up to seven graph embedding models. The code for training the models can be found in `da_graph.py`.

Similar to Ostendorff et al. [18], I use PyTorch-BigGraph (PGB) [76] for training the graph embedding models. PGB supports the training on graphs with multiple node and edge types such as the heterogeneous process industry graph introduced in this work. PGB processes an input graph by taking a list of edges, each defined by a source and target node, and an edge type. It generates an embedding for each node, positioning connected nodes close together in the vector space while pushing apart unconnected ones. As a result, entities with similar neighbor distributions will be located near each other.

PGB was developed for big social network graphs like Twitter, which made it well-suited to work with the large citation graph of S2ORC. For graphs with less than 100K nodes, the developers recommend careful fine-tuning. Looking at the process industry graphs, this is only the case for tenant D (see Table 4.5). PGB allows not only training on heterogeneous graphs but is also scalable to an increasing number of text logs in the future, which makes it a good choice for this work.

Model Parameters Table 4.7 gives an overview of relevant parameters of PGB. I chose the same configuration as Ostendorff et al.

Parameter	Type	Default	SciNCL	Description
operator	str	none	none	The transformation to apply to the embedding of one of the sides of the edge (typically the right-hand one) before comparing it with the other one.
dimension	int		768	The dimension of the real space the embedding live in.
max_norm	float	None	1	If set, rescale the embeddings if their norm exceeds this value.
global_emb	bool	True	False	If enabled, add to each embedding a vector that is common to all the entities of a certain type. This vector is learned during training.
comparator	str	cos	dot	How the embeddings of the two sides of an edge (after having already undergone some processing) are compared to each other to produce a score.
loss_fn	str	ranking	ranking	How the scores of positive edges and their corresponding negatives are evaluated.
num_epochs	int	1	20	The number of times the training loop iterates over all the edges.
num_uniform_negs	int	50	0	The number of negatives uniformly sampled from the currently active partition, per positive edge.
margin	float	0.1	0.15	When using ranking loss, this value controls the minimum separation between positive and negative scores, below which a (linear) loss is incured.
lr	float	0.01	0.1	The learning rate for the optimizer.
eval_fraction	float	0.05	0	The fraction of edges withheld from training and used to track evaluation metrics during training.

Table 4.7: PyTorch-BigGraph (PGB) parameters [76]

Train-Test Split PGB requires as input two edge lists, one for training and one for testing. Since I have multiple relation types, I create a stratified random split. Ostendorff et al. used 99% of the data for training and 1% for testing. The authors do not provide the reasons behind this train-test ratio. Their split resulted in 462M edges for the training and 4.6M edges for the test set, including 52M nodes[2]. In the case of Ostendorff et al., a test split of 1% seems reasonable since it still contains over 4 million edges. However, since I work with smaller datasets, I need to reevaluate the train-test split ratio. Table 4.8 shows exemplary for tenant A how the number of nodes and edges changes based on the train-test ratio. The higher the number of edges in the test split, the higher the number of nodes that were never seen during training. The reason is that many nodes only have one edge, i.e. they can only appear in one data split. To ensure that most nodes are seen during training, I set the train-test ratio to 99-1.

Train			**Test**		
Ratio	Nodes	Edges	Ratio	Nodes	Edges
0.99	112 855	117 911	0.01	2089	1193
0.95	108 968	113 147	0.05	8585	5957
0.9	104 035	107 192	0.1	15 597	11 912

Table 4.8: PyTorch-BigGraph (PGB) train-test split for tenant A.

Text Encoder As mentioned in Section 3.2, I initialize the nodes of the graph model with text embeddings. For text logs, I used their text, and for funclocs, I encoded their short descriptions. The selection of the text encoder was guided by the following criteria. First, it had to be a German language model, given the use of German text data. Second, the embedding dimensionality needed to match that of the graph embedding model to enable node initialization with the generated text embeddings. Finally, the model had to be widely adopted and well-established in the field. Using the number of downloads on huggingface as orientation, I chose *bi-encoder_msmarco_bert-base_german*[3] as the encoder model (over 44K downloads). The developers used GBERT [23] as the base model and trained it on a machine-translated MSMARCO dataset for German. They evaluated the model on the germanDPR dataset and achieved state-of-the-art performance for asymmetric search[3]. I use the SentenceTransformer library to load the encoder [81].

Hyperparameter Tuning While keeping the parameters of Ostendorff et al., I experimented with different numbers of epochs $ep = \{10, 20, 30\}$ and with or without text embedding initialization. I chose tenant A as an example. I trained embeddings on 99% of the data and tested the remaining 1% (1193 edges) on the link prediction task. Table 4.9 shows the link prediction performance for tenant A measured in MRR, Hits@1, Hits@10, and AUC (for all metrics, higher is better and the best value is 100). A definition of the metrics can be found in the PGB documentation[4]. The results show

[2]https://github.com/malteos/scincl
[3]https://huggingface.co/PM-AI/bi-encoder_msmarco_bert-base_german
[4]https://torchbiggraph.readthedocs.io/en/latest/evaluation.html

that initializing nodes with text embeddings greatly improves the link prediction performance. For 10 epochs, the MRR score with text embeddings doubled ($MRR = 36.17$) compared to the MRR score without ($MRR = 16.85$). The same goes for the Hits@10 score which increased from 29.72 to 72 points. These evaluation results highlight the importance of initializing node embeddings with text embeddings for effective training of graph embedding models in the process industry. Increasing the number of epochs only led to marginal increases. Thus, the number of epochs will be kept to 20.

TextEmbed	Epochs	MRR	Hits@1	Hits@10	AUC
no	10	16.85	9.39	29.72	61.11
no	20	16.96	9.39	30.47	59.93
no	30	16.21	8.51	30.43	59.93
yes	10	36.17	21.25	72.00	77.70
yes	20	36.48	21.54	72.25	78.75
yes	30	36.58	21.71	72.38	77.87

Table 4.9: Exemplary hyperparameter tuning of PyTorch BigGraph. Link prediction performance of PyTorch BigGraph embeddings trained on the process industry graph of tenant A with and without text embedding initialization and for a different number of epochs.

Model Evaluation Table 4.10 shows the link prediction performance of the graph models trained for the later experiments. The evaluation was done on 1% of their respective edges. Thus, this is not an evaluation of how well they perform on a shared test set but rather gives an idea of the general training success. I find that the graph models trained on the enriched process industry graphs show a better performance on their respective test sets compared to the models without enrichment. These results were expected since a higher number of edges should make it easier for the graph model to learn the relations between nodes.

Tenant	MRR	Hits@1	Hits@10	AUC	Edge Count
A	36.48	21.54	72.25	78.75	1193
B	46.87	30.08	81.89	86.64	1378
C	37.70	22.61	76.34	81.19	1526
D	48.11	30.63	82.43	83.78	222
A+	47.80	33.91	78.21	85.06	1613
B+	51.80	36.28	83.66	89.30	2136
C+	53.41	37.79	84.60	89.22	3035

Table 4.10: Evaluation of graph embedding models. Link prediction performance of PyTorch BigGraph embeddings trained on the process industry graphs before and after data enrichment. Models were trained with text embedding initialization for 20 epochs.

Hardware PGB performs all computations on the CPU. The training was done with six CPUs of the type Intel® Xeon® Processor E5-2690 v4 @2.60GHz. The text encoding with SentenceBERT was done on a NVIDIA Tesla V100 (16GB).

4.5 Triplet Sampling

Given the learned text log node embeddings, I applied the sampling strategies outlined in Section 3.3 to generate triplets for contrastive learning. The code for creating the triplets is provided in `da_triples.py`.

For each query document, I sampled two triplets, consisting of one easy negative and one hard negative. This provides a diverse similarity signal while ensuring broader document coverage, even within smaller data subsets.

To identify text logs with a similar graph context, i.e., those associated with similar text logs and functional locations, I performed an Approximate Nearest Neighbor (ANN) search in the text log embedding space. I used FAISS [77] to create a flat index of text logs keyed by their node embeddings. By L2 normalizing the embeddings beforehand and using the inner product as a metric, I utilized cosine similarity as the distance measure. After creating the FAISS index and populating it with the text log embeddings, I searched the index using text log embedding queries $\mathbf{d}^Q$ to find the k nearest neighbors in the embedding space as defined by their cosine distance.

I applied the same strategies for sampling positives and negatives as Ostendorff et al., KNN sampling for positives and hard negatives, and filtered random sampling for easy negatives. The sample sizes were set $c^+ = 2$ for positives, $c^-_{easy} = 1$ for easy negatives, and $c^-_{hard} = 1$ for hard negatives. The KNN parameter $k^+ = 2$ was used for positives, and $k^-_{hard} = 50$ for the hard negative. With this, the two closest neighbors were chosen as positive documents. Selecting the closest neighbors is motivated by the low number of direct connections between text logs (see *follows* edge type in Table 4.5). The goal is to ensure that text logs are positioned close in the graph embedding space due to their direct links are included and not skipped by using a higher k^+ value. For the same reason k^-_{hard} was set low because the 50th neighbor would already be sufficiently dissimilar. No further hyperparameter tuning was done for the SciNCL parameters.

Initial experiments indicated that including text sequences, that are too short in the training significantly degrades the language model performance. Based on this finding I introduced a *minimum text length* criterion. Rather than applying it during data pre-processing, I implemented it after training the graph embedding model, just before triplet sampling. This ensures that the highest possible number of nodes and edges are preserved for the graph model training. Keeping short text log nodes and their edges leads to a bigger context for the other text log nodes, i.e. making the learned node embeddings more meaningful. This is especially important since the number of edges is quite low compared to a citation graph, making each additional edge valuable.

Technically, I filtered the text log embedding space before performing the ANN search with FAISS. I set the minimum sentence length to 100 chars (15-25 words). This approach ensures the robustness of text embeddings during language model training by capturing sufficient contextual information, thereby enhancing the model learning process. Table 4.11 shows the number of generated triplets per tenant.

Tenant	A	B	C	D
Queries total	77 016	90 202	77 686	21 494
Queries >= 100 chars	48 621	30 710	45 911	13 348
Triplets (2 per query)	97 242	61 420	91 822	26 696
Queries+ total	79 794	96 465	109 876	-
Queries+ >= 100 chars	52 406	40 729	74 060	-
Triplets+ (2 per query)	104 812	81 458	148 120	-

Table 4.11: Analysis of triplets. Triplets sampled from process industry graph before data enrichment (upper part of table) and after (lower part).

In my experiments, I used a subset of 10K triplets, following Ostendorff et al., who found that using only 1% of their data (around 7K triplets) still outperforms the state-of-the-art on the SciDocs benchmark. To ensure each query document is represented with both of its associated triplets, I randomly sampled query documents rather than individual triplets. Since two triplets are generated per query document, selecting 5K query documents yields the desired 10K triplets. The subset was then shuffled to prevent triplets from the same query document from appearing consecutively. Additionally, I evaluated a combination of triplets from different tenants by creating a balanced mix, ensuring each tenant contributed an equal number of triplets. The triplets were shuffled again to maintain a diverse mix during training.

4.6 Language Model

After introducing several German foundation models in Section 2.1, I chose GBERT-base for my experiments [23]. Since the fine-tuned model will be deployed in production, model size is a critical factor, which led me to focus solely on base-size models. GBERT-base was found to be the best base-size model on the SuperGLEBer benchmark, even surpassing its larger counterpart on certain tasks [27]. Additionally, GBERT-base demonstrated strong performance on document-level tasks in technical domains, such as the medical field [42], making it well-suited for further document-level optimization in the process industry.

The weights from GBERT can be loaded from the Huggingface Hub[5]. Chan et al. [23] warn that GBERT's pre-training data may contain gender, racial, and religious biases. While this is a concern in contexts like journalism or job applications, it is less relevant in the process industry, where operations are highly technical and documentation is typically not including personal information. Thus, GBERT can be applied in process industry settings.

GBERT processes texts up to a maximum of 512 tokens. Texts exceeding this length are truncated, with only the first 512 tokens considered while the rest are ignored. Since the average document length in my datasets is rather short (see Table 4.1), this will apply only to a small number of text logs.

The models were trained on an NVIDIA Tesla V100 (16GB) using the triplet loss formulated in Equation (3.1). The margin ξ was set to 1 like it was done by Ostendorff et al. [18]. I used Adam with weight decay [82, 83] as the optimizer and set the learning rate to $\lambda = 2^{-5}$. The models were trained for 2 epochs with a batch size of 6. The pooling strategy is to use the [CLS] token to represent the document. The code to train the model can be found in `da_model.py`.

[5]`https://huggingface.co/deepset/gbert-base`

Chapter 5

Evaluation

In this chapter, I evaluate the proposed methodology on a domain benchmark of the process industry. The benchmark covers a wide range of NLP tasks like text classification, semantic search, and token classification. I conduct an ablation study by analyzing the influence of different datasets, fine-tuning an already domain-adapted PLM, and further enriching the training data. I discuss the results and provide an outlook for future work.

5.1 Process Industry Applications Benchmark

To evaluate whether a language model successfully adapted to the domain-specific language of a given domain, it needs to be evaluated on domain-specific downstream tasks [43]. The Process Industry Applications (PIA) benchmark is a private benchmark consisting of various NLP applications found in the process industry and is used internally by the collaborating company. The benchmark consists of four document-level and four token-level tasks. While the training objective presented in Section 3.4 focuses on document-level improvements, I still include the token-level tasks to assess whether the model learns finer-grained linguistic patterns that could indirectly enhance performance on those tasks. Table 5.1 gives an overview of the PIA benchmark.

Problem-Solution Classification (PSC) A text classification task that classifies sentences into three classes (problem, solution, or info). Sentences that contain both problems and solutions were excluded from the dataset. The info class contains all entries that are not a problem or a solution. Since the class label distribution is unbalanced (train+val: 0.15, 0.08, 0.77), the F1 score is used as the evaluation metric.

Priority Classification (PC) A text classification task that classifies entry items into five classes of criticality (the higher - the more critical the event): 1) nothing, 2) done, 3) low priority, 4) shift supervisor, and 5) high priority. Since the criticality levels can be ordered, Spearman's correlation coefficient is used to estimate how the predicted classes correlate with the assigned levels.

	Name	Task	Dataset size	# Classes	Metric
PSC	Problem-solution classification	text classification, multiclass	10141	3	F1 macro
PC	Priority classification	text classification, multiclass	10000	5	Spearman's rank correlation coefficient
RL	Record linking	next sentence prediction, binary classification	5212	2	Accuracy
SemS	Semantic search	semantic search	Queries: 205, Docs: 14,680	-	nDCG@10*
PSE	Problem-solution extraction	token classification, multiclass	731	3	F1 sequential
SSP	Sentence Splitting	token classification, binary	1003	2	Accuracy
NER	Named-Entity-Recognition	token classification, multiclass	2906	6	F1 sequential
fNER	Funcloc-NER	token classification, binary	8895	2	F1 sequential

Table 5.1: Process Industry Applications (PIA) benchmark
*nDCG@10 = normalized discounted cumulative gain

Record Linking (RL) Record Linking is a task of the domain-specific Natural Language Inference (NLI), i.e., the task of determining whether a "hypothesis" (i.e., second entry item) is true (entailment) or false (contradiction) given a "premise" (i.e., first entry item). In other words, if the second entry item is locally connected to the first one as a continuation of an event.

Semantic Search (SemS) Given a query, retrieve and rank all documents semantically related to it. The annotated dataset contains three degrees of query-document relatedness: (score 3) full, i.e., a document directly contains a problem/situation described in a query, (score 2) adjacent, i.e., a document contains term/synonym(s) from a query but a query describes either a part or a more general concept, (score 1) partial, i.e., a document partially contains the information requested in a query or the query terms are scattered across the document and do not belong to one concept. To measure the ranking quality, the normalized discounted cumulative gain is calculated for each item in the top 10 positions (nDCG@10).

Problem-Solution Extraction (PSE) A token classification task that tags which sequences of tokens within sentences belong to problem or solution descriptions. Unlike the problem-solution classification task, it considers sentences containing both problems and solutions within.

Sentence Splitting (SSP) Given a domain-specific format of the entry items, which often look like bullet points with a lot of abbreviations, shortenings, and codes, the model needs to learn the starts and ends of the sentences. The task is implemented as a binary token classification task, i.e., start and non-start tokens.

Named Entity Recognition (NER) A domain-specific NER task, where tags may include such classes as person, product, chemical, cardinal, name, and short code of functional locations.

Funcloc-NER (fNER) A more specific version of NER that tags only specific types of tokens that refer to the funcloc short codes, e.g., "A12".

Model Training The huggingface library [79] is used to load the model weights into task-specific models (i.e. AutoModelForSequenceClassification and AutoModelForTokenClassification). The pre-trained model weights are unfrozen and fine-tuned together with the weights of the classification layer. The training parameters are fixed for all tasks (learning rate = 1e-5, weight decay = 0.01, batch size = 16, number of epochs = 3), i.e., no hyperparameter optimization is applied. For the semantic search task, the model weights are loaded with the SentenceTransformer library [81] and no further task-specific training is done. For the base models, I retrieve the document representation by applying mean pooling, i.e. the mean of all output token embeddings. For the fine-tuned models, I use the [CLS] token as document representation since it was optimized by the contrastive learning objective (see Section 3.4).

Evaluation Scores The scores are reported as an average over three seeds. For most of the tasks, the seeds are used to create different train-val-test splits (60-20-20) as well as to initialize the classification layer weights. However, since PSC and fNER have a fixed train-test split, the seeds are only used for the later one. For the SemS task, a single score is reported, since only one test set is given. For all metrics higher values indicate better performance.

Base Models There are no publicly available German models adapted to the process industry. Accordingly, I use GBERT$_{\text{base}}$ [23], a German PLM which was pre-trained on a general language corpus, and GBERT$_{\text{domain}}$, a domain-adapted version of GBERT provided by the collaborating company. The domain model was continually pre-trained with the Masked Language Modeling task on 2.8M records (about 1.5GB) of process industry data (86%) and other domain-related resources (14%). The domain-related data includes patents and regulations for the chemical and pharmaceutical industry and machinery.

5.2 Results and Discussion

In this section, I describe and discuss the experiments done to evaluate the proposed graph-based domain adaptation methodology (see Figure 3.1). The first experiment analyzes the influence of fine-tuning a general-domain PLM (GBERT$_{\text{base}}$) with the proposed methodology. The second experiment evaluates how fine-tuning on different datasets impacts the model performance on the benchmark. The third experiment assesses the effect of further fine-tuning an already domain-adapted model (GBERT$_{\text{domain}}$). The fourth experiment evaluates if the data enrichment described in Section 4.2 led to better triplets for contrastive learning and thus, better benchmark scores.

5.2.1 Influence of Contrastive Fine-tuning

Goal This experiment evaluates the general influence of fine-tuning GBERT$_{\text{base}}$ with the proposed methodology.

Setup I sampled 10K triplets as a mix of all four tenants and fine-tuned GBERT$_{\text{base}}$ with them. The sample size was chosen based on Ostendorff et al. [18], who found that using only 1% of their data (around 7K triplets) still outperformed the state-of-the-art on the SciDocs benchmark [17].

Results Table 5.2 presents the performance of the base and the fine-tuned GBERT model across the eight tasks of the PIA benchmark.

The fine-tuned model outperformed the base model on three tasks (SemS, SSP, fNER) and matched the performance in one NER. For the Semantic Search (SemS) task, fine-tuning nearly doubled the score from 9.18 to 17.73 points. The Sentence Splitting (SSP) score increased slightly by 0.7 points. Similarly, the fNER increased by 0.15 points from 94.61 to 94.76 points. For NER, the reported scores only differ by 0.03 points. While fine-tuning improved the scores for the previously mentioned tasks, it decreased the performance for the rest. The score for the Problem-Solution Classification (PSC) task dropped by 1.4 points and for the Priority Classification (PC) task by 1.2 points. For Record Linking (RL) the scores decreased slightly by 0.2 points. For Problem-Solution Extraction (PSE), the score is 0.7 points lower than the one of the base model. On average, the fine-tuned model outperformed the base model by 1 point. However, the differences between the base and the fine-tuned model are often not significant as shown by the standard deviation of the reported scores. Only the SemS score showed a significant performance increase.

	Document Level				Token Level				Average
Task	**PSC**	**PC**	**RL**	**SemS**	**PSE**	**SSP**	**NER**	**fNER**	
Model / Metric	F1	Coef	Acc	nDCG	F1	Acc	F1	F1	
GBERT$_{\text{base}}$	**69.75**$_{0.94}$	**47.80**$_{1.25}$	**95.91**$_{0.52}$	9.18	**45.21**$_{2.38}$	77.93$_{3.03}$	63.22$_{0.40}$	94.61$_{0.06}$	62.95$_{1.09}$
GBERT$_{\text{base-A-B-C-D}}$	68.32$_{0.55}$	46.57$_{1.36}$	95.70$_{0.31}$	**17.73**	44.49$_{2.58}$	**80.63**$_{0.19}$	**63.25**$_{2.33}$	**94.76**$_{0.07}$	**63.93**$_{1.06}$

Table 5.2: Influence of contrastive fine-tuning reported on PIA benchmark. GBERT$_{\text{base}}$ was fine-tuned on a mix of 10K triplets from all four tenants. SemS is reported as a single score, while the other scores are reported as averages over 3 seeds. The best results are highlighted in bold.

Discussion Since the model was fine-tuned with a document-level objective, I expected an improvement in the document-level tasks. However, only the performance for the SemS task increased while it decreased for PSC, PC, and RL. In the following, I discuss the possible impact of the funcloc similarity signal on the different tasks.

During the contrastive fine-tuning, the model learned to produce closer text embeddings for text logs sharing similar funclocs. This objective works well for the semantic search task. For example, given the query "fix broken pump", we would expect documents related to this issue to be retrieved. Related documents do not necessarily need to mention the word "pump" but can also only contain a funcloc code (e.g., "A1") representing a pump. Since the latter misses the semantic meaning, it is less likely to be retrieved. However, if both text logs share the same functional locations, their representations were placed closer during contrastive learning. Thus, the second text log is more likely to be retrieved as well.

For PC, the funcloc similarity signal does not contain any information to help classify text logs in different priority categories. The additional training of the language model could have introduced more noise, leading to a worse performance. For PSC, using funcloc similarity as a a training objective could have made the differentiation between problem and solution documents harder since they most likely share the same funclocs.

The RL task evaluates whether two text logs are a continuation of an event. This continuation was depicted in the process industry graph as a "follows" relationship (see Figure 3.2). Since this relation was considered for learning text log embeddings, I expected an improvement for this task. However, the RL score did not improve. One reason could be that the "follows" relationship is under-presented in the process industry graph (see Table 4.5) which reduces its effect on the triplet sampling. The dominant similarity signal is the shared funclocs. While text logs that are a continuation of an event are likely to share the same funclocs, more text logs share funclocs but are not a continuation of each other. This could have made the classification more difficult in such cases. Finally, considering that GBERT$_{\text{base}}$ already achieved an accuracy of around 96 percent, the contrastive training could have been too noisy, leading to a performance drop of less than one percentage point.

The increased performance for token-level tasks like SSP and fNER was unexpected, considering the document-level training objective. SSP might have profited from the data pre-processing where newline and carriage returns were replaced by a period (see Section 4.2), helping the model to better separate texts of the process industry. The fNER task could have benefited from funcloc similarity. Optimizing the model to differentiate documents based on shared funclocs could have enhanced the embeddings of tokens related to these locations (e.g., funcloc codes), improving the model's ability to identify these entities.

In benchmarks specialized for evaluating document embeddings like SciDocs [17], text classification is usually done by using the document embeddings as a feature for a linear SVM. This allows a clear interpretation of the quality of embeddings produced by different embedding models. Since fine-tuning optimizes all the weights of the PLM for the specific downstream task, there is a chance that the knowledge introduced by the contrastive learning objective is completely overwritten. To better evaluate the quality of document embeddings for the process industry in the future, the PIA benchmark could be extended by adding feature-based implementations for the text classification tasks. The SemS task is the only task that directly evaluates the document embeddings. Thus, it is the most suited one to measure the impact of the contrastive learning applied in this work.

5.2.2 Influence of Different Datasets

Goal This experiment evaluates the influence of fine-tuning GBERT$_{\text{base}}$ on triplets from single and multiple tenants.

Setup In addition to the A-B-C-D subset from the previous experiment, I sampled five more 10K subsets: one per tenant and a combined subset excluding tenant D. Tenant D was excluded from the latter mix as it is the only pharma tenant within the training data, while the benchmark consists of data from five chemistry tenants and only two from pharma. I fine-tuned GBERT$_{\text{base}}$ on each of these triplet subsets, resulting in five new models.

Results Table 5.3 presents the performance of the base and fine-tuned GBERT models across the eight tasks of the PIA benchmark.

Similar to the previous experiment, all fine-tuned models outperformed GBERT$_{\text{base}}$ on the SemS, SSP, and NER tasks. For SemS, the improvements ranged between 5 to 12 points. For SSP, the increase was up to 1.2 points over the baseline score of 77.93 points, and for fNER, it was up to 0.15 points over the baseline of 94.61 points. For NER, we observe an increase of up to 0.66 points over the baseline of 63.22 points in all but one configuration (GBERT$_{\text{base-A-B-C}}$).

For PSC, RL, and PSE, all fine-tuned models performed worse than GBERT$_{\text{base}}$. PSC scores were up to 1.8 points lower than the baseline of 69.75, RL up to 0.6 points lower than 95.91, and PSE up to 5.9 points lower than 45.21. For PC, three fine-tuned models improved over the baseline while the rest had a lower score. Considering the standard deviation, there is no significant increase or decrease in the tasks besides SemS.

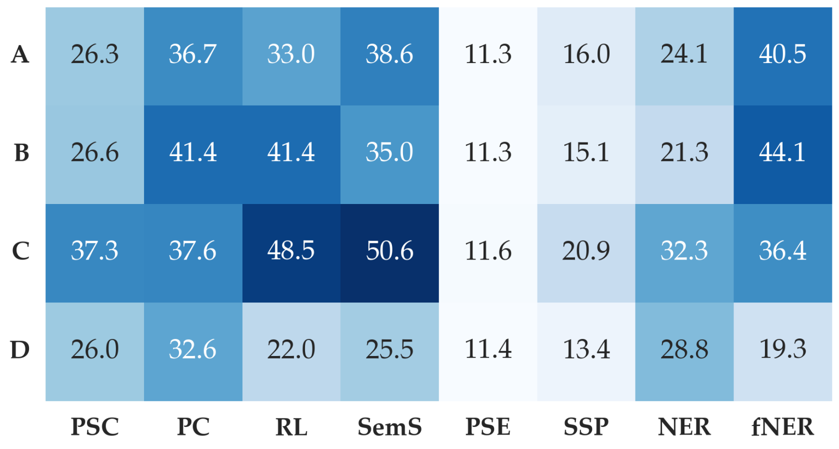
The models fine-tuned on mixed tenant data did not outperform single-tenant models. While the model fine-tuned on A-B-C-D achieved the second-highest benchmark score (63.93), the model fine-tuned only on A-B-C achieved the lowest overall score (63.33) of all fine-tuned models. No fine-tuned model showed superior performance across all benchmark tasks. GBERT$_{\text{base-A}}$ performed the best on semantic search, GBERT$_{\text{base-C}}$ had the highest SSP score, and GBERT$_{\text{base-A-B-C-D}}$ performed the best on the fNER task.

	Document Level				Token Level				Average
Task	**PSC**	**PC**	**RL**	**SemS**	**PSE**	**SSP**	**NER**	**fNER**	
Model / Metric	F1	Coef	Acc	nDCG	F1	Acc	F1	F1	
GBERT$_{\text{base}}$	$\underline{\mathbf{69.75}}_{0.94}$	$47.80_{1.25}$	$\underline{\mathbf{95.91}}_{0.52}$	9.18	$\underline{\mathbf{45.21}}_{2.38}$	$77.93_{3.03}$	$63.22_{0.40}$	$94.61_{0.06}$	$62.95_{1.09}$
GBERT$_{\text{base-A}}$	$68.16_{0.58}$	$47.90_{1.83}$	$95.73_{0.42}$	$\underline{\mathbf{21.12}}$	$41.83_{2.55}$	$80.10_{2.21}$	$\underline{63.84}_{0.62}$	$94.62_{0.15}$	$\underline{\mathbf{64.16}}_{1.19}$
GBERT$_{\text{base-B}}$	$68.28_{0.81}$	$47.63_{1.11}$	$95.76_{0.63}$	17.12	$43.15_{0.45}$	$80.40_{2.69}$	$63.49_{0.17}$	$94.67_{0.05}$	$63.81_{0.84}$
GBERT$_{\text{base-C}}$	$67.94_{0.26}$	$46.93_{1.84}$	$95.28_{0.73}$	14.07	$43.49_{3.32}$	$\underline{\mathbf{81.10}}_{3.03}$	$63.66_{0.45}$	$94.73_{0.07}$	$63.40_{1.39}$
GBERT$_{\text{base-D}}$	$\underline{69.02}_{1.07}$	$\underline{48.19}_{1.03}$	$\underline{95.77}_{0.86}$	14.01	$42.15_{3.14}$	$\underline{80.90}_{2.33}$	$\underline{\mathbf{63.88}}_{0.49}$	$\underline{94.74}_{0.02}$	$63.58_{1.28}$
GBERT$_{\text{base-A-B-C}}$	$68.71_{0.90}$	$\underline{\mathbf{48.87}}_{3.21}$	$95.42_{0.46}$	16.51	$39.36_{2.85}$	$80.07_{3.07}$	$62.99_{0.33}$	$94.71_{0.04}$	$63.33_{1.55}$
GBERT$_{\text{base-A-B-C-D}}$	$68.32_{0.55}$	$46.57_{1.36}$	$95.70_{0.31}$	$\underline{17.73}$	$\underline{44.49}_{2.58}$	$80.63_{0.19}$	$63.25_{2.33}$	$\underline{\mathbf{94.76}}_{0.07}$	$\underline{63.93}_{1.06}$

Table 5.3: Influence of different datasets reported on PIA benchmark. GBERT$_{\text{base}}$ was fine-tuned on 10K triplets from either single tenants or as a combination from multiple tenants. SemS is reported as a single score, while the other scores are reported as averages over 3 seeds. The best results are bold and underlined, second bests are just underlined.

Figure 5.1 shows the vocabulary overlap between the tenant datasets and benchmark datasets. There is no correlation between the vocabulary overlap and the single-tenant model performance. For example, tenant C had the highest vocabulary overlap with the semantic search task while tenant D had the lowest, however, their fine-tuned models got a similar score of around 14 points.

Discussion Although the extent of increase or decrease in performance varied depending on the dataset used, the same tasks consistently improved (SemS, SSP, fNER) or declined (PSC, RL, PSE) across all models. This suggests that the training objective itself is likely the reason for this trend. The only exception is the the Priority Classification (PC) task where three of the six fine-tuned models outperformed the baseline (A, D, and A-B-C). While the contrastive learning objective does not actively encode semantics about the priority of text logs, the improved document representation of the process industry data could have positively influenced the task-specific fine-tuning. However, this effect is inconsistent as the performance degraded for the other three cases.



	PSC	PC	RL	SemS	PSE	SSP	NER	fNER
A	26.3	36.7	33.0	38.6	11.3	16.0	24.1	40.5
B	26.6	41.4	41.4	35.0	11.3	15.1	21.3	44.1
C	37.3	37.6	48.5	50.6	11.6	20.9	32.3	36.4
D	26.0	32.6	22.0	25.5	11.4	13.4	28.8	19.3

Figure 5.1: Vocabulary overlap (%) between tenants and benchmark datasets. Vocabularies for each dataset were created by considering the top 5K most frequent words (excluding stopwords).

The benchmark contains data from multiple tenants. However, the models fine-tuned on multiple tenants (A-B-C and A-B-C-D) did not outperform the single-tenant models. One reason could have been that the number of training samples was too small. For, A-B-C-D each tenant only contributed 2.5K triplets. While the more diverse training signal with data from multiple tenants can improve the generalization capabilities of the language model, it also introduces more noise into the training. Increasing the sample size could help the model to better handle the variations in the training data and become more robust.

The performance differences in single-tenant models could be due to several factors. First, not all benchmark tasks include data from every tenant, so models fine-tuned on tenants consistently represented across tasks may have performed better than those trained on less-represented tenants. Variations in text quality between tenants might have further influenced performance. Finally, using the same hyperparameters for training the graph models, sampling triplets, and fine-tuning the language models across all tenants might have favored a specific tenant.

5.2.3 Influence of Domain Model

Goal This experiment evaluates the effect of fine-tuning an already domain-adapted version of GBERT.

Setup I reduced the number of configurations evaluated for this experiment based on the SemS score from the previous experiment. The SemS score was chosen since it profited the most from the contrastive learning. Accordingly, I removed tenants C and D, which achieved the lowest scores on the SemS task. The domain model was provided by the collaborating company, which continually pre-trained GBERT$_{\text{base}}$ with the Masked Language Modeling task on 2.8M records (about 1.5GB of data) of process industry data and other domain-related resources. I fine-tuned GBERT$_{\text{domain}}$ with the same 10K triplet subsets used in the previous experiment, resulting in four new models.

Results Table 5.4 shows the results of GBERT$_{\text{domain}}$ and its fine-tuned versions on the PIA benchmark. GBERT$_{\text{domain}}$ outperforms GBERT$_{\text{base}}$ in six out of the eight downstream tasks. The average benchmark score is 2 points higher. For the SemS task, the score increased from 9.18 to 16.07 points. Fine-tuning the base model with the contrastive objective matches or outperforms GBERT$_{\text{domain}}$ on the SemS task by up to 5 points. The new best-performing model on SemS is GBERT$_{\text{domain-A-B-C}}$, achieving a score of 25.41, which is an increase of 9.3 points compared to GBERT$_{\text{domain}}$ and 4.3 higher than the previous best score from GBERT$_{\text{base-A}}$. GBERT$_{\text{domain-A-B-C-D}}$ is the only model that performs worse than the domain model on the SemS task. Similar to fine-tuning the base model, the score for SSP and fNER improved in all configurations over the domain model. For SSP this increase is up to 2.1 points and for fNER up to 0.1 points. However, the improvements are insignificant considering the evaluated models' standard deviation.

	Document Level				Token Level				Average
Task	**PSC**	**PC**	**RL**	**SemS**	**PSE**	**SSP**	**NER**	**fNER**	
Model / Metric	F1	Coef	Acc	nDCG	F1	Acc	F1	F1	
GBERT$_{\text{base}}$	$\underline{69.75}_{0.94}$	$47.80_{1.25}$	$\underline{\mathbf{95.91}}_{0.52}$	9.18	$\underline{\mathbf{45.21}}_{2.38}$	$77.93_{3.03}$	$63.22_{0.40}$	$94.61_{0.06}$	$62.95_{1.09}$
GBERT$_{\text{base-A}}$	$68.16_{0.58}$	$47.90_{1.83}$	$95.73_{0.42}$	21.12	$41.83_{2.55}$	$80.10_{2.21}$	$63.84_{0.62}$	$94.62_{0.15}$	$64.16_{1.19}$
GBERT$_{\text{base-B}}$	$68.28_{0.81}$	$47.63_{1.11}$	$95.76_{0.63}$	17.12	$43.15_{0.45}$	$80.40_{2.69}$	$63.49_{0.17}$	$94.67_{0.05}$	$63.81_{0.84}$
GBERT$_{\text{base-A-B-C}}$	$68.71_{0.90}$	$48.87_{3.21}$	$95.42_{0.46}$	16.51	$39.36_{2.85}$	$80.07_{3.07}$	$62.99_{0.33}$	$94.71_{0.04}$	$63.33_{1.55}$
GBERT$_{\text{base-A-B-C-D}}$	$68.32_{0.55}$	$46.57_{1.36}$	$95.70_{0.31}$	17.73	$\underline{44.49}_{2.58}$	$80.63_{0.19}$	$63.25_{2.33}$	$94.76_{0.07}$	$63.93_{1.06}$
GBERT$_{\text{domain}}$	$\underline{\mathbf{70.68}}_{0.70}$	$\underline{48.91}_{1.53}$	$95.77_{0.75}$	16.07	$43.99_{3.10}$	$80.17_{3.15}$	$64.92_{0.36}$	$94.78_{0.03}$	$64.41_{1.37}$
GBERT$_{\text{domain-A}}$	$68.60_{0.68}$	$\underline{\mathbf{52.01}}_{1.40}$	$95.71_{0.44}$	20.20	$40.24_{3.02}$	$80.77_{2.66}$	$64.54_{0.43}$	$94.76_{0.01}$	$64.60_{1.23}$
GBERT$_{\text{domain-B}}$	$69.38_{0.62}$	$48.60_{1.18}$	$95.13_{0.51}$	$\underline{23.79}$	$41.62_{1.75}$	$\underline{\mathbf{82.27}}_{2.45}$	$\underline{65.08}_{0.48}$	$\underline{94.84}_{0.05}$	$\underline{65.09}_{1.01}$
GBERT$_{\text{domain-A-B-C}}$	$69.39_{1.04}$	$48.71_{1.10}$	$95.30_{0.34}$	$\underline{\mathbf{25.41}}$	$43.53_{1.97}$	$80.50_{3.02}$	$\underline{\mathbf{65.11}}_{0.33}$	$94.81_{0.00}$	$\underline{\mathbf{65.34}}_{1.11}$
GBERT$_{\text{domain-A-B-C-D}}$	$68.72_{0.48}$	$47.59_{0.62}$	$\underline{95.87}_{0.33}$	14.34	$40.35_{2.62}$	$\underline{80.93}_{3.07}$	$64.51_{0.37}$	$\underline{\mathbf{94.88}}_{0.03}$	$63.40_{1.07}$

Table 5.4: Influence of domain model reported on PIA benchmark. GBERT$_{\text{domain}}$ was fine-tuned on 10K triplets from either single tenants or as a combination from multiple tenants. SemS is reported as a single score, while the other scores are reported as averages over 3 seeds. The best results are bold and underlined, second bests are just underlined.

Discussion Continual pre-training with MLM leads to a better domain language representation for the process industry, shown by the domain model outperforming the base model. These results match the ones found in the literature (see Section 2.2), emphasizing why continual pre-training with MLM is the prevalent domain adaptation approach. While MLM is not a document-level training objective, the better representation of domain-specific language patterns and words helps the model to identify relevant documents for a given query in the SemS task.

Nevertheless, fine-tuning the base model with a document-level objective outperforms the domain model on the SemS task. This finding suggests that a document-level objective like the triplet loss used in this work, is more suited to improve the document-level domain language representation than continual pre-training with MLM. Especially, since the improvement was achieved with a much smaller training corpus (10K triplets) compared to training with MLM (2.8M records).

Similar to Ostendorff et al. [18] where fine-tuning SciBERT led to better results than fine-tuning the original BERT model, fine-tuning $\text{GBERT}_{\text{domain}}$ leads to the highest SemS score. This shows that the training effect of MLM-based continual pre-training can be further improved by fine-tuning with a document-level objective. However, in one case fine-tuning the domain model led to a worse SemS score which highlights the sensitivity of the training process.

5.2.4 Influence of Data Enrichment

Goal In Section 4.2, I discussed two ways to enrich the process industry data, funcloc retrieval and context expansion. This experiment evaluates if the data enrichment led to better triplets for contrastive learning.

Setup The SemS task was used again to reduce the number of configurations considered for evaluation. In the second experiment, $\text{GBERT}_{\text{domain-B}}$ and $\text{GBERT}_{\text{domain-A-B-C}}$ performed the best on the SemS task. Thus, I only sampled 10K triplets from the enriched data for these two configurations, tenant B and the mix of A-B-C. I fine-tuned $\text{GBERT}_{\text{base}}$ and $\text{GBERT}_{\text{domain}}$, resulting in four new models.

Results Table 5.5 shows the results of the models trained with data enrichment on the PIA benchmark. For the SemS task, fine-tuning $\text{GBERT}_{\text{base}}$ on the enriched triplets led to better SemS scores than the models trained on non-enriched triplets. However, when fine-tuning $\text{GBERT}_{\text{domain}}$ only an equivalent performance is found for tenant B and a drop of 0.7 points for the mix of A-B-C. While the graph embedding models profited from the data enrichment, shown by the higher evaluation performance (see Table 4.10), this effect did not transfer to the evaluation on the benchmark. Nonetheless, all fine-tuned models improved over the baselines $\text{GBERT}_{\text{base}}$ and $\text{GBERT}_{\text{domain}}$ in the SemS task. $\text{GBERT}_{\text{base-B+}}$ even leads to a new high score on the benchmark with 65.36, outperforming the previous best model $\text{GBERT}_{\text{domain-A-B-C}}$ by a small margin.

Task	Document Level				Token Level				Average
	PSC	PC	RL	SemS	PSE	SSP	NER	fNER	
Model / Metric	F1	Coef	Acc	nDCG	F1	Acc	F1	F1	
GBERT$_{\text{base}}$	$\underline{69.75}_{0.94}$	$47.80_{1.25}$	$\underline{\mathbf{95.91}}_{0.52}$	9.18	$\underline{\mathbf{45.21}}_{2.38}$	$77.93_{3.03}$	$63.22_{0.40}$	$94.61_{0.06}$	$62.95_{1.09}$
GBERT$_{\text{domain}}$	$\underline{\mathbf{70.68}}_{0.70}$	$\underline{\mathbf{48.91}}_{1.53}$	$\underline{95.77}_{0.75}$	16.07	$43.99_{3.10}$	$80.17_{3.15}$	$64.92_{0.36}$	$94.78_{0.03}$	$64.41_{1.37}$
GBERT$_{\text{base-B}}$	$68.28_{0.81}$	$47.63_{1.11}$	$95.76_{0.63}$	17.12	$43.15_{0.45}$	$80.40_{2.69}$	$63.49_{0.17}$	$94.67_{0.05}$	$63.81_{0.84}$
GBERT$_{\text{base-A-B-C}}$	$68.71_{0.90}$	$\underline{48.87}_{3.21}$	$95.42_{0.46}$	16.51	$39.36_{2.85}$	$80.07_{3.07}$	$62.99_{0.33}$	$94.71_{0.04}$	$63.33_{1.55}$
GBERT$_{\text{domain-B}}$	$69.38_{0.62}$	$48.60_{1.18}$	$95.13_{0.51}$	23.79	$41.62_{1.75}$	$\underline{\mathbf{82.27}}_{2.45}$	$\underline{65.08}_{0.48}$	$\underline{\mathbf{94.84}}_{0.05}$	$65.09_{1.01}$
GBERT$_{\text{domain-A-B-C}}$	$69.39_{1.04}$	$48.71_{1.10}$	$95.30_{0.34}$	$\underline{\mathbf{25.41}}$	$43.53_{1.97}$	$80.50_{3.02}$	$\underline{\mathbf{65.11}}_{0.33}$	$\underline{94.81}_{0.00}$	$\underline{65.34}_{1.11}$
GBERT$_{\text{base-B+}}$	$69.56_{0.50}$	$47.93_{0.65}$	$95.35_{0.34}$	$\underline{25.16}$	$\underline{45.06}_{1.66}$	$80.47_{2.51}$	$64.70_{0.37}$	$94.69_{0.16}$	$\underline{\mathbf{65.36}}_{0.89}$
GBERT$_{\text{base-A-B-C+}}$	$68.04_{1.35}$	$48.53_{0.61}$	$95.45_{0.46}$	18.66	$44.27_{1.58}$	$80.10_{3.18}$	$63.76_{0.29}$	$94.56_{0.04}$	$64.17_{1.07}$
GBERT$_{\text{domain-B+}}$	$69.72_{0.53}$	$48.53_{1.47}$	$95.17_{0.53}$	23.80	$42.32_{3.71}$	$\underline{81.37}_{3.03}$	$64.92_{0.23}$	$94.74_{0.05}$	$64.61_{1.36}$
GBERT$_{\text{domain-A-B-C+}}$	$69.30_{1.00}$	$48.68_{0.50}$	$95.52_{0.49}$	24.67	$41.71_{0.86}$	$80.70_{2.83}$	$64.45_{0.39}$	$94.75_{0.03}$	$64.58_{0.87}$

Table 5.5: Influence of data enrichment reported on PIA benchmark. GBERT$_{\text{base}}$ and GBERT$_{\text{domain}}$ were fine-tuned on 10K "enriched" triplets from either single tenants or as a combination from multiple tenants, the models are marked with a plus (+). SemS is reported as a single score, while the other scores are reported as averages over 3 seeds. The best results are bold and underlined, second bests are just underlined. Data enrichment led to a better SemS score in three of four cases.

Discussion Data enrichment yields promising results by further improving performance on the SemS task. While the previous experiment showed that combining the domain model with additional fine-tuning amplified the positive results, the strong performance of GBERT$_{\text{base-B+}}$ suggests that fine-tuning the base model with better training samples can achieve comparable scores. This is encouraging, as continual pre-training with MLM requires large datasets, which are often hard to obtain in many domains.

In domains where structural information for documents is not available or not in sufficient quantity, knowledge extraction methods can be applied to retrieve the desired metadata from the text. The effectiveness of this approach was shown for the datasets used in this work, by increasing the number of attributed funclocs by at least 35 percent (see Table 4.6). However, instead of complementing existing metadata, it can also be used to extract new entity types such as chemicals or products. Since text logs mentioning the same chemicals or products are most likely similar, these new entity types can be added to the heterogeneous process industry graph to further deepen the semantic context of text logs within the graph. Future work could explore this more sophisticated similarity signal to enhance triplet sampling.

Maintenance reports are filled with abbreviations and codes that only domain experts can understand, which poses a challenge for adapting language models to this type of data due to the lack of semantic context. To enhance the meaning of funcloc short codes within the text, I inserted the full semantic name before each code, such as "pump A1" instead of just "A1." However, this approach can introduce knowledge noise, potentially altering the original meaning, especially if the semantic name is lengthy. K-BERT [41] suggests a more advanced form of knowledge enrichment by injecting knowledge triplets into a sentence while limiting the impact of such extra knowledge by introducing a vision matrix and soft-positioning. Such advanced knowledge schemes would allow the injection of even more domain knowledge, making it an interesting choice for future experiments.

5.3 Broader Discussion

In this section, I answer the research question and conduct a broader discussion of the results.

Research Question This master thesis addresses the research question of "how to transfer and adapt a graph-based contrastive learning approach (SciNCL [18]) for the domain adaptation of PLMs for the process industry." Using the assumption that two text logs are similar if they share the same funclocs, I constructed a heterogeneous process industry graph representing the relationships between text logs and funclocs. I created graph embeddings to sample similar and dissimilar text logs in a continuous fashion, and trained a language model using a triplet margin loss to learn their (dis)similarity. To evaluate whether the proposed methodology improves the document representations of PLMs for the process industry, I evaluated the models on a domain benchmark. The experiments show that using the funcloc similarity signal substantially improves the document representations of the process industry by increasing the nDCG@10 score of the semantic search task by up to 16 points compared to the baseline GBERT$_{\text{base}}$. Based on these findings, I can conclude that SciNCL was successfully transferred to the process industry domain.

Domain Adaptation Building on the results from Ostendorff et al. [18], SciNCL was now successfully evaluated in two different languages (English and German) and two different domains (science and process industry), showing its applicability to different contexts. Especially in this work, I addressed the challenges of limited available graph data and missing direct links between documents. Solutions involved extracting additional nodes from the unstructured text, using text embeddings to support the graph model training in areas where the graph's structural information is insufficient, and leveraging heterogeneous graph structures to model document similarity. The findings of this thesis should motivate the application of SciNCL or similar graph-based domain adaptation techniques to domains where direct document connections are sparse and large-scale data is difficult to obtain.

Process Industry The proposed methodology improved the performance of a semantic search task in the process industry. Enhancing semantic search is crucial for accessing critical operational insights from unstructured text logs. By improving the efficiency of retrieving relevant maintenance logs, technicians and engineers can make faster decisions, reducing downtime within an industrial plant.

Similarity Signal Encoding document-level similarity enables the integration of additional knowledge into the language model, beyond just plain text. However, incorporating such specific similarity semantics can reduce the model's generalizability across a wider range of downstream tasks. For instance, in Problem-Solution Classification, this information can have a counterproductive effect by making it harder to distinguish between problems and solutions that share a functional location. It is important to recognize these limitations while leveraging the positive effects for other tasks like semantic search.

5.4 Limitation and Future Work

This section discusses the main limitations of this work and suggests possible directions for future improvement.

Privacy and Reproducibility The training and benchmark data are private. Thus, only people with access to these resources can reproduce the experiments. However, I provide a dummy example with the submitted code, which can be used to execute the pipeline from graph creation, over triplet sampling, to model training (see Appendix A).

Hyperparameter Tuning Hyperparameters for graph model training, triplet sampling, and language model training were largely adopted from Ostendorff et al. [18] to reduce the number of variables and narrow the scope of this thesis. Most of the fine-tuned models successfully improve the performance of the base and domain models on the semantic search task. Further optimization of pipeline parameters could lead to even stronger results on the benchmark.

Benchmark Hyperparameter Tuning The PIA benchmark requires further fine-tuning of the models for its text and token classification tasks. The training parameters were fixed (see Section 5.1), since a hyperparameter tuning for seven different tasks for each evaluated model configuration would be out of scope for this thesis. However, optimizing the training parameters for the benchmark tasks might lead to better results than the ones reported in this work.

SciDocs vs PIA Benchmark SciNCL was evaluated on SciDocs [17], a benchmark specifically created for evaluating document embeddings. The individual tasks directly use the embeddings as input, e.g., to predict citations based on the L2 distance between document vectors or as a feature for a linear SVM to classify documents by their topic. In PIA, only the semantic search task directly evaluates the document embeddings. The classification tasks are implemented by adding a classification layer to the model and fine-tuning all model weights for the new task. However, this makes it hard to separate the contribution of contrastive learning from task-specific fine-tuning. To better evaluate the quality of document embeddings for the process industry in the future, the PIA benchmark could be extended by adding feature-based implementations for the text classification tasks.

Extended Knowledge Graph I constructed a simple knowledge graph representing the relationships between text logs and functional locations (funclocs). The edges were created using funclocs explicitly attributed to the text logs, supplemented by those extracted from the text. Expanding knowledge extraction to include other entities, such as chemicals, could enrich the graph with new types of relations and entities. By embedding text logs within a more complex network of relationships, the node embeddings generated by a graph embedding model would capture richer semantic information, enhancing the identification of similar documents and improving triplet sampling for contrastive learning.

Further Knowledge-enrichment of Text Data In this thesis, I applied simple knowledge enrichment by inserting the semantic name of a funcloc before its short code within the text, e.g., "pump A1" instead of just "A1". K-BERT [41] suggests a more advanced form of knowledge enrichment by injecting knowledge triplets into a sentence while limiting the impact of such extra knowledge on the original meaning of the sentence by introducing a vision matrix. Applying such advanced knowledge schemes is likely to have a positive impact on the semantic quality of process industry text, which is characterized by its shortness and abbreviations.

Similarity Threshold for Triplet Sampling For this work, I sampled the two closest nodes in the graph embedding space as positive samples. However, the closest nodes can still be quite distant in the embedding space. Future work could refine the triplet sampling process by applying a cosine similarity threshold. While this would ensure a minimal similarity of positives, it would also decrease the number of potential training samples since not every query document has neighbors within a given similarity threshold. This trade-off needs to be kept in mind.

Scaling of Training Data All reported experiments were conducted using 10K triplets. Although I performed additional experiments with 50K triplets, these results are not included in this thesis due to observed lower SemS scores compared to the 10K versions. This finding contrasts with the results of Ostendorff et al. [18], who reported improved performance on the SciDocs benchmark with larger training sets. The decline in performance in my experiments may be attributed to sub-optimal training. Future work should focus on optimizing training strategies for larger sample sizes.

Combined Learning Objective The domain model was continually pre-trained with the MLM objective, improving the performance for various downstream tasks. Further fine-tuning the domain model with a contrastive loss achieves an even better performance on the semantic search task. However, it also decreases the score for other tasks. Instead of sequentially applying these training schemes, a combination of both loss functions could yield promising results as shown by TwHIN-BERT [65]. Since MLM training usually requires large amounts of data, this might also apply to the combined loss. Future research should experiment with balancing the contribution of each loss and evaluating its effect on the performance and sampling efficiency.

Chapter 6

Conclusion

In this thesis, I proposed a graph-aware domain adaptation method aimed at enhancing the document representations of Pre-trained Language Models (PLMs) for the process industry domain. The core of this method builds upon SciNCL, a neighborhood contrastive learning technique originally designed for the scientific domain, which samples similar and dissimilar documents using citation graph embeddings.

I constructed a process industry graph comprising two types of nodes, functional locations (such as processing units or machines) and text logs (reports about maintenance activities). The underlying assumption for contrastive learning is that text logs sharing the same functional location are similar. A graph embedding model was trained on this industry-specific graph, and embeddings from the text logs were used to sample similar and dissimilar pairs for contrastive learning. A German PLM was then trained on these samples using a triplet margin loss.

To answer the research question of "how to transfer and adapt a graph-based contrastive learning approach (SciNCL) for the domain adaptation of PLMs for the process industry", I evaluated the proposed methodology on a benchmark of the process industry. While this benchmark contains various NLP tasks, only the semantic search task directly evaluates the document embeddings. Thus, it is the most suitable one for assessing the successful transfer of SciNCL to the process industry. The contrastive fine-tuning was applied to four datasets (A, B, C, and D) and their combinations. Two data enrichment (+) approaches were evaluated, funcloc retrieval and context expansion.

All configurations of the proposed methodology outperformed my first baseline GBERT$_{\text{base}}$ on the semantic search task. As the two baselines, I used GBERT$_{\text{base}}$, a general-domain model, and GBERT$_{\text{domain}}$, continually pre-trained with the MLM objective on 2.8M records of the process industry.

The best-performing model, GBERT$_{\text{domain-A-B-C}}$, had a nDCG@10 score of 25.41, 9 points higher than the domain-adapted model it was trained on. This shows that the efforts of continually pre-training with MLM can be combined with the contrastive learning objective to further improve the representations of documents in the process industry. The second best model, GBERT$_{\text{base-B+}}$, had a nDCG@10 score of 25.16, 16 points higher than the general-domain model GBERT$_{\text{base}}$ it was trained on. Compared to GBERT$_{\text{domain-A-B-C}}$, it was trained with "enriched" triplets, showing that improving the triplet quality can lead to similar high scores. Since training with MLM requires large amounts of data that are not available for every domain, further improving the results by improving the triplet quality is a promising research direction.

This work makes the following key contributions:

1. **Method Transfer and Adaptation:** I successfully transferred and adapted SciNCL's contrastive learning methodology from the scientific domain to the process industry domain, addressing the challenge of short texts, limited direct connections between documents, and a limited number of edges per node in this new context.
2. **Heterogeneous Graph Construction:** I introduced a novel heterogeneous graph structure for the process industry, utilizing functional locations and text logs as nodes and creating relationships between them, establishing a functional location similarity signal for contrastive learning.
3. **Evaluation in the Process Industry:** The methodology was applied to the German language and evaluated on a domain-specific benchmark for the process industry, demonstrating improvements in tasks like semantic search.
4. **Exploration of Triplet Quality:** This thesis highlights the potential for enhancing contrastive learning through better triplet sampling, which can be a viable solution when large-scale data for continual pre-training is not available.

In conclusion, the findings of this thesis encourage further exploration of graph-based domain adaptation techniques, especially for domains where direct document connections are sparse and large-scale data is difficult to obtain.

Bibliography

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[2] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," Jul. 2019, arXiv:1907.11692 [cs]. [Online]. Available: http://arxiv.org/abs/1907.11692

[3] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 7871–7880. [Online]. Available: https://aclanthology.org/2020.acl-main.703

[4] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 8342–8360. [Online]. Available: https://aclanthology.org/2020.acl-main.740

[5] X. Guo and H. Yu, "On the Domain Adaptation and Generalization of Pretrained Language Models: A Survey," Nov. 2022, arXiv:2211.03154 [cs]. [Online]. Available: http://arxiv.org/abs/2211.03154

[6] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledgebase," *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, Sep. 2014. [Online]. Available: https://dl.acm.org/doi/10.1145/2629489

[7] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "LEGAL-BERT: The Muppets straight out of Law School," in *Findings of the Association for Computational*

Linguistics: EMNLP 2020. Online: Association for Computational Linguistics, Nov. 2020, pp. 2898–2904. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.261

[8] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, Sep. 2019. [Online]. Available: https://doi.org/10.1093/bioinformatics/btz682

[9] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A Pretrained Language Model for Scientific Text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3615–3620. [Online]. Available: https://aclanthology.org/D19-1371

[10] K. Huang, J. Altosaar, and R. Ranganath, "ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission," *CoRR*, vol. abs/1904.05342, 2019, arXiv: 1904.05342. [Online]. Available: http://arxiv.org/abs/1904.05342

[11] E. Strubell, A. Ganesh, and A. McCallum, "Energy and Policy Considerations for Modern Deep Learning Research," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 09, pp. 13 693–13 696, Apr. 2020, number: 09. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7123

[12] B. Plank, "What to do about non-standard (or non-canonical) language in NLP," in *Proceedings of the 13th Conference on Natural Language Processing*, ser. Bochumer Linguistische Arbeitsberichte, S. Dipper, F. Neubarth, and H. Zinsmeister, Eds., vol. 16, Bochum, Germany, 2016. [Online]. Available: https://www.linguistics.rub.de/konvens16/pub/2_konvensproc.pdf

[13] M. C. May, J. Neidhöfer, T. Körner, L. Schäfer, and G. Lanza, "Applying Natural Language Processing in Manufacturing," *Procedia CIRP*, vol. 115, pp. 184–189, Jan. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827122015062

[14] S. M. R. Naqvi, M. Ghufran, C. Varnier, J.-M. Nicod, K. Javed, and N. Zerhouni, "Unlocking maintenance insights in industrial text through semantic search," *Computers in Industry*, vol. 157-158, p. 104083, May 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361524000113

[15] J. P. Usuga-Cadavid, S. Lamouri, B. Grabot, and A. Fortin, "Using deep learning to value free-form text data for predictive maintenance," *International Journal of Production Research*, vol. 60, no. 14, pp. 4548–4575, Jul. 2022. [Online]. Available: https://doi.org/10.1080/00207543.2021.1951868

[16] R. Kinney, C. Anastasiades, R. Authur, I. Beltagy, J. Bragg, A. Buraczynski, I. Cachola, S. Candra, Y. Chandrasekhar, A. Cohan, M. Crawford, D. Downey, J. Dunkelberger, O. Etzioni, R. Evans, S. Feldman, J. Gorney, D. Graham, F. Hu, R. Huff, D. King, S. Kohlmeier, B. Kuehl, M. Langan, D. Lin, H. Liu, K. Lo, J. Lochner, K. MacMillan, T. Murray, C. Newell, S. Rao, S. Rohatgi, P. Sayre, Z. Shen, A. Singh, L. Soldaini, S. Subramanian, A. Tanaka, A. D. Wade, L. Wagner, L. L. Wang, C. Wilhelm, C. Wu, J. Yang, A. Zamarron, M. Van Zuylen, and D. S. Weld, "The Semantic Scholar Open Data Platform," Jan. 2023, arXiv:2301.10140 [cs]. [Online]. Available: http://arxiv.org/abs/2301.10140

[17] A. Cohan, S. Feldman, I. Beltagy, D. Downey, and D. Weld, "SPECTER: Document-level Representation Learning using Citation-informed Transformers," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 2270–2282. [Online]. Available: https://aclanthology.org/2020.acl-main.207

[18] M. Ostendorff, N. Rethmeier, I. Augenstein, B. Gipp, and G. Rehm, "Neighborhood Contrastive Learning for Scientific Document Representations with Citation Embeddings," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 11 670–11 688. [Online]. Available: https://aclanthology.org/2022.emnlp-main.802

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., Dec. 2017, pp. 6000–6010. [Online]. Available: https://dl.acm.org/doi/10.5555/3295222.3295349

[20] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," 2018.

[21] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators," in *8th International Conference on Learning Representations*. Addis Ababa, Ethiopia: arXiv, Mar. 2020, arXiv:2003.10555 [cs]. [Online]. Available: http://arxiv.org/abs/2003.10555

[22] P. He, X. Liu, J. Gao, and W. Chen, "DeBERTa: Decoding-enhanced BERT with Disentangled Attention," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=XPZIaotutsD

[23] B. Chan, S. Schweter, and T. Möller, "German's Next Language Model," in *Proceedings of the 28th International Conference on Computational Linguistics*, D. Scott, N. Bel, and C. Zong, Eds.

Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 6788–6796. [Online]. Available: https://aclanthology.org/2020.coling-main.598

[24] R. Scheible, F. Thomczyk, P. Tippmann, V. Jaravine, and M. Boeker, "GottBERT: a pure German Language Model," *ArXiv*, Dec. 2020. [Online]. Available: https://arxiv.org/abs/2012.02110

[25] B. Minixhofer, F. Paischer, and N. Rekabsaz, "WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, Eds. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 3992–4006. [Online]. Available: https://aclanthology.org/2022.naacl-main.293

[26] A. Dada, A. Chen, C. Peng, K. Smith, A. Idrissi-Yaghir, C. Seibold, J. Li, L. Heiliger, C. Friedrich, D. Truhn, J. Egger, J. Bian, J. Kleesiek, and Y. Wu, "On the Impact of Cross-Domain Data on German Language Models," in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 13 801–13 813. [Online]. Available: https://aclanthology.org/2023.findings-emnlp.922

[27] J. Pfister and A. Hotho, "SuperGLEBer: German Language Understanding Evaluation Benchmark," in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, K. Duh, H. Gomez, and S. Bethard, Eds. Mexico City, Mexico: Association for Computational Linguistics, Jun. 2024, pp. 7904–7923. [Online]. Available: https://aclanthology.org/2024.naacl-long.438

[28] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised Cross-lingual Representation Learning at Scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 8440–8451. [Online]. Available: https://aclanthology.org/2020.acl-main.747

[29] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer, "Multilingual Denoising Pre-training for Neural Machine Translation," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 726–742, Dec. 2020. [Online]. Available: https://direct.mit.edu/tacl/article/96484

[30] M. Ostendorff, T. Blume, and S. Ostendorff, "Towards an Open Platform for Legal Information," in *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, ser. JCDL '20. New York, NY, USA: Association for Computing Machinery, Aug. 2020, pp. 385–388. [Online]. Available: https://dl.acm.org/doi/10.1145/3383583.3398616

[31] P. J. Ortiz Suárez, L. Romary, and B. Sagot, "A Monolingual Approach to Contextualized Word Embeddings for Mid-Resource Languages," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 1703–1714. [Online]. Available: https://aclanthology.org/2020.acl-main.156

[32] J. Tiedemann, "Parallel Data, Tools and Interfaces in OPUS," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, N. Calzolari, K. Choukri, T. Declerck, M. U. Doğan, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, Eds. Istanbul, Turkey: European Language Resources Association (ELRA), May 2012, pp. 2214–2218. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2012/pdf/463_Paper.pdf

[33] K. S. Kalyan, A. Rajasekharan, and S. Sangeetha, "AMMU: A survey of transformer-based biomedical pretrained language models," *Journal of Biomedical Informatics*, vol. 126, p. 103982, Feb. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1532046421003117

[34] K. Wang, N. Reimers, and I. Gurevych, "TSDAE: Using Transformer-based Sequential Denoising Auto-Encoderfor Unsupervised Sentence Embedding Learning," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 671–688. [Online]. Available: https://aclanthology.org/2021.findings-emnlp.59

[35] L. Xu, H. Xie, Z. Li, F. L. Wang, W. Wang, and Q. Li, "Contrastive Learning Models for Sentence Representations," *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 4, pp. 1–34, Aug. 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3593590

[36] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple Contrastive Learning of Sentence Embeddings," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6894–6910. [Online]. Available: https://aclanthology.org/2021.emnlp-main.552

[37] K. K. Bressem, J.-M. Papaioannou, P. Grundmann, F. Borchert, L. C. Adams, L. Liu, F. Busch, L. Xu, J. P. Loyen, S. M. Niehues, M. Augustin, L. Grosser, M. R. Makowski, H. J. W. L. Aerts, and A. Löser, "medBERT.de: A comprehensive German BERT model for the medical domain," *Expert Systems with Applications*, vol. 237, p. 121598, Mar. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417423021000

[38] R. Zhang, R. Gangi Reddy, M. A. Sultan, V. Castelli, A. Ferritto, R. Florian, E. Sarioglu Kayi, S. Roukos, A. Sil, and T. Ward, "Multi-Stage Pre-training for Low-Resource Domain

Adaptation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Online: Association for Computational Linguistics, Nov. 2020, pp. 5461–5468. [Online]. Available: https://aclanthology.org/2020.emnlp-main.440

[39] A.-S. Gnehm, E. Bühlmann, and S. Clematide, "Evaluation of Transfer Learning and Domain Adaptation for Analyzing German-Speaking Job Advertisements," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, N. Calzolari, F. Béchet, P. Blache, K. Choukri, C. Cieri, T. Declerck, S. Goggi, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, J. Odijk, and S. Piperidis, Eds. Marseille, France: European Language Resources Association, Jun. 2022, pp. 3892–3901. [Online]. Available: https://aclanthology.org/2022.lrec-1.414

[40] J. Hong, T. Kim, H. Lim, and J. Choo, "AVocaDo: Strategy for Adapting Vocabulary to Downstream Domain," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 4692–4700. [Online]. Available: https://aclanthology.org/2021.emnlp-main.385

[41] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, "K-BERT: Enabling Language Representation with Knowledge Graph," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, pp. 2901–2908, Apr. 2020, number: 03. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/5681

[42] M. Lentzen, S. Madan, V. Lage-Rupprecht, L. Kühnel, J. Fluck, M. Jacobs, M. Mittermaier, M. Witzenrath, P. Brunecker, M. Hofmann-Apitius, J. Weber, and H. Fröhlich, "Critical assessment of transformer-based AI models for German clinical notes," *JAMIA Open*, vol. 5, no. 4, p. ooac087, Nov. 2022. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9663939/

[43] A. Idrissi-Yaghir, A. Dada, H. Schäfer, K. Arzideh, G. Baldini, J. Trienes, M. Hasin, J. Bewersdorff, C. S. Schmidt, M. Bauer, K. E. Smith, J. Bian, Y. Wu, J. Schlötterer, T. Zesch, P. A. Horn, C. Seifert, F. Nensa, J. Kleesiek, and C. M. Friedrich, "Comprehensive Study on German Language Models for Clinical and Biomedical Text Understanding," in *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, N. Calzolari, M.-Y. Kan, V. Hoste, A. Lenci, S. Sakti, and N. Xue, Eds. Torino, Italia: ELRA and ICCL, May 2024, pp. 3654–3665. [Online]. Available: https://aclanthology.org/2024.lrec-main.324

[44] N. Kozaeva, S. Hamotskyi, and C. Hanig, "Development and Evaluation of a German Language Model for the Financial Domain," in *Proceedings of the Joint Workshop of the 7th Financial Technology and Natural Language Processing, the 5th Knowledge Discovery from Unstructured Data in Financial Services, and the 4th Workshop on Economics and Natural Language Processing @ LREC-COLING 2024*, C.-C. Chen, X. Liu, U. Hahn, A. Nourbakhsh, Z. Ma,

C. Smiley, V. Hoste, S. R. Das, M. Li, M. Ghassemi, H.-H. Huang, H. Takamura, and H.-H. Chen, Eds. Torino, Italia: ELRA and ICCL, May 2024, pp. 40–49. [Online]. Available: https://aclanthology.org/2024.finnlp-1.5

[45] S. Hamotskyi, N. Kozaeva, and C. Hänig, "FinCorpus-DE10k: A Corpus for the German Financial Domain," in *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, N. Calzolari, M.-Y. Kan, V. Hoste, A. Lenci, S. Sakti, and N. Xue, Eds. Torino, Italia: ELRA and ICCL, May 2024, pp. 7277–7285. [Online]. Available: https://aclanthology.org/2024.lrec-main.639

[46] C. Klamm, I. Rehbein, and S. P. Ponzetto, "FrameASt: A Framework for Second-level Agenda Setting in Parliamentary Debates through the Lense of Comparative Agenda Topics," in *Proceedings of the Workshop ParlaCLARIN III within the 13th Language Resources and Evaluation Conference*, D. Fišer, M. Eskevich, J. Lenardič, and F. de Jong, Eds. Marseille, France: European Language Resources Association, Jun. 2022, pp. 92–100. [Online]. Available: https://aclanthology.org/2022.parlaclarin-1.13

[47] T. Walter, C. Kirschner, S. Eger, G. Glavaš, A. Lauscher, and S. P. Ponzetto, "Diachronic Analysis of German Parliamentary Proceedings: Ideological Shifts through the Lens of Political Biases," in *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, Sep. 2021, pp. 51–60. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9651887

[48] S. M. R. Naqvi, C. Varnier, J.-M. Nicod, N. Zerhouni, and M. Ghufran, "Leveraging Free-Form Text in Maintenance Logs Through BERT Transfer Learning," in *Progresses in Artificial Intelligence & Robotics: Algorithms & Applications*, L. Troiano, A. Vaccaro, N. Kesswani, I. Díaz Rodriguez, and I. Brigui, Eds. Cham: Springer International Publishing, 2022, pp. 63–75.

[49] F. Akhbardeh, T. Desell, and M. Zampieri, "MaintNet: A Collaborative Open-Source Library for Predictive Maintenance Language Resources," in *Proceedings of the 28th International Conference on Computational Linguistics: System Demonstrations*, M. Ptaszynski and B. Ziolko, Eds. Barcelona, Spain (Online): International Committee on Computational Linguistics (ICCL), Dec. 2020, pp. 7–11. [Online]. Available: https://aclanthology.org/2020.coling-demos.2

[50] A. Dima, S. Lukens, M. Hodkiewicz, T. Sexton, and M. P. Brundage, "Adapting natural language processing for technical text," *Applied AI Letters*, vol. 2, no. 3, p. e33, 2021, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ail2.33. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ail2.33

[51] M. P. Brundage, T. Sexton, M. Hodkiewicz, A. Dima, and S. Lukens, "Technical language processing: Unlocking maintenance knowledge," *Manufacturing Letters*, vol. 27, pp.

42–46, Jan. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2213846320301668

[52] K. Zhong, T. Jackson, A. West, and G. Cosma, "Natural Language Processing Approaches in Industrial Maintenance: A Systematic Literature Review," *Procedia Computer Science*, vol. 232, pp. 2082–2097, Jan. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050924002060

[53] J. P. Usuga Cadavid, B. Grabot, S. Lamouri, R. Pellerin, and A. Fortin, "Valuing free-form text data from maintenance logs through transfer learning with CamemBERT," *Enterprise Information Systems*, vol. 16, no. 6, p. 1790043, 2020, publisher: Taylor & Francis _eprint: https://doi.org/10.1080/17517575.2020.1790043. [Online]. Available: https://doi.org/10.1080/17517575.2020.1790043

[54] L. Martin, B. Muller, P. J. Ortiz Suárez, Y. Dupont, L. Romary, E. de la Clergerie, D. Seddah, and B. Sagot, "CamemBERT: a Tasty French Language Model," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 7203–7219. [Online]. Available: https://aclanthology.org/2020.acl-main.645

[55] H. Le, L. Vial, J. Frej, V. Segonne, M. Coavoux, B. Lecouteux, A. Allauzen, B. Crabbé, L. Besacier, and D. Schwab, "FlauBERT: Unsupervised Language Model Pre-training for French," in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, N. Calzolari, F. Béchet, P. Blache, K. Choukri, C. Cieri, T. Declerck, S. Goggi, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, and S. Piperidis, Eds. Marseille, France: European Language Resources Association, May 2020, pp. 2479–2490. [Online]. Available: https://aclanthology.org/2020.lrec-1.302

[56] F. Carlsson, A. C. Gyllensten, E. Gogoulou, E. Y. Hellqvist, and M. Sahlgren, "Semantic Re-tuning with Contrastive Tension," Oct. 2020. [Online]. Available: https://openreview.net/forum?id=Ov_sMNau-PF

[57] Y. Xiao, S. Zheng, J. Shi, X. Du, and J. Hong, "Knowledge graph-based manufacturing process planning: A state-of-the-art review," *Journal of Manufacturing Systems*, vol. 70, pp. 417–435, Oct. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612523001577

[58] M. Stewart, M. Hodkiewicz, W. Liu, and T. French, "MWO2KG and Echidna: Constructing and exploring knowledge graphs from maintenance data," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, p. 1748006X221131128, Nov. 2022, publisher: SAGE Publications. [Online]. Available: https://doi.org/10.1177/1748006X221131128

[59] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, "Pre-trained models for natural language processing: A survey," *Science China Technological Sciences*, vol. 63, no. 10, pp. 1872–1897, Oct. 2020. [Online]. Available: https://doi.org/10.1007/s11431-020-1647-3

[60] X. Wang, T. Gao, Z. Zhu, Z. Zhang, Z. Liu, J. Li, and J. Tang, "KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 176–194, 2021, place: Cambridge, MA Publisher: MIT Press. [Online]. Available: https://aclanthology.org/2021.tacl-1.11

[61] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: Enhanced Language Representation with Informative Entities," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, A. Korhonen, D. Traum, and L. Màrquez, Eds. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 1441–1451. [Online]. Available: https://aclanthology.org/P19-1139

[62] M. E. Peters, M. Neumann, R. Logan, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith, "Knowledge Enhanced Contextual Word Representations," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 43–54. [Online]. Available: https://aclanthology.org/D19-1005

[63] B. Jin, G. Liu, C. Han, M. Jiang, H. Ji, and J. Han, "Large Language Models on Graphs: A Comprehensive Survey," Feb. 2024, arXiv:2312.02783 [cs]. [Online]. Available: http://arxiv.org/abs/2312.02783

[64] M. Yasunaga, J. Leskovec, and P. Liang, "LinkBERT: Pretraining Language Models with Document Links," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, S. Muresan, P. Nakov, and A. Villavicencio, Eds. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 8003–8016. [Online]. Available: https://aclanthology.org/2022.acl-long.551

[65] X. Zhang, Y. Malkov, O. Florez, S. Park, B. McWilliams, J. Han, and A. El-Kishky, "TwHIN-BERT: A Socially-Enriched Pre-trained Language Model for Multilingual Tweet Representations at Twitter," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Long Beach CA USA: ACM, Aug. 2023, pp. 5597–5607. [Online]. Available: https://dl.acm.org/doi/10.1145/3580305.3599921

[66] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *SIGKDD Explor. Newsl.*, vol. 14, no. 2, pp. 20–28, Apr. 2013. [Online]. Available: https://dl.acm.org/doi/10.1145/2481244.2481248

[67] Y. Zhang, Z. Shen, C.-H. Wu, B. Xie, J. Hao, Y.-Y. Wang, K. Wang, and J. Han, "Metadata-Induced Contrastive Learning for Zero-Shot Multi-Label Text Classification,"

in *Proceedings of the ACM Web Conference 2022*, ser. WWW '22. New York, NY, USA: Association for Computing Machinery, Apr. 2022, pp. 3162–3173. [Online]. Available: https://dl.acm.org/doi/10.1145/3485447.3512174

[68] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "PathSim: meta path-based top-K similarity search in heterogeneous information networks," *Proc. VLDB Endow.*, vol. 4, no. 11, pp. 992–1003, Aug. 2011. [Online]. Available: https://dl.acm.org/doi/10.14778/3402707.3402736

[69] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "MetaGraph2Vec: Complex Semantic Path Augmented Heterogeneous Network Embedding," in *Advances in Knowledge Discovery and Data Mining*, D. Phung, V. S. Tseng, G. I. Webb, B. Ho, M. Ganji, and L. Rashidi, Eds. Cham: Springer International Publishing, 2018, pp. 196–208.

[70] M. Bucher, S. Herbin, and F. Jurie, "Hard Negative Mining for Metric Learning Based Zero-Shot Classification," in *ECCV 16 WS TASK-CV: Transferring and Adapting Source Knowledge in Computer Vision*, Amsterdam, Netherlands, Oct. 2016. [Online]. Available: https://hal.science/hal-01356757

[71] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, "Sampling Matters in Deep Embedding Learning," in *2017 IEEE International Conference on Computer Vision (ICCV)*. Venice: IEEE, Oct. 2017, pp. 2859–2867. [Online]. Available: http://ieeexplore.ieee.org/document/8237571/

[72] M. Asada, N. Gunasekaran, M. Miwa, and Y. Sasaki, "Representing a Heterogeneous Pharmaceutical Knowledge-Graph with Textual Information," *Frontiers in Research Metrics and Analytics*, vol. 6, p. 670206, Jul. 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8281808/

[73] Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, and H. Poon, "Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing," *ACM Transactions on Computing for Healthcare*, vol. 3, no. 1, pp. 1–23, Jan. 2022, arXiv:2007.15779 [cs]. [Online]. Available: http://arxiv.org/abs/2007.15779

[74] T. Wang and P. Isola, "Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere," in *Proceedings of the 37th International Conference on Machine Learning*. PMLR, Nov. 2020, pp. 9929–9939, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v119/wang20k.html

[75] N. Saunshi, O. Plevrakis, S. Arora, M. Khodak, and H. Khandeparkar, "A Theoretical Analysis of Contrastive Unsupervised Representation Learning," in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 5628–5637, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v97/saunshi19a.html

[76] A. Lerer, L. Wu, J. Shen, T. Lacroix, L. Wehrstedt, A. Bose, and A. Peysakhovich, "Pytorch-BigGraph: A Large Scale Graph Embedding System," in *Proceedings of the 2nd Conference on Systems and Machine Learning*, vol. 1. Palo Alto, CA, USA: mlsys.org, Apr. 2019, pp. 120–131. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2019/hash/1eb34d662b67a14e3511d0dfd78669be-Abstract.html

[77] J. Johnson, M. Douze, and H. Jégou, "Billion-Scale Similarity Search with GPUs," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, Jul. 2021, conference Name: IEEE Transactions on Big Data. [Online]. Available: https://ieeexplore.ieee.org/document/8733051

[78] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 815–823, iSSN: 1063-6919.

[79] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-Art Natural Language Processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Q. Liu and D. Schlangen, Eds. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[80] K. Lo, L. L. Wang, M. Neumann, R. Kinney, and D. Weld, "S2ORC: The Semantic Scholar Open Research Corpus," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 4969–4983. [Online]. Available: https://aclanthology.org/2020.acl-main.447

[81] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. [Online]. Available: https://aclanthology.org/D19-1410

[82] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 2017, arXiv:1412.6980 [cs]. [Online]. Available: http://arxiv.org/abs/1412.6980

[83] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," Jan. 2019, arXiv:1711.05101 [cs, math]. [Online]. Available: http://arxiv.org/abs/1711.05101

Appendix A

Code and Data

The project code and dummy data were submitted as a ZIP file.

Dummy Data The experiments were done on private industry data which can not be shared. However, dummy data is provided to give an idea on how to run the code. I created two synthetic German datasets with ChatGPT[1] describing maintenance operations at two different chemical plants with 100 entries each (see `data/da_example`). The data does not cover the full complexity of real process industry data, there might be logical errors and other deviations.

README The README includes a detailed walkthrough for running the project with the dummy data.

[1]`https://openai.com/index/chatgpt/`

Appendix B

AI Usage Statement

In this thesis, I have used ChatGPT or another AI as follows:

- during brainstorming
- for paraphrasing related work
- for proofreading and optimizing
- for the development of software source texts
- for optimizing and restructuring software source texts
- for creating dummy data

I hereby declare that I have stated all uses completely.