Degree Project in Computer Science

First cycle, 30 credits

Advancing Entity Resolution: Creating a Unified Pipeline for Scalable Blocking and Accurate Matching

A Study of Supervised and Unsupervised Methods, Graph-Based Approaches, and Large Language Models

TRISTAN PERROT

Advancing Entity Resolution: Creating a Unified Pipeline for Scalable Blocking and Accurate Matching

A Study of Supervised and Unsupervised Methods, Graph-Based Approaches, and Large Language Models

TRISTAN PERROT

Degree Programme in Computer Science and Engineering
Date: January 2, 2025

Supervisor: Haibo Li
Examiner: Anders Hedman
School of Electrical Engineering and Computer Science
Host company: Wavestone SA
Swedish title: Detta är den svenska översättningen av titeln
Swedish subtitle: Detta är den svenska översättningen av undertiteln

Abstract

Entity Resolution (ER), the task of identifying and linking records that refer to the same real-world entity, is a cornerstone of data integration and analytics. However, achieving both scalability and accuracy in ER pipelines remains a persistent challenge, as traditional methods struggle with large datasets, noisy data, and computational complexity. This problem is significant due to its critical role in domains such as e-commerce, healthcare, and scientific research, where data consolidation and consistency are vital. Despite the wealth of existing research, the complexity of balancing high performance in both blocking and matching stages has left significant room for improvement, making it a suitable and impactful topic for a master's thesis.

To address this, I conducted a comprehensive study of ER techniques, focusing on both blocking and matching. I explored graph-based approaches for blocking to improve scalability and reduce computational overhead. For the matching phase, I leveraged supervised learning using a cross-encoder architecture based on the Sentence bidirectional encoder representations from transformers (SBERT) framework, achieving state-of-the-art results. Additionally, I investigated unsupervised graph-based methods with SBERT embeddings and the potential of Large language models (LLMs) for zero-shot inference. By integrating these techniques, I developed a robust and scalable pipeline capable of handling diverse datasets and scenarios.

The results demonstrate that supervised matching with cross-encoders significantly outperforms other approaches in accuracy, while graph-based and unsupervised methods provide valuable scalability and adaptability insights. This thesis not only benchmarks the performance of these methods against state-of-the-art approaches but also offers a unified perspective on current techniques. The findings enable researchers and practitioners to design more effective pipelines, bridging the gap between scalability and precision, and paving the way for future innovations in the field.

Keywords

Canvas Learning Management System, Docker containers, Performance tuning Choose the most specific keyword from those used in your domain, see for example: the ACM Computing Classification System (https://www.acm.org/publications/computing-classification-system/how-to-use), the IEEE Taxonomy (https://www.ieee.org/publications/services/the

saurus-thank-you.html), PhySH (Physics Subject Headings) (https://physh.aps.org/), ...or keyword selection tools such as the National Library of Medicine's Medical Subject Headings (MeSH) (https://www.nlm.nih.gov/mesh/authors.html) or Google's Keyword Tool (https://keywordtool.io/)

Formatting the keywords:

- The first letter of a keyword should be set with a capital letter and proper names should be capitalized as usual.
- Spell out acronyms and abbreviations.
- Avoid ”stop words” - as they generally carry little or no information.
- List your keywords separated by commas (”,”).

Since you should have both English and Swedish keywords - you might think of ordering them in corresponding order (*i.e.*, so that the n$^{\text{th}}$ word in each list correspond) - this makes it easier to mechanically find matching keywords.

Sammanfattning

Nyckelord

Canvas Lärplattform, Dockerbehållare, Prestandajustering

Résumé

Résumé en français.

Mots-clés

5-6 mots-clés

Acknowledgments

I would like to thank xxxx for having yyyy. Or in the case of two authors: We would like to thank xxxx for having yyyy.

Paris, France, January 2025
Tristan Perrot

Contents

List of Figures

List of Tables

Listings

List of acronyms and abbreviations

AI	Artificial intelligence
BERT	Bidirectional encoder representations from transformers
EDA	exploratory data analysis
ER	Entity Resolution
GDPR	General Data Protection Regulation
GPT	Generative pre-trained transformer
k-NN	k-nearest neighbors
LLM	Large language model
ML	Machine Learning
NLP	Natural Language Processing
PCA	Principal component analysis
POC	Proof-of-Concept
RNN	Recurrent neural network
SBERT	Sentence bidirectional encoder representations from transformers
SDG	Sustainable Development Goal
UN	United Nations

List of Symbols Used

The following symbols will be later used within the body of the thesis.

σ	The total mass of angels per unit area, see equation (C.1),	page 51
A	The area of the needle point	
a	The number of angels per unit area, see equation (C.1)	
A_{circle}	The area of a circle, see equation (C.3),	page 51
D_{circle}	The diameter of a circle, see equation (C.2),	page 51
m	The mass of one angel	
N	The number of angels per needle point,	page 51
r	The radius of a circle, see equation (C.2),	page 51

Chapter 1

Introduction

This chapter describes the specific problem that this thesis addresses, the context of the problem, the goals of this thesis project, and outlines the structure of the thesis.

In this thesis, we will examine Entity Resolution (ER) principles, workflow, methods and possibilities.

1.1 Background

ER is the task of finding records in a data set that refer to the same entity across different data sources (e.g., data files, books, websites, and databases). ER is necessary when joining different datasets based on entities that may or may not share a common identifier. ER has been addressed for many decades since the formulation of its probabilistic foundations in the late 1960s. Since then, two main families of methods have emerged: rule-based and Machine Learning (ML)-based ER. Currently, the top 3 best-scoring models are Large language model (LLM)-based. The main goal is to compare state-of-the-art approaches with more traditional ones in term of precision, usability and scalability. Several people are facing the need to unify different data sources, because they derive from various systems, because they collected open data or because they obtained data through web-scraping. This problem is encountered in a lot of sector.

1.2 Problem

Data is ubiquitous in today's world and can be collected from a variety of sources using different methods. Often, the same data can be retrieved from multiple systems or sources. In such cases, it becomes necessary to unify the data by merging the two datasets, identifying entities common to both. This process involves ER, where pairs of identical entities are matched to combine datasets. The datasets may share similar features or differ significantly, and they can range from highly textual to highly structured formats. ER, also known as record linkage, de-duplication, or entity matching, is a critical task in data management and information integration. It involves identifying and linking records from two datasets that refer to the same real-world entity, such as a person, organization, product, or event. The task addresses the challenge of integrating information from distinct datasets to ensure consistency and accuracy.

1.2.1 Original problem and definition

The original problem of ER arises when different datasets (or records within the same dataset) contain overlapping but not identical information about the same entities. Variations in the way entities are represented (due to data entry errors, missing information, inconsistent formatting, or use of synonyms) make it challenging to determine whether two records refer to the same entity.

The formal definition of ER is:

Given two datasets $D_1 = \{d_{1,1}, d_{1,2}, ..., d_{1,m}\}$ and $D_2 = \{d_{2,1}, d_{2,2}, ..., d_{2,n}\}$, and a similarity function $S(d_{1,i}, d_{2,j})$ that computes the similarity between any record $d_{1,i}$ from D_1 and $d_{2,j}$ from D_2, find all pairs of records $(d_{1,i}, d_{2,j})$ such that $d_{1,i}$ and $d_{2,j}$ refer to the same real-world entity, subject to a similarity threshold τ.

1.2.2 Scientific and engineering issues

The ER task involve scientific and engineering challenges that could lead to potential failure of the algorithm.

- **Data heterogeneity:** Records may come from different sources, each with its own format and schema.
- **Variability and noise:** Names, addresses, and other attributes may be inconsistent or incomplete.

- **Scalability:** Datasets can be large, making pairwise comparisons computationally expensive (complexity in $O(n^2)$).
- **Ambiguity:** Some records might be very similar but refer to different entities, leading to false matches.

1.3 Purpose

This degree project aims to design and implement a Proof-of-Concept (POC) for an end-to-end ER pipeline that surpasses existing benchmarks in terms of accuracy and efficiency. The focus is on leveraging LLMs and machine learning techniques to match records between two datasets, even when traditional identifiers are unavailable. This research is driven by the goal of creating adaptive and reusable tools for ER that can address real-world challenges of data integration across diverse systems and domains.

This project stands to benefit a range of stakeholders, from data scientists and analysts to organizations and the broader research community. Data scientists and analysts, who often face the daunting task of integrating disparate datasets, will gain access to a reusable, scalable tool that automates and simplifies ER. By reducing the need for manual data matching, this solution saves time and effort, allowing these professionals to focus on extracting insights rather than wrestling with data inconsistencies. Organizations and institutions also stand to gain significantly, as a robust ER pipeline can unify data from diverse systems, ensuring consistency and accuracy. This capability is especially critical in domains like healthcare, finance, and retail, where decisions hinge on the quality of integrated data. Moreover, the research community benefits from new insights into how LLMs can enhance ER, advancing the field and paving the way for further innovations. Ultimately, the ripple effect reaches end-users and society at large, as improved data integration enables better services, whether through enhanced healthcare systems, more efficient public policy, or streamlined e-commerce experiences.

The project raises several important ethical, sustainability, and social considerations. From an ethical perspective, privacy is a key concern since ER often involves processing sensitive personal data. Ensuring compliance with data protection regulations, such as General Data Protection Regulation (GDPR), is critical to avoid potential harm to individuals. Additionally, bias in the underlying datasets or models could lead to unfair outcomes, underscoring the need for fairness and transparency in algorithm design and evaluation. Sustainability is another significant concern, as the computational demands of LLMs can result in a substantial carbon footprint. To mitigate this impact, the project emphasizes optimizing algorithms and leveraging energy-efficient hardware. On a broader social level, while improved data integration offers tangible benefits (such as more accurate healthcare records or effective public policy) it could also exacerbate inequalities. Access to such advanced Artificial intelligence (AI) tools may be limited to organizations with significant computational resources, increasing the digital divide. Addressing these issues holistically ensures that the project not only advances technical benchmarks but also aligns with ethical, sustainable, and socially responsible principles.

1.4 Goals

The overarching goal of this degree project is to develop an end-to-end ER pipeline that surpasses current benchmarks [1] in accuracy and efficiency. This pipeline aims to provide a scalable, reusable, and adaptive solution for integrating and matching records across two datasets, leveraging state-of-the-art machine learning techniques and LLMs. This has been divided into the following five sub-goals:

1. **State-of-the-Art Comparison**

   Perform a detailed comparison of existing ER approaches, focusing on traditional rule-based methods, machine learning techniques, and recent advancements using LLMs.

2. **Pipeline Design and Implementation**

   Design an ER pipeline that incorporates cutting-edge techniques, particularly attention-based models, to handle diverse datasets effectively. Ensure the pipeline is modular, scalable, and adaptable to different types of datasets without requiring significant reconfiguration.

3. **Proof-of-Concept Development**

   Develop a POC to validate the proposed pipeline using benchmark datasets, demonstrating its capability to accurately match entities with minimal errors.

4. **Efficiency and Scalability Optimization**

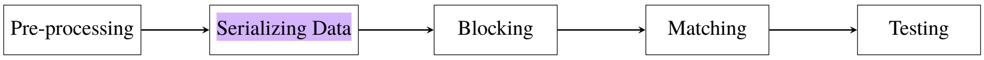


Figure 1.1: ER high level workflow

Optimize the pipeline for computational efficiency, focusing on reducing the time and resources required for both training and inference stages.
Evaluate scalability to ensure the pipeline performs well with both small and large datasets.

5. **Evaluation and Benchmarking**

   Evaluate the pipeline using established metrics such as precision, recall, and F1 score, along with benchmarks for training efficiency and resilience to noisy or unbalanced data.
   Compare results against existing state-of-the-art ER models to establish performance superiority.

1.5 Research Methodology

The ER high level workflow is defined on the Figure 1.1

ER has traditionally been approached through various methodologies, including rule-based methods, Bayesian models, and more recently, deep learning approaches. Each of these methods has distinct strengths and limitations. Rule-based methods rely on manually crafted rules and heuristics, which are interpretable and straightforward but often lack the adaptability required to handle complex datasets or evolving patterns in data. Bayesian models provide a probabilistic framework for ER and are effective for certain structured data problems but can struggle with scalability and the high dimensionality of modern datasets. Deep learning models, on the other hand, have shown exceptional performance in capturing complex relationships in data due to their ability to learn rich representations, particularly with the advent of LLMs.

For this project, I have chosen a hybrid approach combining blocking techniques and LLMs-based deep learning models. Blocking will utilize k-nearest neighbors (k-NN) algorithms with different embeddings to efficiently reduce the candidate pairs for comparison. This step is essential for managing the computational complexity of ER tasks, especially for large datasets. For the training and matching phase, I will build upon LLMs-based models such as Bidirectional encoder representations from transformers (BERT), which currently represent the state-of-the-art in ER. BERT and similar transformer models excel at encoding complex semantic relationships, making them highly suitable for handling unstructured or semi-structured datasets with large textual descriptions.

A key reason for selecting this methodology is its balance between efficiency and accuracy. Blocking methods significantly reduce computational overhead while preserving potential matches, and LLMs-based models maximize the accuracy of final decisions. Additionally, I aim to explore zero-shot learning techniques. These methods leverage pre-trained models to generalize to unseen datasets without additional training, offering practical advantages for scenarios where labeled data is scarce or non-existent.

Other methodologies, such as purely rule-based systems, were not selected due to their limitations in scalability, adaptability, and performance when compared to transformers. While rule-based systems provide interpretability, their rigidity makes them unsuitable for the dynamic and diverse nature of the datasets I plan to address. Bayesian models were also not chosen as a primary approach, given their lower efficiency and difficulty in handling the high variability present in modern ER tasks.

My philosophical assumptions align with the pragmatic research paradigm, focusing on the practical applicability and effectiveness of the developed pipeline. By leveraging modern machine learning techniques and balancing computational efficiency with cutting-edge accuracy, I aim to create a robust, scalable solution that addresses the real-world challenges of ER.

1.6 Delimitations

This thesis project is focused solely on the task of matching entities across two distinct datasets to identify records that refer to the same real-world entity. The scope is deliberately limited to this specific task, as it aligns with the project's goal of developing an efficient, accurate and reusable ER pipeline for the matching of data sets. Other variations of ER, such as deduplication (where the objective is to identify and cluster duplicate records within a single dataset), are explicitly excluded at the beginning of this study. Similarly, any techniques or algorithms related to hierarchical clustering or collective ER are beyond the scope of this project.

The decision to limit the scope in this way is intentional to ensure a focused and in-depth exploration of the chosen problem domain. By concentrating on cross-dataset matching, the project can dedicate resources to optimizing the accuracy and efficiency of the pipeline without being distracted by related but distinct tasks. Including deduplication or other forms of ER would introduce additional complexity, such as different evaluation metrics, variations in data preparation, and model designs, potentially diluting the project's focus and outcomes.

Moreover, the datasets used for this study will be preselected from existing benchmarks, which provide fixed formats and ground truth labels. This means that the project will not explore challenges associated with data set creation, such as raw data extraction, data cleaning, or schema matching. Although these are integral to many real-world ER tasks, they are excluded here to allow the project to prioritize the development of the matching pipeline itself.

Finally, the study will not address broader concerns, such as creating a generalized framework ER for all types of data (e.g. image or geospatial data), since the focus is strictly on textual or tabular data sets. This ensures that the solutions developed are specifically tailored to the datasets and evaluation criteria used in this project. Although these delimitations narrow the scope of the thesis, they are essential to ensure that the project goals are both achievable and impactful within the given time and resource constraints.

1.7 Structure of the thesis

The chapter 2 presents relevant background information about ER. The chapter 3 presents the methodology and method used to solve the problem of ER, the ones existing and the ones I used. The chapter 4 presents the methods that I have created and how they work. The chapter 5 provides the results of my work and a deep analysis of it. Finally, the chapter 7 will conclude the project and provide future work ideas.

Chapter 2

Background

This chapter provides basic background information about ER. Additionally, this chapter describes how ML algorithms work and specifically how BERT work and the Sentence BERT architecture. The chapter also describes related work on ER, blocking and matching existing techniques.

2.1 Machine Learning

ML is a branch of AI that focuses on developing algorithms and models capable of learning patterns and making decisions or predictions based on data. Unlike traditional programming, where explicit instructions are written to solve a problem, ML systems are trained on datasets to identify underlying relationships and generalize this knowledge to new, unseen data. This ability to learn from data makes ML highly versatile, with applications ranging from natural language processing and computer vision to predictive analytics and decision-making systems. ML techniques are broadly categorized into supervised learning, where models are trained on labeled data; unsupervised learning, which finds patterns in unlabeled data; and reinforcement learning, where systems learn by interacting with an environment to achieve a goal. In the context of this project, ML enables the creation of adaptive and efficient models for ER, leveraging data to achieve higher accuracy and scalability compared to traditional rule-based approaches.

2.1.1 Supervised Machine Learning

Supervised machine learning is a method where models are trained on labeled datasets, meaning the input data is paired with corresponding output labels.

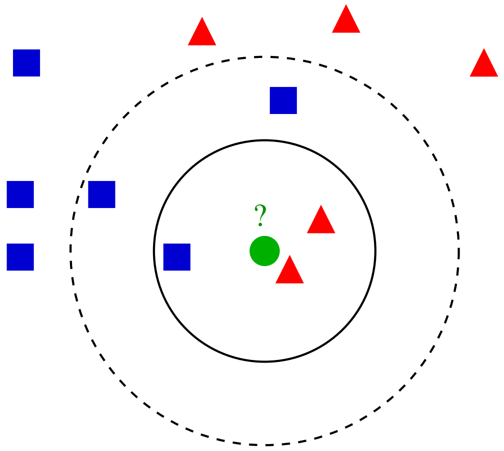


Figure 2.1: Example of k-NN classification.

The algorithm learns to map inputs to the correct outputs by minimizing the difference between its predictions and the true labels during training. This approach is commonly used for tasks such as classification, where data is assigned to predefined categories, and regression, where the goal is to predict continuous values. Supervised learning requires substantial labeled data to perform well, which can be a limitation in cases where annotating data is costly or time-consuming. In ER, supervised ML techniques are often applied when labeled pairs of matching and non-matching entities are available, enabling the model to learn specific patterns for identifying matches.

2.1.2 Unsupervised Machine Learning

Unsupervised machine learning operates on unlabeled data, seeking to discover hidden patterns or structures within the dataset. Techniques such as clustering group data points with similar characteristics, like the k-NN algorithm (illustration on Figure 2.1) algorithm, while dimensionality reduction methods like Principal component analysis (PCA) simplify data representation without losing significant information. Unsupervised learning is particularly useful when labeled data is unavailable or impractical to obtain. In the context of ER, unsupervised ML can be used to identify potential matches by grouping similar records based on features, without relying on predefined labels. This makes it a valuable tool for exploratory data analysis or initial stages of an ER pipeline.

2.2 Neural Network and BERT Architecture

2.2.1 Neural Network

Neural networks are a class of ML models inspired by the structure and functioning of the human brain, consisting of layers of interconnected nodes, or neurons, that process data and extract patterns. Each neuron performs a mathematical operation on the input it receives, applies an activation function to introduce non-linearity, and passes the result to the next layer. Neural networks are highly flexible and can approximate complex functions, making them suitable for a wide range of tasks, from image and speech recognition to Natural Language Processing (NLP). In the context of this project, neural networks—particularly modern architectures like transformers—play a crucial role in ER. Their ability to learn rich, high-dimensional representations from textual data enables them to capture subtle semantic relationships between records, achieving accuracy levels unattainable with simpler machine learning methods.

2.2.2 Transformer

The transformer architecture is a groundbreaking neural network design introduced in the paper ”Attention is All You Need” [2]. It is renowned for its ability to model sequential data, such as text, without relying on the limitations of recurrence, as in traditional Recurrent neural networks (RNNs). At its core, the transformer leverages a mechanism called self-attention, which allows the model to weigh the importance of different input tokens relative to each other, regardless of their position in the sequence. This makes transformers highly efficient for handling long-range dependencies in data.

The self-attention mechanism computes the output for each token in the input sequence by considering all other tokens. Mathematically, it can be expressed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

Here:

- Q, K, V are matrices representing the query, key, and value vectors derived from the input embeddings.

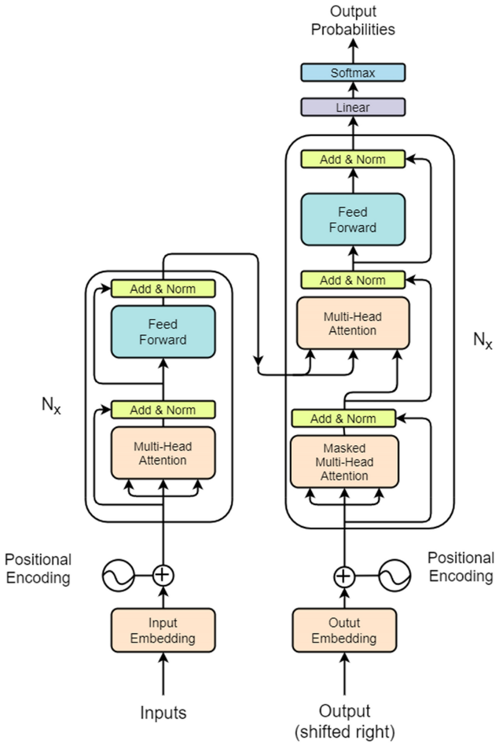


Figure 2.2: The transformer architecture [2]

- d_k is the dimensionality of the key vectors, used to scale the dot product for numerical stability.
- The result is a weighted sum of the value vectors, where the weights are determined by the similarity of the query and key vectors.

Transformers also incorporate positional encodings to inject information about the relative or absolute position of tokens, as they lack inherent sequential bias. This combination of self-attention and positional information enables transformers to excel in natural language processing tasks. The general architecture is presented on the Figure 2.2.

For this project, transformer models like BERT are crucial for ER due to their ability to encode nuanced contextual information from textual data. By capturing semantic relationships between records, these models enhance the accuracy and robustness of the resolution pipeline, surpassing traditional approaches in both precision and scalability.

2.2.3 BERT

BERT is a transformer-based model designed for natural language understanding tasks. Introduced by Google AI in 2018 [3], BERT revolutionized NLP by introducing a bidirectional approach to text encoding, allowing it to capture context from both left and right of a given token. This bidirectional capability distinguishes BERT from previous models like Generative pretrained transformer (GPT), which process text unidirectionally, and enables it to understand the full context of a word in relation to its surroundings.

BERT is built upon the transformer encoder architecture, and its design comprises several key components:

1. **Input Representation**: BERT's input is a combination of:

    - **Token embeddings**: Represent the individual words or subwords.
    - **Segment embeddings**: Distinguish between different parts of input text (e.g., two sentences in a pair).
    - **Positional embeddings**: Encode the position of each token in the sequence to introduce order.

    The final input vector for each token is the sum of these embeddings.

2. **Transformer Layers**: BERT employs multiple transformer encoder layers (e.g., 12 in BERT-base and 24 in BERT-large). Each layer comprises:

    - **Self-Attention Mechanism**: Computes relationships between all tokens. This allows BERT to consider the influence of every token on every other token bidirectionally.
    - **Feedforward Network**: A fully connected layer that applies nonlinear transformations to each token's representation.
    - **Normalization and Residual Connections**: Stabilize training and improve gradient flow.

3. **Pretraining Tasks**: BERT is pretrained using two unsupervised tasks that enable it to learn robust contextual representations:

    - **Masked Language Modeling (MLM)**: Randomly masks tokens in the input and trains the model to predict the missing words based on context.

- **Next Sentence Prediction (NSP)**: Trains the model to understand sentence relationships by predicting whether a given sentence follows another in the text.

2.2.4 SBERT

Sentence bidirectional encoder representations from transformers (SBERT) is an adaptation of BERT specifically designed for generating meaningful and efficient sentence embeddings. Introduced by Reimers and Gurevych in 2019 [4], SBERT addresses the computational limitations of BERT in pairwise comparison tasks by enabling the generation of fixed-size vector representations for sentences. These embeddings can then be compared directly using cosine similarity or other distance metrics, making SBERT highly suitable for tasks like semantic similarity, clustering, and information retrieval.

1. **Base Architecture**: SBERT builds on the BERT architecture, retaining its transformer encoder layers and the ability to capture rich contextual information. However, instead of outputting contextualized token embeddings for every word, SBERT generates a single fixed-length embedding for an entire sentence or text input. This is achieved by pooling the token embeddings from BERT, typically using either:

    - **CLS Token Pooling**: Using the embedding of the special [CLS] token from the last transformer layer.
    - **Mean Pooling**: Computing the mean of all token embeddings for the sentence.

2. **Training with Siamese/Triplet Networks**: Unlike BERT, which was not designed for efficient embedding comparison, SBERT fine-tunes BERT using Siamese or triplet network structures. In these setups:

    - Two or more identical BERT models (sharing weights) encode sentences independently into embeddings.
    - The embeddings are compared using a similarity function (e.g., cosine similarity).
    - The training objective minimizes the difference between the predicted similarity score and the ground truth (e.g., matching sentence pairs in semantic similarity tasks).

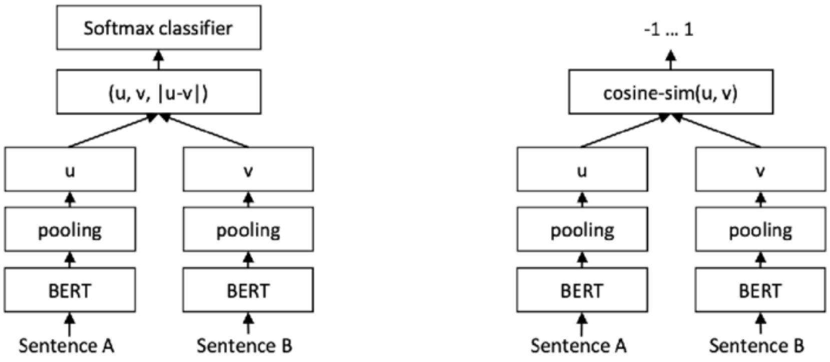


Figure 2.3: SBERT architecture with classification objective function (left) and the regression objective function (right) [4]

3. **Pretraining and Fine-tuning Tasks**: SBERT is fine-tuned on datasets specific to sentence-level tasks, such as:
   - **Natural Language Inference (NLI)**: Learning whether sentence pairs entail, contradict, or are neutral with respect to each other.
   - **Paraphrase Identification**: Training on pairs of sentences that are semantically identical or different.

Mathematical Representation

Given two sentences S_1 and S_2 , SBERT encodes them into embeddings E_1 and E_2:

$$E_1 = \text{BERT}(S_1), \quad E_2 = \text{BERT}(S_2) \tag{2.2}$$

The similarity between the sentences can then be computed using cosine similarity:

$$\text{Similarity}(S_1, S_2) = \frac{E_1 \cdot E_2}{\|E_1\| \|E_2\|} \tag{2.3}$$

This design enables SBERT to handle sentence-pair tasks with significantly reduced computational costs compared to traditional BERT approaches, which require recomputation for every pair.

The architecture is presented on the Figure 2.3.

2.3 Related work area

2.3.1 Data Matching: Definitions and Literature Insights

ER involves identifying and resolving different representations of the same real-world entities across or within datasets. This process is critical in ensuring data quality, enabling accurate analytics, and supporting decisions based on integrated data. Challenges such as missing unique identifiers, variations in data representation, and typographical errors make data matching a complex and computationally intensive task [5][6][7].

The work of Christen (2012) [5] provides foundational insights into data matching, outlining its core challenges and presenting a structured process that includes data preprocessing, blocking, and record comparison. This approach is complemented by the survey by Elmagarmid et al. (2007) [6], which categorizes various similarity metrics and algorithms for duplicate detection. These include probabilistic methods, supervised learning models, and hybrid approaches tailored for diverse data environments.

Christophides et al. (2020) [7] expand on these ideas, highlighting the complexities of entity resolution in the Big Data era. Their survey emphasizes the need for scalable workflows to manage the volume, variety, and velocity of data while addressing issues such as heterogeneity and incompleteness. They introduce schema-agnostic methods and incremental entity resolution techniques to cater to real-time and large-scale applications, underscoring the importance of innovations in blocking and matching methodologies to handle Big Data.

These studies collectively underscore the importance of designing robust and scalable systems for entity resolution, combining traditional principles with modern advancements to address the challenges posed by evolving data landscapes.

2.3.2 Deep Learning Upgrades in Data Matching

Deep learning has transformed the field of data matching (or entity resolution) by enabling the development of models that surpass traditional methods in both scalability and accuracy. Leveraging pre-trained transformer-based language models, such as BERT [3], RoBERTa, and their successors, these advancements have redefined how entities are compared and matched. These models excel at learning highly contextualized embeddings, allowing them to understand semantic relationships in a nuanced way. For instance, models like Ditto [8] have demonstrated that casting entity matching as a sequence-pair classification problem significantly enhances matching quality, achieving improvements of up to 29% in F1-score compared to prior state-of-the-art solutions [9].

One of the major advantages of deep learning is its ability to handle diverse and noisy data types, such as textual, semi-structured, and even unstructured data. Techniques like contrastive learning, as applied in frameworks such as TriBERTa, further improve entity resolution by fine-tuning embeddings to better differentiate between matching and non-matching records [10]. Moreover, the integration of domain-specific optimizations, including data augmentation, summarization of long texts, and injecting domain knowledge, has enabled models like Ditto [8] to better adapt to real-world applications [9].

Generative LLMs, such as GPT-4, offer a complementary approach by supporting zero-shot or few-shot learning, reducing the reliance on extensive labeled datasets. These models demonstrate higher robustness to unseen entities, making them suitable for cross-domain and low-resource scenarios [11][12][13].

The introduction of hybrid approaches, such as multi-task learning frameworks, also enhances both blocking and matching phases by integrating them into a cohesive pipeline. For instance, systems like BERT-ER delay the deep interaction phase, allowing for efficient blocking while retaining high accuracy during matching. Such approaches reduce the computational burden of exhaustive pairwise comparisons without sacrificing performance [14].

These advancements underscore deep learning's transformative role in data matching, providing robust, scalable, and accurate solutions that address the complexities of modern datasets. They also pave the way for future innovations in adaptive, domain-aware, and computationally efficient entity resolution systems.

2.3.3 Blocking for Entity Resolution

Blocking is a critical step in the entity resolution pipeline, designed to reduce the computational cost of pairwise comparisons by clustering potentially matching entities into smaller candidate sets. This pre-matching step ensures that only records within the same block are compared, thus avoiding the quadratic complexity of naive exhaustive comparisons. Traditional methods like key-based and rule-based blocking rely on fixed attributes or handcrafted rules to generate blocks, but they often suffer from scalability issues and are sensitive to noisy or heterogeneous data [15][16][17].

Emerging Techniques in Blocking

1. **Graph-Based Blocking:** A graph-based approach leverages pre-trained contextual embeddings to map database tuples into a vector space. Nodes in a k-NN graph represent tuples, and edges connect nodes with similar embeddings. This method utilizes community detection algorithms to create blocks, significantly reducing computational complexity while maintaining high recall rates. The use of transformer-based embeddings, such as RoBERTa and BART, further enhances the quality of blocks by capturing deep semantic similarities [15].

2. **Contrastive Learning for Blocking:** SC-Block employs supervised contrastive learning to position matching records closer in an embedding space. This approach enhances traditional blocking by training on existing labeled data, producing smaller candidate sets while preserving high F1 scores. SC-Block integrates seamlessly with downstream matching models and has demonstrated significant runtime reductions in large-scale benchmarks [18].

3. **Universal Dense Blocking:** The UniBlocker framework introduces domain-independent pre-training for dense blocking, leveraging self-supervised contrastive learning on large-scale tabular data. Unlike traditional domain-specific approaches, UniBlocker adapts to various domains without additional fine-tuning, making it highly efficient for large and heterogeneous datasets [19].

4. **Hybrid Approaches:** ShallowBlocker bridges classical and modern blocking methods by combining set similarity joins with lightweight filtering techniques. It integrates absolute and relative similarity metrics to balance recall and precision, offering a hands-off, interpretable solution that competes with deep learning methods in effectiveness and scalability [20].

5. **Semantic-Aware Meta-Blocking:** SeMBlock enhances traditional blocking by incorporating semantic relationships using locality-sensitive hashing (LSH) with transformer-based embeddings like BERT. By constructing semantic-aware graphs and pruning edges, SeMBlock reduces unnecessary comparisons while maintaining high precision and recall [17].

2.4 Summary

ER has been extensively studied, with traditional methods relying on rule-based and statistical models for data matching. While effective in structured scenarios, these approaches struggle with scalability and noisy data. Recent advancements in deep learning have significantly improved entity resolution, particularly in matching and blocking tasks. Transformer-based models, such as BERT [3] and SBERT [4], enable capturing semantic nuances, greatly enhancing matching precision and robustness. Methods like Ditto and TriBERTa exemplify this progress by integrating domain-specific optimizations and contrastive learning techniques.

Blocking, a crucial step for scalability, has also evolved with novel techniques. Graph-based approaches, such as k-NN graphs, leverage pre-trained embeddings to cluster records effectively. Dense blocking methods like UniBlocker utilize pre-trained transformers and self-supervised learning to handle diverse datasets without fine-tuning. Hybrid solutions, such as ShallowBlocker and SC-Block, combine classical and deep learning techniques, striking a balance between efficiency and effectiveness. Semantic-aware frameworks like SeMBlock incorporate transformer embeddings with locality-sensitive hashing, further improving blocking accuracy and reducing computational costs. In my project, I will only rely on graph based blocking techniques thanks to the simplicity and effectiveness of it.

These developments underscore the transformative role of modern machine learning and embedding-based approaches in improving the performance, scalability, and applicability of entity resolution pipelines across diverse domains.

Moreover, the Papers with Code benchmark for ER [1] provides a centralized repository of results from various entity matching papers, enabling researchers to compare their methods against state-of-the-art approaches. This benchmark will serve as a valuable resource for evaluating the performance of the methods developed in this project within the broader research community.

Chapter 3

Methods

The purpose of this chapter is to provide an overview of the research method used in this thesis. Section 3.1 describes the research process. Section 3.2 details the research paradigm. Section 3.3 focuses on the data collection techniques used for this research. Section 3.4 describes the experimental design. Section 3.5 explains the techniques used to evaluate the reliability and validity of the data collected. Section 3.6 describes the method used for the data analysis. Finally, Section 3.7 describes the framework selected to evaluate the ER.

3.1 Research Process

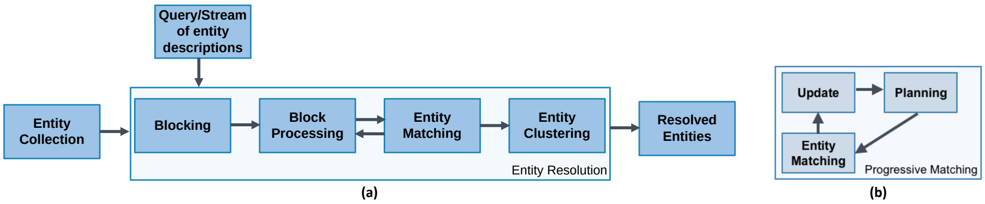


Figure 3.1: (a) The generic end-to-end workflow for Entity Resolution. (b) Budget-aware Matching. [7]

The research process for this project follows a structured workflow designed to address the key challenges of scalability and performance in ER. This workflow incorporates best practices from machine learning and data science, with a focus on optimizing blocking and matching techniques (described in 3.1. The steps are as follows:

1. **Data Loading and Preprocessing:**
   - Load datasets from the CompER benchmark [21], ensuring compatibility with the entity resolution pipeline.
   - Perform exploratory data analysis (EDA) to understand the structure, types of features, missing values, and relevant attributes.
   - Preprocess the data, including handling missing values, encoding categorical features, normalizing textual data, and splitting datasets into train, validation, and test sets.
2. **Blocking Implementation:**
   - Develop and implement blocking techniques to improve scalability by reducing the number of candidate pairs.
   - Experiment with blocking methods such as k-NN and other embedding-based approaches.
   - Evaluate the effectiveness of the blocking stage by measuring the reduction in candidate pairs and the recall of potential matches.
3. **Matching and Model Training:**
   - Fine-tune transformer models such as BERT [3] or SBERT [4] to perform the matching of candidate pairs generated during the blocking stage.
   - Test supervised and unsupervised approaches, including zero-shot learning methods, to assess their applicability in scenarios with limited labeled data.
   - Optimize model parameters and architecture to balance performance and computational efficiency.
4. **Evaluation and Testing:**
   - Evaluate the pipeline using key metrics such as precision, recall, F1-score, and accuracy on the test set.
   - Measure computational performance, including training time, inference time, and scalability as the dataset size increases.
   - Compare results against existing state-of-the-art methods to ensure the proposed solutions achieve improved accuracy and efficiency.
5. **Iterative Refinement:**

   - Use insights from evaluation to refine both the blocking and matching stages.
   - Implement additional preprocessing or feature engineering techniques based on identified bottlenecks.
   - Explore alternate models or architectures to further enhance scalability and performance.

6. **Documentation and Deployment:**

   - Document the entire workflow, including key decisions, methods, and results.
   - Provide detailed code and instructions to ensure reproducibility and ease of reuse in similar tasks.

3.2 Research Paradigm

The research is guided by a pragmatic research paradigm, emphasizing practical solutions to real-world problems in ER. This paradigm aligns with the project's goals of improving both scalability and overall performance while maintaining applicability to diverse datasets and scenarios.

1. **Philosophical Assumptions:**

   - The project adopts a problem-solving approach, focusing on achieving measurable improvements in ER pipelines.
   - It values the integration of theory (e.g., transformer-based model advancements) with practical experimentation and application.

2. **Methodological Framework:**

   - Combines quantitative evaluation (e.g., metrics-based assessment) with experimental analysis to optimize each stage of the pipeline.
   - Employs iterative development, allowing for continuous improvement based on empirical results.

3. **Ethics and Societal Issues:**

   (a) **Data Privacy:** Recognizing that entity resolution often involves sensitive data, the project exclusively uses publicly available datasets (e.g., CompER) to avoid privacy concerns. The methodology ensures compliance with data protection regulations such as GDPR.

(b) **Bias and Fairness:** The potential for bias in datasets and algorithms is acknowledged. Efforts are made to evaluate and mitigate any biases in the training data and the resulting models to avoid unfair outcomes in real-world applications.

(c) **Environmental Impact:** The high computational demands of LLMs raise concerns about sustainability. The project incorporates efficient algorithmic optimizations and explores lightweight models wherever feasible to reduce the environmental footprint.

(d) **Social Implications:** Entity resolution systems can significantly influence societal functions, from healthcare to public policy. Ensuring transparency, fairness, and usability of the developed pipeline supports ethical deployment and societal benefits.

4. **Reproducibility and Scalability:**

   - Prioritizes reproducibility by using publicly available datasets (e.g., CompER) and fixed train-test splits.
   - Focuses on scalability as a core objective, ensuring the developed methods can handle increasing data volumes and complexity.

By integrating ethical considerations with technical goals and adhering to a pragmatic paradigm, the project aims to develop a scalable, efficient, and ethically responsible entity resolution pipeline capable of addressing real-world challenges.

3.3 Data Collection

The primary dataset utilized in this project is the CompER benchmark [21], which serves as a comprehensive resource for evaluating ER models. The CompER benchmark [21] is a collection of 21 entity matching tasks designed to evaluate, compare, and improve the reproducibility of matching methods. These tasks span diverse domains, such as products, restaurants, and scholarly articles, offering a variety of datasets with predefined training, validation, and test splits. CompER addresses common limitations in existing benchmarks by standardizing the data splits and providing both matching and non-matching pairs. This consistency supports fair and reproducible comparisons across different ER methods.

The choice of a public benchmark dataset like CompER aligns with the project's goal of adhering to ethical and transparent practices. By relying on publicly available data, the project avoids the need for proprietary or sensitive data, thereby minimizing privacy concerns. Nevertheless, social and ethical considerations remain relevant. Although the dataset itself is publicly accessible, it is crucial to evaluate whether any inherent biases exist in the data, such as over-representation or under-representation of specific entities, which could affect the fairness and generalizability of the model. Moreover, the project adheres to strict data usage guidelines, ensuring compliance with any licensing terms associated with the dataset and avoiding any misuse of the information contained within.

3.4 Experimental design and Planned Measurements

3.4.1 Test environment/test bed/model

The aim of this project is to create a POC for an ER framework that can be used on the fly. To ensure reproducibility, the implementation relies heavily on Python, leveraging widely adopted libraries such as PyTorch [22] and Transformers [23] from the Hugging Face ecosystem. These libraries provide the tools needed to build, fine-tune, and deploy state-of-the-art transformer models for the ER task.

The models used in this project are open-source transformer models available on the Hugging Face platform. These pre-trained models were chosen for their proven performance and flexibility in natural language processing tasks. Specific models and configurations are documented in the appendix to facilitate reproduction.

For computational resources, the training process utilized NVIDIA GPUs provided by my home institution to accelerate the fine-tuning of the models. While GPU access significantly reduces training time and enhances scalability, it is not a strict requirement. All models and scripts are designed to be compatible with CPUs, albeit with longer processing times. Additionally, key Python code excerpts are included in the appendix to provide further clarity on the implementation.

To replicate the test environment, a system with Python 3.8 or later is recommended. Installing PyTorch, Transformers, and other necessary dependencies can be done using package managers like pip. The exact versions of these libraries, along with the installation steps, are detailed in the appendix to ensure consistency and compatibility. With this setup, others can reproduce the results and experiment with modifications to the ER pipeline as needed.

3.4.2 Hardware/Software to be used

This project was developed using a combination of hardware and software tools to create a robust and efficient test environment. For hardware, the primary development was conducted on a personal computer connected to linux virtual machines, with sufficient processing power and memory to handle the requirements of the project. In addition, high-performance GPUs provided by my home institution were utilized to accelerate model training and experimentation. These GPUs, equipped with substantial VRAM, significantly reduced computation time and facilitated the fine-tuning of large transformer models. However, the framework is designed to be flexible and can also be executed on CPU-only systems, albeit with longer processing times.

On the software side, the project relied heavily on Python as the programming language, due to its extensive ecosystem of libraries and ease of use for machine learning tasks. Key libraries included PyTorch and the Hugging Face Transformers library, which provided the tools necessary for working with state-of-the-art language models. For data manipulation and preprocessing, libraries like Pandas [24] and NumPy [25] were essential, while Scikit-learn [26] was used for implementing evaluation metrics. Jupyter Notebooks served as the primary development interface, allowing for iterative testing and debugging. The project also made use of virtual environments to manage dependencies and ensure consistency across different setups. This combination of hardware and software created a reliable and scalable environment for developing, testing, and refining the ER pipeline.

3.5 Assessing reliability and validity of the data collected

3.5.1 Validity of method

The validity of the method used in this project is ensured by relying on the CompER benchmark [21], a well-prepared dataset specifically designed for evaluating ER models. The benchmark allows direct comparisons between the developed model and other state-of-the-art approaches, providing an objective measure of performance. However, while benchmarks offer consistency and reproducibility, they may not fully capture the complexities of real-world ER problems, where data is often messy, diverse, and unstructured. To address this, the models will be rigorously tested on the benchmark's predefined test set to confirm their accuracy and generalizability. Additionally, to bridge the gap between benchmark evaluation and practical application, a full pipeline integrating supervised or unsupervised methods will be implemented. This approach ensures that the model is not only valid in a controlled environment but also meaningful in real-world scenarios.

3.5.2 Reliability of method

The reliability of the method is supported by the fixed splits of the CompER benchmark [21], which ensure that the same training, validation, and test sets are used across different experiments. This consistency allows for the repeatability of results, as identical conditions are maintained. To further enhance reliability during development, random seed values will be fixed for all processes involving randomness, such as data shuffling and weight initialization. This ensures that the model's performance remains consistent across multiple runs under the same conditions. By adopting these practices, the methodology guarantees a high level of reliability, enabling reproducibility and fair evaluation of the results.

3.5.3 Data validity

The validity of the CompER benchmark [21] stems from its well-defined structure and comprehensive coverage of real-world matching scenarios. Each dataset is curated with clear ground truth labels for both matches and non-matches, ensuring accuracy in evaluation. The inclusion of diverse datasets, ranging from highly structured to semi-structured and textual data, ensures that the benchmark tests methods across a wide range of challenges. Furthermore, the profiling of datasets based on dimensions such as schema complexity, textuality, and sparsity allows researchers to understand the specific challenges associated with each task. However, it is important to note that while the benchmark is representative of typical ER problems, it may not fully capture the nuances of more complex, domain-specific real-world scenarios

3.5.4 Reliability of data

The reliability of the CompER benchmark [21] is reinforced by its standardized fixed splits for training, validation, and testing. This eliminates variability and ensures that experimental results are reproducible across different studies and implementations. The inclusion of fixed non-matching pairs further enhances reliability, as these are generated using a consistent heuristic, reducing the risk of researcher bias. Additionally, detailed documentation and profiling dimensions ensure transparency, making it easier to replicate experiments under identical conditions. This high degree of reliability makes CompER a trusted resource for benchmarking ER models.

3.6 Planned Data Analysis

3.6.1 Data Analysis Technique

Although the data in the CompER benchmark [21] has already been analyzed to some extent, further EDA will be performed to gain deeper insights into the datasets and prepare them for effective model training. The analysis will begin with an overview of the datasets, examining key aspects such as the number of columns, the type of features (e.g., categorical, numerical, textual), and the presence of missing values. This step ensures a thorough understanding of the structure and quality of the data. Additionally, the relevance of each feature to the ER task will be assessed, identifying which attributes contribute most to distinguishing between matching and non-matching pairs.

Further analysis may involve serialized techniques to preprocess the data. For example, datasets might be analyzed both with and without column names to test how attribute semantics influence the model's performance. Features may also be re-ordered or grouped differently to evaluate the robustness of the model to changes in input structure. Techniques like missing value imputation, feature selection, and encoding categorical variables will also be employed as needed to standardize and prepare the data for model input.

These additional steps aim to complement the benchmark's baseline evaluation by identifying opportunities to optimize feature engineering and assess the impact of various preprocessing strategies on the ER pipeline's performance.

3.6.2 Software Tools

The development and implementation of this project rely on a suite of software tools to facilitate efficient coding, testing, and collaboration. Python serves as the primary programming language due to its versatility and widespread use in machine learning and data processing. Development is conducted interactively using Jupyter Notebooks, which enable iterative testing and visualization of results, while Visual Studio Code is employed for managing and editing larger scripts and maintaining the overall project structure.

Version control is managed through Git, ensuring that changes are tracked, and collaboration is seamless. To maintain consistency across different environments, virtual environments are created using tools like virtualenv, which help isolate dependencies and ensure reproducibility. Together, these high-level tools provide a robust, scalable, and adaptable software environment for developing the entity resolution pipeline.

3.7 Evaluation framework

The evaluation of the models developed in this project will be conducted using a comprehensive framework designed to measure both their performance and practical usability. Key metrics include precision, recall, F1-score, and accuracy, which provide insights into the effectiveness of the model in correctly identifying matching and non-matching pairs. These metrics are particularly important in the context of ER, where false positives and false negatives can have significant implications.

Beyond standard performance metrics, the framework will also assess the training time and inference time to evaluate the model's efficiency and scalability. The number of parameters in the model will be considered, along with the computational complexity, to ensure that the solution remains practical for real-world applications. Scalability testing will be performed by observing the model's performance and resource requirements as the dataset size increases, ensuring it can handle larger and more complex datasets.

The evaluation process will leverage tools from the Scikit-learn framework [26], which provides robust implementations for calculating key metrics and generating detailed reports. This comprehensive evaluation framework ensures that the models are not only accurate but also efficient and scalable, meeting both theoretical and practical requirements for ER tasks.

3.8 System documentation

Comprehensive system documentation will be provided to ensure the reproducibility, clarity, and reusability of the project. A detailed documentation package will be included in the appendix, offering a complete overview of the system's architecture, methodologies, and implementation details. This documentation will serve as a guide for replicating the experiments and adapting the code for future use in similar tasks.

The Python code and Jupyter Notebooks will be meticulously documented, following best practices to maintain readability and usability. Inline comments and structured docstrings will explain the purpose and functionality of each module, function, and class. Additionally, the Notebooks will contain step-by-step instructions, including code explanations, intermediate outputs, and visualizations, making the development process transparent and accessible.

The documentation will also provide installation instructions, software requirements, and a guide for setting up the environment using virtual environments. This ensures that users can seamlessly replicate the test environment. By creating detailed and clear documentation, the project becomes a reusable resource for future research and practical applications in ER.

Chapter 4

What you did

4.1 Hardware/Software design …/Model/Simulation model & parameters/…

Figure 4.1 shows a simple icon for a home page. The time to access this page when served will be quantified in a series of experiments. The configurations that have been tested in the test bed are listed in Table 4.1. In 7.0 % of cases, there was an error indicating xxxxx.

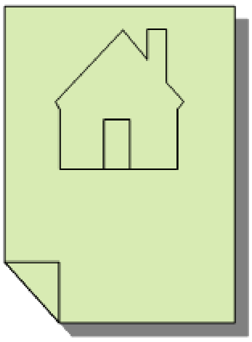


Figure 4.1: Homepage icon

4.2 Implementation …/Modeling/Simulation/…

Two commonly used simulators are:

Mininet This simulator uses traffic control (`tc`) to simulate

Table 4.1: Configurations tested

Configuration	Description
1	Simple test with one server
2	Simple test with one server

network devices connected by links with specific bandwidth, packet loss rates, qdisc methods, etc.

ns-2 or ns-3 simulator These simulators are very useful for simulating wireless communication links between moving devices. You can specify the mobility patterns of the nodes.

4.2.1 Some examples of coding

Listing 4.1 shows an example of a simple program written in C code.

Listing 4.1: Hello world in C code

```
int main() {
printf("hello, world");
return 0;
}
```

In contrast, Listing 4.2 is an example of code in Python to get a list of all of the programs at KTH.

Listing 4.2: Using a python program to access the KTH API to get all of the programs at KTH

```
KOPPSbaseUrl = 'https://www.kth.se'

def v1_get_programmes():
    global Verbose_Flag
    #
    # Use the KOPPS API to get the data
    # note that this returns XML
    url = "{0}/api/kopps/v1/programme".format(KOPPSbaseUrl)
    if Verbose_Flag:
        print("url: " + url)
```

```
#
r = requests.get(url)
if Verbose_Flag:
    print("result␣of␣getting␣v1␣programme:␣{}".format(r.text)
#
if r.status_code == requests.codes.ok:
    return r.text           # simply return the XML
#
return None
```

4.2.2 Some examples of figures in tikz

These figures are just some examples to show that you can draw your own figures for in your thesis. This has two advantages: (i) you do not have to worry about copyrights – as these are your own figures and (ii) the text is now readable and not simply a picture of text – so screen readers can read the figure's contents to someone who is listening to the contents of your thesis.

4.2.2.1 Azure's Form Recognizer

Figure 4.2 shows the processing of key-value extraction from a PDF document using Azure's Form Recognizer.

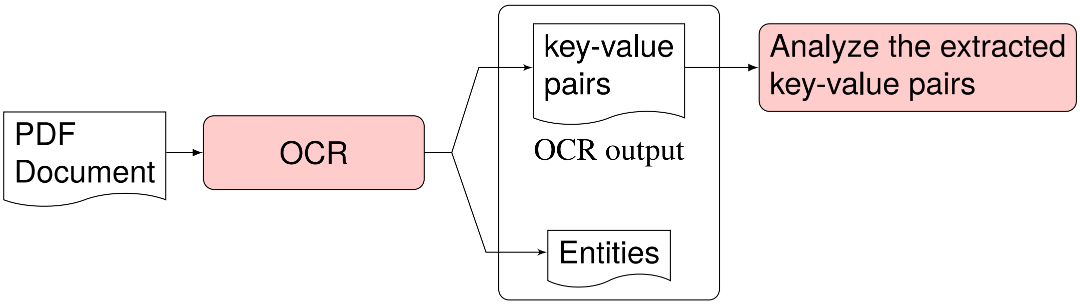


Figure 4.2: The processing of key-value extraction from a PDF document using Azure's Form Recognizer

4.2.2.2 Hyper-V with Containers

Figure 4.3 shows how Hyper-V deals with containers.

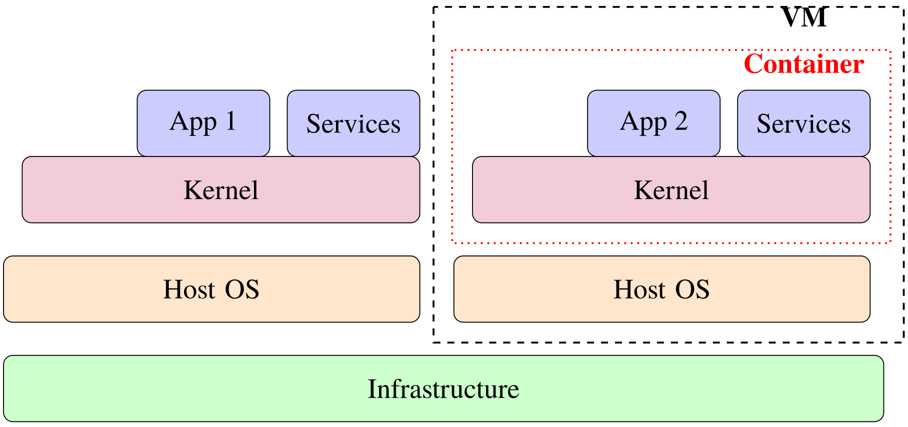
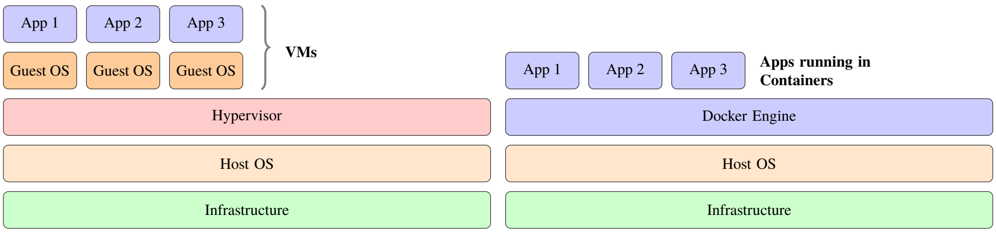


Figure 4.3: Hyper-V with containers

4.2.2.3 VM versus Containers

Figure 4.4 shows a comparison of virtual machines (VMs) versus containers.



(a) VM

(b) Containers

Figure 4.4: Virtual machines (VMs) versus Containers

Chapter 5

Results and Analysis

In this chapter, we present the results and discuss them.

5.1 Major results

Some statistics of the delay measurements are shown in Table 5.1. The delay has been computed from the time the GET request is received until the response is sent.

Table 5.1: Delay measurement statistics

Configuration	**Average delay (ns)**	**Median delay (ns)**
1	467.35	450.10
2	1 687.5	901.23

Table 5.2 shows the measurement of round trip times from four hosts to and from a server.

Table 5.2: Result for the ping measurements of RTT for 4 hosts

	host to server RTT in ms			
Host	min	avg	max	mdev
h1	5.625	5.625	5.625	0.0
h2	2.909	2.909	1.909	0.0
h3	5.007	5.007	5.007	0.0
h4	2.308	2.308	2.308	0.0

Figure 5.1 shows an example of the performance as measured in the experiments.

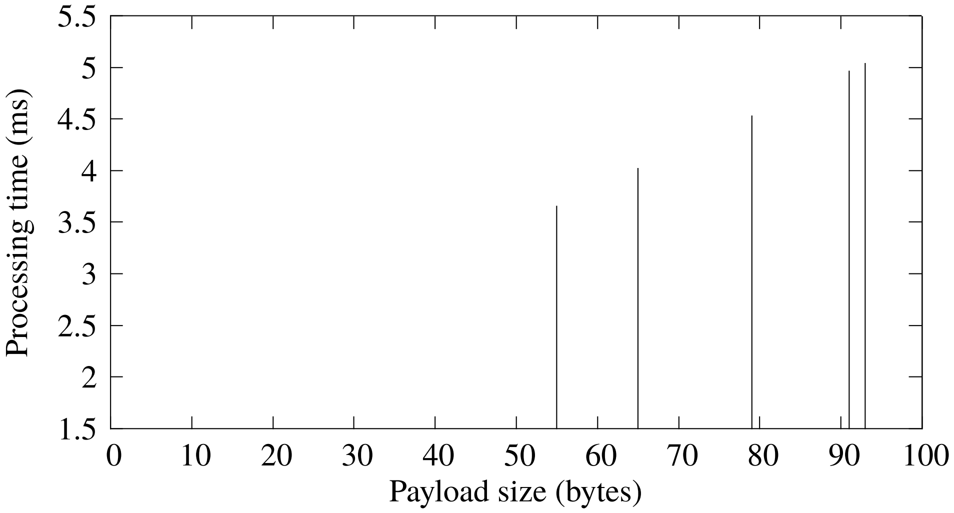


Figure 5.1: Processing time vs. payload length

Given these measurements, we can calculate our processing bit rate as the inverse of the time it takes to process an additional byte divided by 8 bits per byte:

$$\text{bit rate} = \frac{1}{\frac{\text{time}_{\text{byte}}}{8}} = 20.03 \quad kb/s$$

Table 5.3 shows another table in which some values have been set in bold (using \B) to emphasize them. Note how the S formatting has been modified so that it considers the weight of the characters and this is able to decimal align even these hold-faced numbers with the numbers in the column above them.

Table 5.3: Median values of sandwich attributes

	sites	
Attribute	**A**	**B**
price (in SEK)	36.5	71.3
protean (g)	97.2	100.0
salt (mg)	9.7	9.3
Average customer rating in %	**82.2**	**89.9**

Figure 5.2 shows a stacked bar chart using pgfplots. It illustrates how easy it is to take a set of data and make a stacked bar plot. One of the features is the shifted values – this is very useful when the bar itself is too small to put the value into.

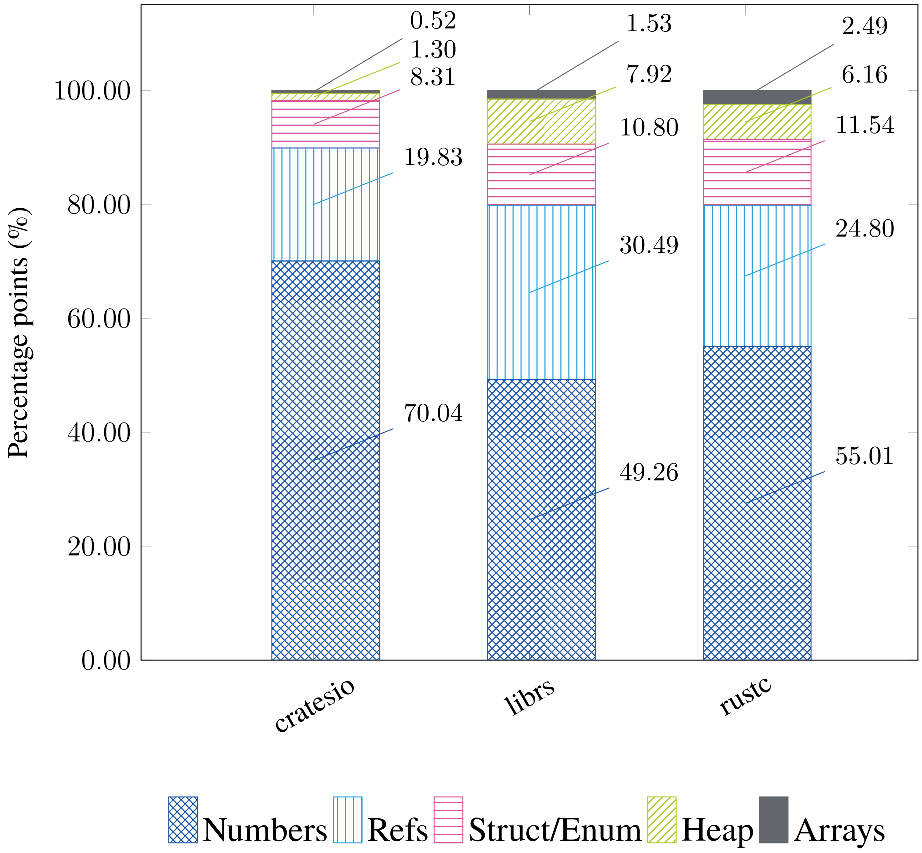


Figure 5.2: Rust types distribution for the compiler, crates.io, and lib.rs. (percentage) - appears here with the permission of the author - see the thesis at `https://urn.kb.se/resolve?urn=urn%3Anbn%3Ase%3Akth%3Adiva-332124`

5.2 Reliability Analysis

5.3 Validity Analysis

Chapter 6

Discussion

Chapter 7

Conclusions and Future work

7.1 Conclusions

7.2 Limitations

7.3 Future work

Due to the breadth of the problem, only some of the initial goals have been met. In these section we will focus on some of the remaining issues that should be addressed in future work. ...

7.3.1 What has been left undone?

The prototype does not address the third requirment, *i.e.,* a yearly unavailability of less than 3 minutes; this remains an open problem. ...

7.3.1.1 Cost analysis

The current prototype works, but the performance from a cost perspective makes this an impractical solution. Future work must reduce the cost of this solution; to do so, a cost analysis needs to first be done. ...

7.3.1.2 Security

A future research effort is needed to address the security holes that results from using a self-signed certificate. Page filling text mass. Page filling text mass. ...

7.3.2 Next obvious things to be done

In particular, the author of this thesis wishes to point out xxxxxx remains as a problem to be solved. Solving this problem is the next thing that should be done. ...

7.4 Reflections

One of the most important results is the reduction in the amount of energy required to process each packet while at the same time reducing the time required to process each packet.

The thesis contributes to the United Nations (UN) Sustainable Development Goals (SDGs) numbers 1 and 9 by xxxx.

References

[1] "Papers with Code - Entity Resolution — paperswithcode.com," https://paperswithcode.com/task/entity-resolution, [Accessed 18-11-2024]. [Pages 4 and 19.]

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762 [Pages xiii, 11, and 12.]

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019. [Online]. Available: https://arxiv.org/abs/1810.04805 [Pages 13, 16, 19, and 22.]

[4] 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*. Association for Computational Linguistics, Nov. 2019. [Online]. Available: https://arxiv.org/abs/1908.10084 [Pages xiii, 14, 15, 19, and 22.]

[5] P. Christen, "Data matching," in *Data-Centric Systems and Applications*, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:59924038 [Page 16.]

[6] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, pp. 1–16, 2007. [Online]. Available: https://api.semanticscholar.org/CorpusID:386036 [Page 16.]

[7] V. Christophides, V. Efthymiou, T. Palpanas, G. Papadakis, and K. Stefanidis, "End-to-end entity resolution for big data: A survey," 2020. [Online]. Available: https://arxiv.org/abs/1905.06397 [Pages xiii, 16, and 21.]

[8] Y. Li, J. Li, Y. Suhara, A. Doan, and W.-C. Tan, “Deep entity matching with pre-trained language models,” *Proceedings of the VLDB Endowment*, vol. 14, no. 1, pp. 50–60, Sep. 2020. doi: 10.14778/3421424.3421431. [Online]. Available: https://arxiv.org/pdf/2004.00584 [Page 17.]

[9] ——, “Effective entity matching with transformers,” *The VLDB Journal*, vol. 32, no. 6, pp. 1215–1235, Jan. 2023. doi: 10.1007/s00778-023-00779-z. [Online]. Available: https://doi.org/10.1007/s00778-023-00779-z [Page 17.]

[10] B. T. Foua, J. R. Talburt, and X. Xu, “Large language model-based representation learning for entity resolution using contrastive learning,” in *2023 International Conference on Computational Science and Computational Intelligence (CSCI)*, 2023. doi: 10.1109/CSCI62032.2023.00010 pp. 15–22. [Page 17.]

[11] R. Peeters and C. Bizer, “Entity matching using large language models,” 2024. [Online]. Available: https://arxiv.org/abs/2310.11244 [Page 17.]

[12] H. Li, L. Feng, S. Li, F. Hao, C. J. Zhang, Y. Song, and L. Chen, “On leveraging large language models for enhancing entity resolution,” 2024. [Online]. Available: https://arxiv.org/abs/2401.03426 [Page 17.]

[13] R. Wu, S. Chaba, S. Sawlani, X. Chu, and S. Thirumuruganathan, “Zeroer: Entity resolution using zero labeled examples,” in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD/PODS ’20. ACM, May 2020. doi: 10.1145/3318464.3389743 p. 1149–1164. [Online]. Available: http://dx.doi.org/10.1145/3318464.3389743 [Page 17.]

[14] B. Li, Y. Miao, Y. Wang, Y. Sun, and W. Wang, “Improving the efficiency and effectiveness for bert-based entity resolution,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 15, pp. 13 226–13 233, May 2021. doi: 10.1609/aaai.v35i15.17562. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17562 [Page 17.]

[15] J. B. Mugeni and T. Amagasa, “A graph-based blocking approach for entity matching using pre-trained contextual embedding models,” in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, ser. SAC ’22. New York, NY, USA: Association

for Computing Machinery, 2022. doi: 10.1145/3477314.3507689. ISBN 9781450387132 pp. 357–364. [Online]. Available: https://doi.org/10.1145/3477314.3507689 [Page 18.]

[16] G. Papadakis, J. Svirsky, A. Gal, and T. Palpanas, "Comparative analysis of approximate blocking techniques for entity resolution," *Proc. VLDB Endow.*, vol. 9, no. 9, pp. 684–695, May 2016. doi: 10.14778/2947618.2947624. [Online]. Available: https://doi.org/10.14778/2947618.2947624 [Page 18.]

[17] D. Javdani, H. Rahmani, and G. Weiss, "Semblock: A semantic-aware meta-blocking approach for entity resolution," *Intelligent Decision Technologies*, vol. 15, no. 3, pp. 461–468, 2021. doi: 10.3233/IDT-200207. [Online]. Available: https://doi.org/10.3233/IDT-200207 [Page 18.]

[18] A. Brinkmann, R. Shraga, and C. Bizer, "Sc-block: Supervised contrastive blocking within entity resolution pipelines," 2023. [Online]. Available: https://arxiv.org/abs/2303.03132 [Page 18.]

[19] T. Wang, H. Lin, X. Han, X. Chen, B. Cao, and L. Sun, "Towards universal dense blocking for entity resolution," 2024. [Online]. Available: https://arxiv.org/abs/2404.14831 [Page 18.]

[20] N. Barlaug, "Shallowblocker: Improving set similarity joins for blocking," 2023. [Online]. Available: https://arxiv.org/abs/2312.15835 [Page 18.]

[21] A. Primpeli and C. Bizer, "Profiling entity matching benchmark tasks," in *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, ser. CIKM '20. New York, NY, USA: Association for Computing Machinery, 2020. doi: 10.1145/3340531.3412781. ISBN 978-1-4503-6859-9/20/10. [Online]. Available: https://data.dws.informatik.uni-mannheim.de/benchmarkmatchingtasks/index.html#toc1 [Pages 22, 24, 27, and 28.]

[22] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019. [Online]. Available: https://arxiv.org/abs/1912.01703 [Page 25.]

[23] 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. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6 [Page 25.]

[24] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, Stéfan van der Walt and Jarrod Millman, Eds., 2010. doi: 10.25080/Majora-92bf1922-00a pp. 56 – 61. [Page 26.]

[25] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. doi: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2 [Page 26.]

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Pages 26 and 29.]

Appendix A

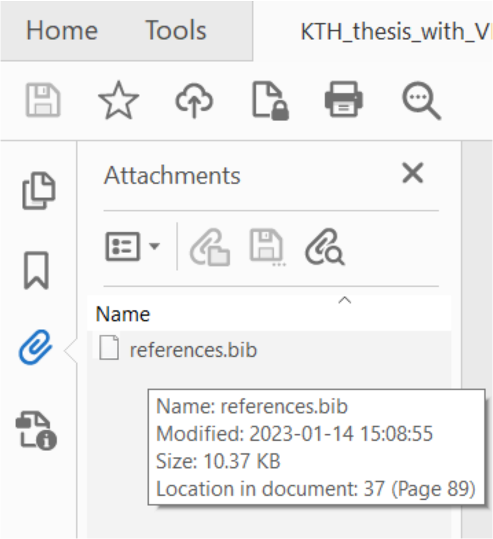
Supporting materials

The BibTeX references used in this thesis are attached.

Some source code relevant to this project can be found at https://github.com/gqmaguirejr/E-learning and https://github.com/gqmaguirejr/Canvas-tools.

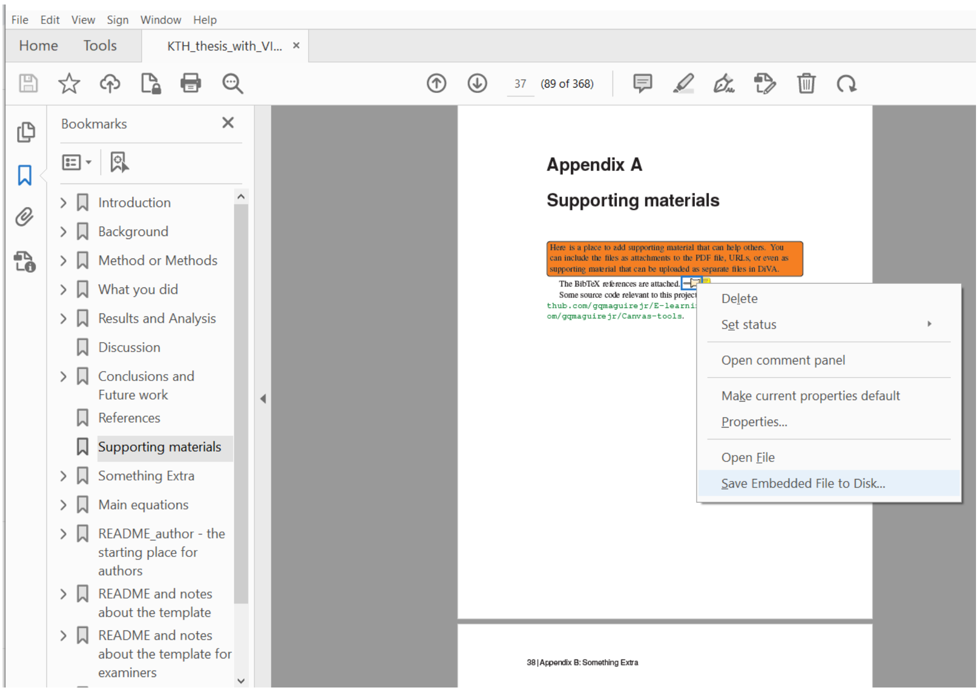
Your reader can access the attached (embedded) files using a PDF tool such as Adobe Acrobat Reader using the paperclip icon in the left menu, as shown in Figure A.1 or by right-clicking on the push-pin icon in the PDF file and then using the menu to save the embedded file as shown in Figure A.2.

An argument for including supporting material in the PDF file is that it will be available to anyone who has a copy of the PDF file. As a result, they do not have to look elsewhere for this material. This comes at the cost of a larger PDF file. However, the embedded files are encoded into a compressed stream within the PDF file; thus, reducing the number of additional bytes. For example, the references.bib file that was used in this example is 10 617 B in size but only occupies 4 261 B in the PDF file.



Name: references.bib
Modified: 2023-01-14 15:08:55
Size: 10.37 KB
Location in document: 37 (Page 89)

Figure A.1: Adobe Acrobat Reader using the paperclip icon for the attached references.bib file



Appendix A

Supporting materials

Here is a place to add supporting material that can help others. You can include the files as attachments to the PDF file, URLs, or even as supporting material that can be uploaded as separate files in DiVA.

The BibTeX references are attached.

Some source code relevant to this project

thub.com/gqmaguirejr/E-learni

om/gqmaguirejr/Canvas-tools.

Figure A.2: Adobe Acrobat Reader after right-clicking on the push-pin icon for the attached references.bib file

Appendix B

Something Extra

B.1 Just for testing KTH colors

You have selected to optimize for print output

- Primary color
  - kth-blue
  - kth-blue80
- Secondary colors
  - kth-lightblue
  - kth-lightred
  - kth-lightred80
  - kth-lightgreen
  - kth-coolgray
  - kth-coolgray80

black

Appendix C

Main equations

This appendix gives some examples of equations that are used throughout this thesis.

C.1 A simple example

The following example is adapted from Figure 1 of the documentation for the package nomencl (https://ctan.org/pkg/nomencl).

$$a = \frac{N}{A} \tag{C.1}$$

The equation $\sigma = ma$ follows easily from Equation (C.1).

C.2 An even simpler example

The formula for the diameter of a circle is shown in Equation (C.2) area of a circle in eq. (C.3).

$$D_{circle} = 2\pi r \tag{C.2}$$

$$A_{circle} = \pi r^2 \tag{C.3}$$

Some more text that refers to (C.3).

Appendix D

README_author - the starting place for authors

This document, written by Gerald Q. Maguire Jr, describes the thesis template that I have developed for use at KTH Royal Institute of Technology (KTH). It is important to note that the template is **not prescriptive**, as not every thesis will have all of the parts that the template shows. However, if there is something that you decide to leave out, you should make a conscious decision to do so and you should consider the impact this may have on your thesis being approved by the examiner.

Fundamental to the design of the template are several key factors:

- Helping students be successful in their degree project,
- Helping students produce a high-quality thesis, and
- Supporting all of the (relevant) phases of the degree project process.

This document is a work in progress.

D.1 Advice for Author or Authors

One of the hardest problems an author faces is getting started writing, *i.e.,* the blank sheet of paper – empty file barrier. The template provides a non-blank starting point; hence, avoiding the blank paper barrier. Additionally, the template provides some initial structure, basically, an Introduction, Methods, Results, and Discussion (IMRAD) structure, so that there are hints of where to place material. Moreover, there are places (and notes) about material that the student should consider adding; for example, the “required reflections” section in the final chapter.

The template (located in the file `examplethesis.tex`) also provides some examples of commonly occurring types of content, so that one can easily find examples of how to include a figure, table, code listing, *etc*. These examples are not meant to be exhaustive and quite often the student will probably need to learn new LaTeX commands in the course of writing their thesis.

As an author, the first step is to configure the LaTeX engine that you will use to process the files - see Appendix D.2. The second step will be to configure the template - see Appendix D.3. The third step will be to make sure that the information about you, your supervisor(s), and the examiner are correct in the file `custom_configuration.tex` - this information uses the macros described in Appendix D.4. Now that you have a lot of the administrative details taken care of it is time to start to write - see Appendix D.6.

Note that if you are using Overleaf:

Make your own copy of the template If you have opened the template from a URL, in the upper left-hand corner, click on **Menu**. Then select **Copy Project** - this will give you your own private copy.

Use a helpful project name I suggest you include your name in the project name so that when you share it with your supervisor(s) and examiner, they will know it is your project.

Invite your supervisor(s9 and examiner to your project You can invite your supervisor(s) and examiner to your project and they can directly comment on and correct your drafts.

Log in to Overleaf with your KTH account If you log in to Overleaf with your KTH account, you get a version of Overleaf that lets you turn on “Track changes” which is very useful (particularly if you have invited your supervisor(s) and examiner to join your project). It also gives you a bit more of a time budget to compile (which can be useful if you have a lot of Tikz figures or other things that take a lot of time for the LaTeX engine to render).

.

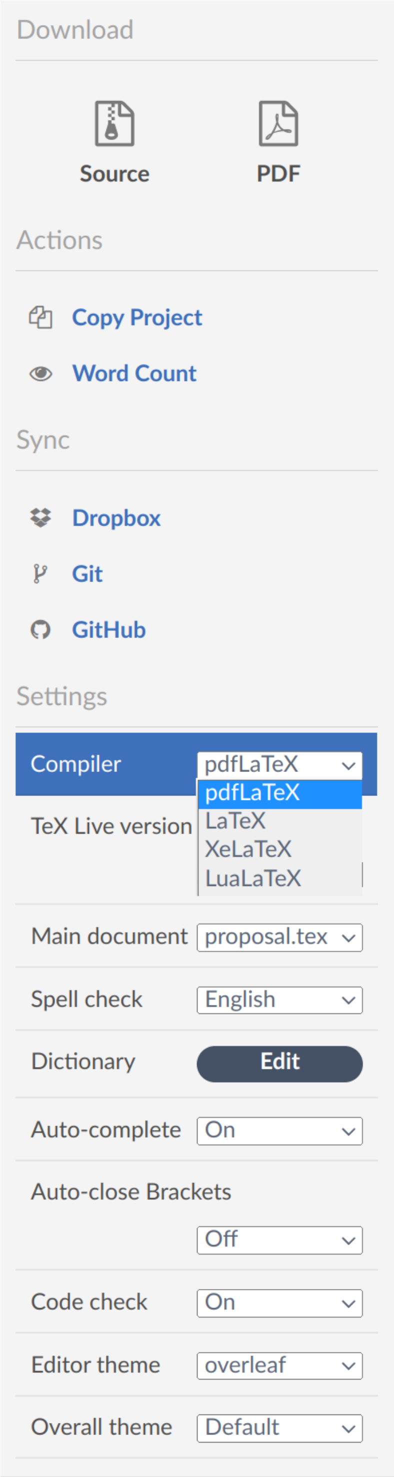
If you have more detailed questions about the template itself - see Appendix E.

D.2 Author configuration of the LaTeX engine

The template should work with pdfLaTeX, XeLaTeX, and LuaLaTeX. If you are using Overleaf, I strongly recommend using XeLaTeX — as this will get the `Arial` fonts correct for the KTH cover. If you are running the compiler on your local machine and you use XeLaTeX **and** you have `Arial` as a system font, then it will be able to use it. Similarly, for LuaLaTeX. For pdfLaTeX I have used \fontfamilyhelvet, *i.e.,* Helvetica, as it is a sans serif font.

One student reported problems with fontspec not loading the fonts properly when running locally with macOS 12.4, TeXLive 2022, LaTeX Workshop on VS Code, and XeLaTeX - the solution is described at `https://tug.org/TUGboat/tb39-2/tb122robertson-fontspec.pdf`.

If you are using Overleaf, it is easy to select the compiler (*i.e.,* TeX engine) by using the drop-down menu, as shown in Figure D.1.



Compiler pdfLaTeX

TeX Live version

Main document proposal.tex

Spell check English

Dictionary

Auto-complete On

Auto-close Brackets Off

Code check On

Editor theme overleaf

Overall theme Default

Figure D.1: Selecting a compiler (*i.e.*, TeX engine) in Overleaf

D.3 Author configuration of the template

The template is designed to handle a thesis written in English or Swedish. You can set the default language to ‘english’ or ‘swedish’ by passing an option to the documentclass. Note that the language option is written in all lowercase letters; for example, to set the document’s language to English:

```
\documentclass[english]{kththesis}
```

To set the document’s language to Swedish (uncomment the following line):

```
\documentclass[swedish]{kththesis}
```

The language option ‘swedish’ sets the conditional `\ifinswedish` to true. Among many other things, this conditional is used to configure the KTH cover and the title page to use the chosen language.

The two most common bibliographic engines are supported, *i.e.*, BibTeX and BibLaTeX. To set the language to English and use the bibliographic engine to BibTeX you would say:

```
\documentclass[english, bibtex]{kththesis}
```

To set the language to Swedish and use the bibliographic engine to BibLaTeX you would say:

```
\documentclass[swedish, biblatex]{kththesis}
```

The above illustrates that you can pass multiple options to the document class separated by commas. Also, note that the options were passed as all lowercase letters.

You can, of course, also modify the formatting of the citations and bibliography. See for example the following code snippet:

```
\ ifbiblatex
   %\usepackage[language=english,bibstyle=authoryear,
      ↪ citestyle=authoryear, maxbibnames=99]{biblatex}
   %\usepackage[style=numeric,sorting=none,backend=biber
      ↪ ]{biblatex}
   \usepackage[bibstyle=authoryear,citestyle=authoryear,
      ↪ maxbibnames=99,language=english]{biblatex}
   % alternatively you might use another style, such as
      ↪ IEEE
   %\usepackage[style=ieee]{biblatex}
   \addbibresource{references.bib}
   %\DeclareLanguageMapping{norsk}{norwegian}
```

```
\else
   % The line(s) below are for BibTeX
   \bibliographystyle{bibstyle/myIEEEtran}
   %\bibliographystyle{apalike}
\fi
```

To optimize for digital output (this changes the color palette) add the option: `digitaloutput`. There are also options for A4 or G6 paper: `a4paper` or `g5paper` (respectively). The is an option for `nomenclature`, to produce and refer to equations as shown in Appendix C . Finally, there are options for a 1st cycle thesis or 2nd cycle thesis: `bachelor` and `master` (respectively); however, these two options are **not** currently used.

One of the first things that the author(s) will want to do is add the working title and subtitle to the thesis. This is done using the \title, \subtitle, \alttitle, and \altsubtitle macros as shown below:

```
\title{This is the title in the language of the thesis}
\subtitle{A subtitle in the language of the thesis}

% give the alternative title - i.e., if the thesis is in
   ↪  English,
% then give a Swedish title
\alttitle{Detta är den svenska översättningen av titeln}
\altsubtitle{Detta är den svenska översättningen av
   ↪ undertiteln}
% alternative, if the thesis is in Swedish, then give an
   ↪  English title
%\alttitle{This is the English translation of the title}
%\altsubtitle{This is the English translation of the
   ↪ subtitle}
```

Setting these values once and then using them in many places reduces the work to change them while at the same time ensuring consistency.

Some additional configuration that the author(s) might do is to set the values of the macros related to the course cycle, course code, date of the thesis, number of credits, degree/exam name, subject area, and if the degree is done external to KTH to set the host information (see the file *custom_configuration.text*). Consider the snippet below for a student admitted to the “Bachelor’s Programme in Information and Communication Technology (TCOMK)” program and enrolled in the degree project course “IA150X Degree Project in Information and Communication Technology, First Cycle 15.0 credits” and working at a company “Företaget AB”:

```
\hostcompany{Företaget AB} % Remove this line if the
    ↪ project was not done at a host company

\date{\today}

\courseCycle{1}
\courseCode{IA150X}
\courseCredits{15.0}

\programcode{TCOMK}
\degreeName{Bachelors degree}
% Note that the subject area for a Bachelor's thesis (
    ↪ Kandidatexamen)
% should be either Technology or Architecture
% If the thesis is in Swedish, these would be: teknik |
    ↪ arkitektur
% -- Note the use of lower case for the Swedish subject
    ↪ area
\subjectArea{Technology}
```

Note that in the above macros you have to give the English or Swedish names in the arguments to \degreeName and \subjectArea - as shown below:

```
\degreeName{Kandidatexamen}
\subjectArea{teknik}
```

For a CDATE student enrolled in the course “DA231X Degree Project in Computer Science and Engineering, Second Cycle 30.0 credits”, the cycle, program, course code, degree, and subject area information would be:

```
\programcode{CDATE}
\courseCycle{2}
\courseCode{DA231X}
\courseCredits{30.0}
\degreeName{Degree of Master of Science in Engineering}
\subjectArea{Computer Science and Engineering}
```

The set of possible values for the English or Swedish names in the arguments to \degreeName are:

```
\degreeName{Higher Education Diploma}
\degreeName{Högskoleexamen}

\degreeName{Bachelors degree}
\degreeName{Kandidatexamen}
```

```
\degreeName{Master of Architecture}
\degreeName{Arkitektexamen}

\degreeName{Degree of Master of Science in Engineering}
\degreeName{Civilingenjörs}

\degreeName{Magister}
\degreeName{Magisterexamen}

\degreeName{Degree of Master of Science}
\degreeName{Masterexamen}

\degreeName{Master of Science in Engineering and Master
   ↪ of Arts in Education degree}
\degreeName{Civilingenjör och lärare examen}

\degreeName{Degree of Master of Science in Secondary
   ↪ Education}
\degreeName{Ämneslärarexamen}

\degreeName{Both}  # Degree Project in the Field of
   ↪ Technology <teknikområde> and the Main Field of
   ↪ Study <huvudområde>
\degreeName{Same}  # The case when the field of
   ↪ technology <teknikområde> and main field of study <
   ↪ huvudområde> are the same.
```

For the last two cases, the code compares the values of subjectArea and secondSubjectArea.

You can find a list of the program codes and school acronyms in the file: `lib/schools_and_programs.ins`.

There are a set of rules about what is to be displayed on the KTH cover. These can be found at https://www.kth.se/social/group/sprakkommitten/page/omrade-for-examensarbete/.

One of the reasons for many of the macros shown above and below is to collect the information that is needed to report the approved thesis in Digitala Vetenskapliga Arkivet (DiVA) and to report the title(s) and grade in Lokalt adb–baserat dokumentationssystem (LADOK).

National subject categories are a **required** field in the DiVA record. These categories follow a definition by SCB (nowadays known as Statistikmyndigheten or in English: Statistics Sweden) and HSV (Högskoleverket - nowadays known as Universitetskanslersämbetet (UK-ämbetet) and Universitets- och högskolerådet (UHR) or in English: Swedish Higher Education Authority and Swedish Council for Higher Education). While these codes refer to research areas, these codes are also used in KTH to indicate the area of the thesis. The guidance that I received from the Linköping University library was that one should try to use 5-digit codes when possible. Some examples of these codes are shown in Table D.1.

`\nationalsubjectcategories{}` comma separated list of national subject category codes - each a 3 or 5 digit code

An example for a thesis in Computer Science and Computer Systems:

```
\nationalsubjectcategories{10201, 10206}
```

You can find the subjects and their codes in:
`https://www.scb.se/contentassets/3a12f556522d4bdc887c4838a37c7ec7/standard-for-svensk-indelning--av-forskningsamnen-2011-uppdaterad-aug-2016.pdf`
and
`https://www.scb.se/contentassets/10054f2ef27c437884e8cde0d38b9cc4/oversattningsnyckel-forskningsamnen.pdf`

Table D.1: Examples of some national subject categories and their codes

Code	Category (in Swedish)	Category (in English)
102	Data- och informationsvetenskap (Datateknik)	Computer and Information Sciences
10201	Datavetenskap (datalogi)	Computer Sciences
10202	Systemvetenskap, informationssystem och informatik (samhällsvetenskaplig inriktning under 50804)	Information Systems (Social aspects to be 50804)
10203	Bioinformatik (beräkningsbiologi) (tillämpningar under 10610)	Bioinformatics (Computational Biology) (applications to be 10610)
10204	Människa-datorinteraktion (interaktionsdesign) (Samhällsvetenskapliga aspekter under 50803)	Human Computer Interaction (Social aspects to be 50803)
10205	Programvaruteknik	Software Engineering
10206	Datorteknik	Computer Engineering
10207	Datorseende och robotik (autonoma system)	Computer Vision and Robotics (Autonomous Systems)
10208	Språkteknologi (språkvetenskaplig databehandling)	Language Technology (Computational Linguistics)
10209	Medieteknik	Media and Communication Technology
10299	Annan data- och informationsvetenskap	Other Computer and Information Science
202	Elektroteknik och elektronik	Electrical Engineering, Electronic Engineering, Information Engineering
20201	Robotteknik och automation	Robotics
20202	Reglerteknik	Control Engineering
20203	Kommunikationssystem	Communication Systems
20204	Telekommunikation	Telecommunications
20205	Signalbehandling	Signal Processing
20206	Datorsystem	Computer Systems
20207	Inbäddad systemteknik	Embedded Systems
20299	Annan elektroteknik och elektronik	Other Electrical Engineering, Electronic Engineering, Information Engineering

D.4 Author macros

It is assumed that there can only be 1 or 2 authors. For many years now 2nd cycle theses are expected to only have one author.

For the author or first author, there are a number of macros defined to store information about the author, so that it can later be used in multiple places – for example, the KTH cover (produced with `\kthcover`), the title page (produced with `\titlepage`, the “For DIVA” section at the end of the thesis (produced with `\divainfo{pg:lastPageofPreface}{pg:lastPageofMainmatter}`), and possibly a JavaScript Object Notation (JSON) file named `fordiva.json` produced as a by product of the `\divainfo`. Note that the actual section name has DiVA set in all caps - which hopefully should not occur in the thesis! If the string DiVA set in all caps, does have to appear, then the section heading should be preceded by four euro signs and followed by four more euro signs (as is done this doucment).

The author-related macros are:

`\authorsLastname{}`	the last name of the author[*]
`\authorsFirstname{}`	the first name of the author
`\email{}`	the KTH e-mail address of the author
`\kthid{}`	the author’s kthid, this generally starts with the string ·“u1” and is a unique identifier for every KTH user.
`\authorsSchool{}`	the value is generally of the form: `\schoolAcronym{EECS}`. The currently supported school acronyms are: ABE, CBH, EECS, ITM, and SCI. These are defined in the file `schools_and_programs.ins`.

If the first author is not in Stockholm, Sweden when the acknowledgements are written, then add that information via the macros described below. This information will be used when generating the acknowledgements signature. The acknowledgements signature is the text at the end of the

[*] Note that the author’s name can include a suffix such as “, Jr.” or “ Jr.”, i.e., the suffix can be separated with a comma or not – as the author prefers to write their name.

acknowledgements and it gives the place where the author(s) is/are when writing the acknowledgements and also gives the date and name(s).

`\authorCity{A City}` specify the city

`\authorCountry{A Country}` specify the country

`\authorCityCountryDate{}` pass into this function the month and year for the acknowledgement. This can be a string such as January 2022 or it can be a LaTeX expression, such as \MONTH\enspace\the\year.

If there is a second author and the place, month, and year are **all** the same, then specify the month and year for only the **first** author:

```
\authorCityCountryDate{\MONTH\enspace\the\year}
```

If there is a second author and the place is different, then say:

```
\authorCityCountryDate{}
```

If there is a second author, the macros are:

`\secondAuthorsLastname{}` the last name of the 2nd author

`\secondAuthorsFirstname{}` the first name of the 2nd author

`\secondemail{}` the KTH e-mail address of the 2nd author

`\secondkthid{}` the 2nd author's kthid

`\secondAuthorsSchool{}` the school of the 2nd author

If the second author is not in the same place as the first author, then add the relevant information using the macros below. This information will be used when generating the acknowledgements signature.

`\secondAuthorCity{A City}` specify the city

`\secondAuthorCountry{A Country}` specify the country

`\secondAuthorCityCountryDate{\MONTH\enspace\the\year}` pass into this function the month and year for the acknowledgement

If the second author is the same place as the first author, then comment out or delete the \secondAuthorCityCountryDate{} as shown below:

```
%\secondAuthorCityCountryDate{}
```

D.5 Starting to write - for the impatient

For those who are impatient and rapidly want to start writing, I suggest you start by configuring the `custom_configuration.tex` file (see Appendix D.3), your working title, and abstract (Appendix D.6.1). After this, a quick way to start writing the text in your document is to go to the table of contents in Overleaf and click on a chapter or section - this will utilize a hyperlink to go to that part of the PDF file. Next, click on the left-going arrow near the top of the border between the LaTeX on the left and the PDF on the right; this will take you (close) to the correct place in the source file where you can start to modify the content and write.

D.6 Starting to write

As you write you will notice ”todo” notes in the template. They follow the following conventions:

```
\generalExpl{Comments/directions/... in English}
\sweExpl{Text på svenska}
\engExpl{English descriptions about formatting}
\sweExpl{warnings}
```

If you do not want to see these notes, you can, of course, redefine the above macros to output nothing. If you do not want to see any notes, then add the option **final** to the \documentclass arguments near the top of the file `examplethesis.tex`.

D.6.1 Working abstract

I generally recommend that every student start by writing a working abstract, this will help you keep your focus. To find where you can start to enter your abstract, look in the `examplethesis.tex` file for the line:

```
\generalExpl{Enter your abstract here!}
```

There is lots of information already in the template to help you with entering text, equations, *etc.,* in your abstract. **NB** Abstracts are supposed to stand by themselves, this means no footnotes, no cross-references, no figures, no tables, *etc.*

I suggest avoiding the use of the defined acronyms in abstracts *i.e.,* spell them out rather than using the glossary commands. This is due to the fact that the `glossaries` package (that is being used to support acronyms) does not directly provide support for multiple languages and because I do not understand how to programmatically create plurals of acronyms in Swedish or other languages. Even in an English abstract, it is desirable to avoid using the glossary commands - as this makes subsequent processing of the abstracts harder - since one has to make sure that the list of acronyms and their definitions are provided to any program that will process this LaTeX source code. For this reason, later versions of this template include the acronyms.tex file after the metadata for DiVA.

D.6.2 Structure of the abstracts and summaries

The basic LaTeX structure for an abstract or summary is shown below (for the case of an English abstract and a Swedish summary *i.e.,* sammanfattning):

```
\begin{abstract}
  \markboth{\abstractname}{}
\begin{scontents}[store-env=lang]
eng
\end{scontents}

\begin{scontents}[store-env=abstracts,print-env=true]
here is where you abstract goes.
\end{scontents}

\subsection*{Keywords}
\begin{scontents}[store-env=keywords,print-env=true]
% If you set the EnglishKeywords earlier, you can
   ↪ retrieve them with:
\InsertKeywords{english}
% If you did not set the EnglishKeywords earlier then
   ↪ simply enter the comma separate keywords here:
%such as: Canvas Learning Management System, Docker
   ↪ containers, Performance tuning
\end{scontents}
\end{abstract}

\cleardoublepage
\babelpolyLangStart{swedish}
\begin{abstract}
    \markboth{\abstractname}{}
\begin{scontents}[store-env=lang]
swe
```

```
\end{scontents}
\begin{scontents}[store-env=abstracts,print-env=true]
Swedish summary goes here
\end{scontents}
\subsection*{Nyckelord}
\begin{scontents}[store-env=keywords,print-env=true]
% SwedishKeywords were set earlier, hence we can use
   ↪ alternative 2
\InsertKeywords{swedish}
\end{scontents}
\end{abstract}
\babelpolyLangStop{swedish}
```

It is important to note that the contents of the `scontents` environment for the abstracts are stored **verbatim**, *i.e.*, the LaTeX is **not** executed. The reason for this is to be able to later have a program that can manipulate the source LaTeX to convert it to HTML for use in announcements, calendar events, and for DiVA. This means that if you write the following:

```
\begin{scontents}[store-env=abstracts,print-env=true]
\input{abstract.txt}
\end{scontents}
```

what will end up in your abstract in the metadata save for DiVA will simply be: `″\inputabstract.tex″` – which means that someone will have to cut and paste your actual abstract to insert it into DiVA.

It is also important to that that the following lines:

```
\begin{scontents}[store-env=lang]
eng
\end{scontents}
```

must be before the `scontents` environment for the abstracts and keywords – as these lines indicate what language the subsequent abstract and keywords are in. The three-character code used for the language is the ISO 639-2 Code – specifically the ”B” (bibliographic) variant of these codes — as these codes are used in the DiVA metadata to tag what language is used.

D.6.3 Abstracts must be able to stand alone

The abstract needs to be able to stand alone; therefore, you **cannot** include citations to your references – as the references are **not** part of the abstract! It is possible (but very rare) to have footnotes as part of the abstract. However, you should be aware that quite often, if the abstract is manually entered in

DiVA, the footnote might not be entered. In this case, unless your full text is available (*e.g.,* via DiVA), a reader might not have an easy way to find out what the footnote says.

D.6.4 Acronyms

You may want to define an acronym to help you with your writing, as this can both reduce the amount of typing and help your reader by providing consistent use of acronyms. The acronyms' definitions can be found in the file *lib/acronyms.tex*. The file contains some examples. I generally try to sort the lines to help find which acronyms I already have defined and keep track of the new one(s) I want to add.

D.6.5 Some predefined macros to help when writing

The file *lib/defines.tex* includes some macros that will help you when writing. This includes \etc, to give you "*etc.,* ", \eg, \ie, and \etal. The file also defines \first, \Second, ... \eighth to give you (i), (ii), (iii), ... $(viii)$. Note that 'Second' is written with an initial capital letter to avoid conflict with the unit 'second' in the `siunitx` package.

D.6.6 Additional abstract(s)

All theses at KTH are **required** to have an abstract in both *English* and *Swedish*. However, in addition to this, many students want to add abstracts in additional languages. The template comes pre-configured with places for abstracts in several other languages. If there is a language that you want to use that is not already supported, there are directions for how to add an additional language. If there are abstracts in languages that you do not want, please delete them or comment them out (see Appendix D.6.7).

D.6.7 Removing and hiding parts that you do not want

It is quite likely that you will find parts of the template that you do not want/need. One way of dealing with this is to delete them, and another way is to comment them out. Personally, I like to comment things out, in case I actually do want to be able to read it in the LaTeX file or uncomment it later. To comment out a portion of the file, simply use the following environment:

```
\begin{comment}
```

```
  **** what you want to comment out ****
\end{comment}
```

For example, if you are not interested in the Swedish language `todo` notes, you can look for lines with “\sweExpl” in them and comment them out (or delete them).

D.6.8 Removing the README_notes

At some point you will no longer want this README information. You can remove it by removing the line \include{README_notes/README_notes} – from the *examplethesis.tex* file. You can then remove the **README_notes** directory.

Unless you are an examiner or an administrator you can delete the file: `README_notes/README_examiner_notes.tex` and delete the include of this file from near the end of the template (*i.e.*, *examplethesis.tex*. You can also delete the directory **README_notes/README_examiner-figures**.

D.7 Copyright or Creative Commons License

It is possible to have several variants of the bookinfo page[*]:

copyright — If you want to have a bookinfo page, include the line saying \bookinfopage.

Creative Commons (CC) — If you want to have a bookinfo page but want to have a Creative Commons license, then include \bookinfopage and use and configure the `doclicense` package as described below.

none — If you do **not** want to have a bookinfo page, comment the line saying \bookinfopage and add a \cleardoublepage.

For background about Creative Commons licenses, see: `https://www.kb.se/samverkan-och-utveckling/oppen-tillgang-och-bibsamkonsortiet/open-access-and-bibsam-consortiu`

[*] When printed double sided, the bookinfo page is the back of the title page.

m/open-access/creative-commons-faq-for-researchers.html and https://kib.ki.se/en/publish-analyse/publish-your-article-open-access/open-licence-your-publication-cc.

Note that the lowercase version of the Creative Commons license has to be used in the modifier, *i.e.*, one of: by, by-nc, by-nd, by-nc-nd, by-sa, by-nc-sa, or zero. For the list of supported licenses, see the documentation for the doclicense package.

Note that if the doclicense package is used, it automatically redefines \bookinfopage to be \bookinfopageCC.

D.7.1 Example configuration to have a CC BY-NC-ND license

```
\usepackage[
   type={CC},
   modifier={by-nc-nd},
   version={4.0},
   hyphenation={RaggedRight},
]{doclicense}
```

Note that the option “hyphenation=RaggedRight” can be used with the configuration of the package to set the license information with a ragged right margin rather that as a filled and justified paragraph.

D.7.2 Example configuration to have a CC BY-NC-ND license with a Euro symbol rather than a Dollar sign

```
\usepackage[
   type={CC},
   modifier={by-nc-nd},
   version={4.0},
   imagemodifier={-eu-88x31}, % to get Euro symbol rather
      ↪  than Dollar sign
   hyphenation={RaggedRight},
]{doclicense}
```

D.7.3 Example configuration to have a CC0 license

```
\usepackage[
   type={CC},
   modifier={zero},
   version={1.0},
]{doclicense}
```

D.8 Use of fonts within the thesis

The choice of fonts is a very individual matter and may be affected by the kind of content that you are trying to write, the language that you are writing in, and what you want to convey to your reader. However, some points to keep in mind are:

- Use fonts with serifs for the body of your thesis, their presence makes it much easier for your reader.
- Use sans serif fonts for headings. This helps your reader distinguish them from the body.
- Be very careful when using fonts that are not widely available[*]. Unless you embed the fonts that you have used, your readers may not see what you want them to see. Ideally, you should embed all fonts – even if you only embed the subset you use.
- Although there are fonts that have a huge number of characters in them, they might not have the characters that you need.
- There are also fonts that, although they have a vast number of characters in them, do not have the math table that LaTeX needs to be able to set mathematical content[†].
- Many fonts are proprietary, thus you need to consider whether you have an appropriate license to use them.

What can you do when the fonts you use are missing characters that you need to use? One solution is to use a font that has the character(s) that you want and then make use of them in the places that you need to.

[*] For example, even though it is widely used. not everyone has the Arial font. Additionally, it is a proprietary font; thus, you need to have an appropriate license to use it.

[†] An example of such a font is Google's Noto font. Even though it includes a vast number of characters, it lacks a math table – although there is an awareness of this missing feature

D.9 One big thesis file or a master file with includes of the parts

While many students split their thesis into multiple files (such as `introduction.tex`, `background.tex`, `method.tex`, `what-you-did.tex`, `results-and-analysis.tex`, `discussion.tex`, `conclusion-and-future-work.tex`) and then include these in their main document (with a series of \includexxxx), my experience is that it actually makes it hard to be consistent in the thesis. For example, you cannot do a simple global replacement when you have decided to introduce a particular acronym. It also makes searching for things difficult, as Overleaf's search function only works on individual files*. Personally, I find it hard to correct LaTeX errors when the file is split in this way - since Overleaf does not make it simple to find the root cause of a problem when the chapters are included in this way. There can be a problem with compiling the project in Overleaf, as Overleaf doe not always handle the separate files as one document (unless you use the functions to tell LaTeX that a file is part of a larger document and identify the parent document). Although I have had some students do this splitting successfully and they liked being able to compile just a part of their report; I've personally had strange errors occur with it - hence I did not use this with the template.

There are some advantages to splitting the document into different parts:

- Overleaf has a limit on the number of changes that it can track - but this limit is per file! [Yes, I have gotten bitten by this when I have put in more changes and comments than the limit and had to stop marking up a manuscript.]

- Additionally, Overleaf has a per file size limit (*i.e.,* how large a file can be) - again this is per file [Yes, I have gotten bitten by this when exporting a Jupyter notebook that produced a LaTeX file larger than 50 MB.]

- This is useful when different students (in a 1st cycle degree project) are writing different parts of the report (in this case the divisions can be even at the section or subsection level).

Similarly, many students like to group their figures along with their chapters, *i.e.,* introducing a folder for each part of the report and placing both

*Note that this is not a problem if you use emacs and a tags file, as this can do searching over the whole set of files and even a tags based query and replace.

the text and the figures relevant to this section into the relevant folder. A similar approach can be used with included code snippets, tables, *etc*.

Ultimately, I think the main issue is the degree to which the separate files are separate and can be worked on as if they were very independent. This generally is true in third-cycle theses, as the chapters tend to be rather independent - typically with one conference/journal paper as the focus of a chapter. However, my experience is that first-cycle theses have very highly interdependent parts, while 2nd cycle theses are split between highly interdependent and highly independent.

However, some might find the question of splitting or not to be a matter of taste or perhaps different ways to approach organizing their writing. So you and your supervisor(s) and examiners might want to discuss what choice is most suitable for your purposes.

Appendix E

README and notes about the template

This document, written by Gerald Q. Maguire Jr, describes the thesis template that I have developed for use at KTH Royal Institute of Technology (KTH) and provides some background about why it is the way that it is. It is important to note that the template is **not prescriptive**, as not every thesis will have all the parts the template shows. However, if there is something that you decide to leave out, you should make a conscious decision to do so, and you should consider the impact this may have on your thesis being approved by the examiner.

Fundamental to the design of the template are several key factors:

- Helping students be successful in their degree project,
- Helping students produce a high-quality thesis, and
- Supporting all of the (relevant) phases of the degree project process.

Several thousand theses are written each year by KTH students. Every approved thesis will be entered into Digitala Vetenskapliga Arkivet (DiVA) (independent of whether the full text is made available via DiVA). Collecting the data necessary for DiVA was a major driving force in the design of the template. This data is useful for many of the phases of the degree project, such as announcing the oral presentation.

This template is **not** designed for use by TIMTM and TMMTM students - as students in these two programs are using a different structure for their reports (there is another template available for them).

This document is a work in progress.

E.1 Introduction

This template evolved (radically) from an earlier thesis template that was widely used at KTH. The direction of this evolution was based on the DOCX template developed over many years for use with students for whom I was the examiner and/or supervisor. The suggested structure and contents of the thesis reflect my experience as an examiner for more than 600 degree projects and the experience I have had as a teacher and examiner for the course *II2202 Research Methodology and Scientific Writing*. The template also reflects my interest as a member of KTH's Language Committee in facilitating the parallel use of English and Swedish at KTH, as well as supporting other languages. The latter aspect reflects my experience with double-degree students, who often need to have at least the abstract of their thesis in their home university's language(s). The thesis template also reflects my experience in entering the metadata for hundreds of theses into DiVA and announcing a very large number of degree project seminars.

Appendix E.4 describes several different groups of users and how the template is relevant to them.

Several major thoughts have influenced the design of this template:

Thought 1 The template should help a student be successful in their degree project and help them produce a high-quality thesis in conjunction with their degree project.

Thought 2 The template should help support all of the (relevant) phases of the degree project process.

Thought 3 Redundant data entry should be minimized to increase consistency.

Thought 4 There are several thousand theses written each year at KTH. Theses are the second most common type of publication at KTH.

Thought 5 Every approved thesis will have at least its metadata entered into DiVA. DiVA features multi-language support for title, subtitle, abstract, and keywords.

E.2 Deliminations

This template is **not** designed for use by TIMTM and Media Management TMMTM students - as students in these two programs are using a different structure for their reports (there is another template available for them).

Additionally, I have been told by one of my colleagues in applied mathematics that theses in this area generally do not follow the Introduction, Methods, Results, and Discussion (IMRAD) structure.

Some parts of the template are conditional based on the value of a switch: `\ifinswedish`. The idea is to easily have a single template that supports theses written in English or Swedish. However, in many places, the conditional has not been used but could be. Examples of this include the Swedish names for chapters and sections. Generally, this information is in a note after the English chapter or section name. More complete implementation of the use of this condition remains as future work.

The template does not fully support the G5 paper format. In particular, the KTH cover (produced with `\kthcover)` and back cover (produced with `\kthbackcover)`) have only been adapted for A4 paper. Support for G5 paper remains as future work.

The handling of the subject area (Swedish: Område för examensarbete) is currently incomplete and remains as future work. Personally, I'm still struggling to understand the rules and how one knows what the correct values are (especially for cases of (i) dual degrees and (ii) combinations of technical subjects and education degrees).

E.3 Structure of the files for the template

Table E.1 shows the structure of the files for the template. These files are generally taken from an existing Overleaf project, a ZIP file, or a github.

One hope is that by automatically extracting information from various sources, this information is more likely to be *correct* and *consistent* (supporting Thought 3). This approach has been used to generate two of the files used for the template. These files are:

1. The file `custom_configuration.tex` contains macros and values for configuring a project. These values are generally expected to be known at the start of the project, *e.g.,* author(s), supervisor(s). examiner, course code for the degree project program code, *etc*. While this file can be manually edited, it was designed to be generated by a program that I have written that extracts most of the data from the Canvas course being used in conjunction with the degree project. One of the goals of using such a program is to extract data from Canvas automatically, the KTH profile Application Programming Interface (API), Kurs- och

programplaneringssystemet (KOPPS), and other sources. The macros for defining this information are described in Sections D.4, E.5.1, and E.5.2 - for authors, supervisors, and examiner (respectively).

2. The file `schools_and_programs.ins` contains the English and Swedish names of schools and programs. A program extracted this information from KOPPS.

We will assume that these files have been generated by someone. Later we will examine who this someone might be for each of these files.

Table E.1: Structure of files for the template

bibstyle	**directory containing files related to the style of the bibliography**	
	`myIEEEtran.bst`	a bibtex style file
figures	**directory containing files for figures**	
lib	**directory containing various library files**	
	`acronyms.tex`	a place to define the acronyms that will or might be used
	`defines.tex`	some generally useful defines
	`includes-after-hyperref.tex`	a special include file for packages that have to be included **after** the hyperref package
	`includes.tex`	a centralized place to include packages that might be useful
	`kthcolors.tex`	defines a number of colors from the KTH palette
	`pdf_related_includes.tex`	includes to be able to add the title and other information to the PDF file
	`schools_and_programs.ins`	English and Swedish names of schools and the programs
`custom_configuration.tex`		macros and values for configuring a project
`examplethesis.tex`		an example of the thesis itself
`kth_logo.png`		the KTH logo for use on the cover
`KTH_ROYAL_INSTITUTE_OF_TECHNOLOGY_logotype.png`		KTH logotype for use on the English language cover
`kththesis.cls`		the kththesis class file
`README_notes.tex`		these notes
`references.bib`		references that may be cited in the thesis

E.4 Expected users and their differences

This template is relevant to several different sets of users:

Users 1 Author or Authors (see Appendix D.1),

Users 2 Those working together with the author(s) during the degree project process (see Appendix E.5),

Users 3 Administrative staff working with the document after it has been approved by the examiner (see Appendix E.6), and

Users 4 The (hopefully) many (human) readers of the final document (see Appendix E.7).

Users 5 The (hopefully) many computers reading the metadata and the full text of the final document (see Appendix E.7.1).

Users 6 Those who are maintaining or updating this template (see Appendix E.7.2).

Each of these different sets of users has different needs and perspectives. The following subsections describe these needs and perspectives.

For information for authors, see Appendix D - located in the file `README_author.tex`.

E.5 Those working in parallel with the authors(s) during the degree project

Those working together with the author(s) during the degree project process include the examiner, supervisor(s), and the opponent(s).

E.5.1 Supervisor

If a degree project is done in industry, there is generally an industrial supervisor in addition to the academic supervisor(s). The template supports up to 3 supervisors (typically an academic supervisor, an industrial supervisor, and sometimes an additional academic or industrial supervisor). The choice of up to three reflects my experience and observation of prior theses in DiVA. Note that there is expected to be at least one supervisor. The supervisors are enumerated as A, B, and C. For each of A, B, and C as appropriate, replace the ”X” in the following macros:

\supervisorXsLastname{}	the last name of the supervisor
\supervisorXsFirstname{}	the first name of the supervisor
\supervisorXsEmail{}	e-mail address of the supervisor

If the supervisor is from within KTH, then add their KTHID, School, and Department info:

\supervisorXsKTHID{}	the supervisor's kthid
\supervisorXsSchool{}	the school of the supervisor
\supervisorXsDepartment{}	the department of the supervisor

If the supervisor is from outside of KTH, then add their organization with:

\supervisorXsOrganization{}	the supervisor's organization

E.5.2 Examiner

I assume that there is only a single examiner for a given thesis*. For this examiner, the relevant macros are:

\examinersLastname{}	the last name of the examiner
\examinersFirstname{}	the first name of the examiner
\examinersEmail{}	e-mail address of the examiner

If the examiner is from within KTH, then add their KTHID, School, and Department info:

\examinersKTHID{}	the examiner's kthid
\examinersSchool{}	the school of the examiner
\examinersDepartment{}	the department of the examiner

*Statistically, there are very few theses with multiple examiners, and this generally occurs for students either in a double degree program or when there are two students in a 1st cycle degree project from different schools, then there might be one examiner for each student. As the case of more than one examiner occurs very infrequently, I have left it for future work. The pseudo-JSON structure is set up to handle multiple examiners, but additional macros would be needed in a similar fashion as used for multiple supervisors, and this metadata would have to be conditionally added where appropriate.

If the examiner is from outside of KTH, then add their organization with:

\examinersOrganization{} the examiner's organization

I assume that someone (such as the examiner) will generate the file: `custom_configuration.tex`. This assumption is based upon the fact that the examiner knows who the student or students are who will be working on a given degree project, who the supervisor or supervisors are, what program the student is in, course code, ... Ideally, this file should be generated automatically by some computer program so that each student or pair of students in a group gets a customized template automatically via the Canvas course. However, currently, the file is generated using a command line program (`create_customized_JSON_file.py`) to generate a JavaScript Object Notation (JSON) file. Subsequently, a separate program (`customize_LaTeX_project.py`) takes this JSON data and creates the appropriate LaTeX commands and inserts this information into the file and then inserts this file into a ZIP file, either replacing or augmenting the `custom_configuration.tex` within this ZIP file (if one exists). There is an option for this second program `-initialize` that causes the program to simply replace the file rather than appending the new information to the end of the file.

The above programs are available from https://github.com/gqmaguirejr/E-learning. The README file for this GitHub contains information about how to run the programs, their options, and gives examples.

E.5.3 Opponent(s) and oral presentation

Unlike the supervisors and examiner, the macros related to the opponent and oral presentation are in the `examplethesis.tex` file. The macro for the opponent(s) is:

\opponentsNames{} the names (in normal name order) of the opponent or opponents

When there are multiple opponents, separate their names with '\&'; for example, A. B. Normal \& A. X. E. Normalè.

For the oral presentation, the following macros are filled in once the examiner has scheduled your oral presentation:

\presentationDateAndTimeISO{} date and time of the presentation is ISO format, for example: 2022-03-15 13:00

`\presentationLanguage{}` three letter abbreviation for the language of the presentation according to three letter ISO 639-2 Code – specifically the ”B” (bibliographic) variant of these codes (note that this is the same language code used in DiVA), generally eng or swe

`\presentationRoom{}` a room name and/or “via Zoom https://kth-se.zoom.us/j/ddddddddddd”

`\presentationAddress{}` location of the room, for example: Isafjordsgatan 22 (Kistagången 16)

`\presentationCity{}` city where the presentation occurs, generally: Stockholm

E.6 Administrative staff

Once a thesis is approved by the examiner we need to add the TRITA number. The TRITA number is assigned by the student affairs office of the school from an annual series of numbers.

E.6.1 What is a TRITA number and why does each approved thesis get assigned one?

TRITA stands for Transactions for the Royal Institute of Technology, with the letter ”A” appended to it. The TRITA definition is the 1971 report, ”Mall för publikationsserier vid Kungl. Tekniska högskolan i Stockholm””, TRITA-LIB-1001, `http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-127656`.

The format for TRITA numbers for degree projects is TRITA-⟨school acronym⟩-EX-YYYY:nnnn, where nnnn is a sequential number starting from 1 each year with the numbers assigned in chronological order to approved theses (”numren delas ut kronologiskt först när examinatorn godkänt arbetet.” - according to one of KTH’s archivists). Note that the list of assigned TRITA numbers is archived each year[*]. The year, YYYY, is based on the year that

[*] It seems that this archiving is done twice a year.

the thesis was *approved*.

The TRITA number value can be set with a macro that takes two arguments: series and year:number as shown below:

```
% for entering the TRITA number for a thesis
\trita{TRITA-EECS-EX}{2022:00}
```

E.6.2 Where does the TRITA number go?

The TRITA number will appear on the back cover of the thesis. It is also stored as part of the metadata entered into DiVA.

E.6.3 What does this mean in practice?

Currently, at EECS the TRITA number is only assigned to the thesis when the examiner has approved the thesis and submitted the PDF of the approved thesis (with cover) to the student affairs office. Of course, this does not make much sense because the back cover is already on the thesis! This means that someone in the student affairs office must either (i) edit the sequential number part of the TRITA number (using some PDF tool) or (ii) they need to make a new back cover and replace the existing back cover. A better solution would be to inform the examiner of the TRITA number and the examiner can see that this number is inserted into the macro shown above and this can enable the number to appear on the back cover and as an added bonus be included in the metadata for DiVA.

Note that it is expected that in 2023, this process will change – thus the assignment of the TRITA number and the application of the back cover would be done by the student affairs office (as only they have the relevant information)[*].

E.6.4 Entering the metadata into DiVA

If a thesis has used this template the “For DIVA” page contains the metadata for DiVA and an administrator can cut and paste this data into DiVA. Alternatively, this metadata can be extracted with a program from the PDF file to produce a JSON file that can subsequently be used to create a MODS file for import into DiVA. The LaTeX compiler can in many cases produce a file called “fordiva.json” that contains the metadata.

[*] Note to maintainers: This means that the back cover can be removed from this template.

The programs that can be used to extract data and to take a JSON file and create a MODS file are available from https://github.com/gqmaguirejr/E-learning.

Note that the import of the MODS file does **not import the collaboration data**, even though this is in the file. This is a limitation of the DiVA import function. Therefore, this information has to be manually entered along with uploading of the PDF file itself.

E.7 (Human) Readers of the thesis

Some theses have very few downloads from DiVA, while some have had hundreds of thousands of downloads. Therefore, you should remember that you have a wide range of human readers of your thesis. The readers include other students looking for information related to their own thesis or because they are interested in the future work that you have suggested to work on for their own degree project. Additionally, researchers who are looking for your results may find your thesis relevant to them. In many cases, companies will look at theses for ideas about what the state of the art is - in several cases, theses have been important as "prior art" and this invalidated patents that had been issued if the patent was submitted after the thesis became public (hence it pays to get theses public as soon as possible). Other human readers are the UKÄ review teams that examine the degree programs offered at KTH. Finally, as KTH is a public agency, it is important that the general public know what is done at KTH[*].

E.7.1 Machines reading the metadata or full text of the thesis

The file `pdf_related_includes.tex` contains LaTeX code that stores the title, author(s), and keyword information into the PDF document in such a way that if you ask for the properties of the PDF file you will get this data. This information makes it easier for machines to get this information from the PDF file.

Additionally, many search engines (such as Google's search engine) mine DiVA for the metadata and if the full text of the thesis is published via DiVA then they also process the full text of the thesis. The result is that search engines can find the content in these theses. This is likely to increase the

[*] This is an important part of the Swedish Offentlighetsprincipen.

probability that someone will download your thesis if they think it is relevant to them – increasing the number of your human readers (see Appendix E.7).

E.7.2 Template author and maintainers

KTH periodically changes the cover design for theses, introduces new programs of study, eliminates programs of study, reorganizes administratively, and faculty move between schools, departments, and divisions. It can be expected that this template will need to evolve with these changes.

For example, if there is a change in schools or programs then there needs to be changes made to the file `schools_and_programs.ins`. While the current file was extracted from KOPPS, the program that does this will need to be replaced because further development of KOPPS has been terminated by KTH's central IT unit which plans to transition all of this information to Lokalt adb–baserat dokumentationssystem (LADOK).

As another example, on 13 December 2021 there was a change in the KTH cover for 1st and 2nd theses, and the cover generator web service was shutdown. The initial draft version of the cover used a proprietary font (TheSans B4 SemiLight and TheSans B6 SemiBold). The version that was publicly introduced uses another proprietary font (Arial) and officially only existed as a DOCX file for a thesis in Swedish. The result is that I had to make my own version in LaTeX to try to emulate the DOCX cover. This lead to a lot of effort, but one can get a reasonable cover with the correct font as described in Appendix D.2.

E.8 While writing

As was noted in Appendix D.1, the thesis template contains lots of examples, notes, and comments. One method to provide additional information is the use of \todo. Several different types of `todo` notes have been used in the thesis. These are described in Appendix E.8.1.

E.8.1 Conventions for todo notes

The example thesis text includes extensive comments, directions, and warnings. These follow the form shown below:

```
\generalExpl{Comments/directions/... in English}
\sweExpl{Text på svenska}
\engExpl{English descriptions about formatting}
```

```
\warningExpl{warning}
```

and appear as:

Each of the above is a macro, so as usual in LaTeX you can redefine it - even defining it to produce nothing! Several previous students have placed these re-definitions in the `custom_configuration.tex` file.

E.8.2 Turning on and off the README_notes

As the various README notes are targeted at different readers, you may or not want to see them. It is very easy to turn them on or off by adding or removing a percent ('%') character before the relevant \begin{comment} and \end{comment} comments around each set of notes.

For example, if you are a student writing a thesis, I suggest turning off everything except for the `README_author.tex` and `README_notes.tex` sets of notes. However, I would suggest keeping the other README files around (at least for a little while) as a source of examples of how to do things. Despite having spent a very large number of hours working on the template and drafts of students' theses, I find some of the README files very helpful as a reminder of how to do things.

E.8.3 Removing the README_notes

At some point you will no longer want this README information. You can remove it by removing the line \include{README_notes/README_notes} – from the `examplethesis.tex` file. If you have removed the other README* files from the **README_notes** directory, you can then remove the **README_notes** directory.

E.8.4 Removing the README_notes

At some point, you will no longer want this README information. You can remove it by removing the line \include{README_notes/README_notes} – from the `examplethesis.tex` file. If you have removed the other README* files from the **README_notes** directory, you can then remove the **README_notes** directory.

E.8.5 Removing unused fonts

This version of the template may also have some font information, in the form of Opentype Font files (with the extension ".otf") and TrueType Font font files (with the extension ".ttf"). If you are not using these fonts (and no longer are using any of the README files), then you can delete these font files.

README acronyms

API	Application Programming Interface
DiVA	Digitala Vetenskapliga Arkivet
IMRAD	Introduction, Methods, Results, and Discussion
JSON	JavaScript Object Notation
KOPPS	Kurs- och programplaneringssystemet
KTH	KTH Royal Institute of Technology
LADOK	Lokalt adb–baserat dokumentationssystem
TIMTM	Interactive Media Technology first
TMMTM	Media Management first

TRITA – EECS-EX 2024:0000
Stockholm, Sweden 2025

www.kth.se

€€€€ For DIVA €€€€

{
”Author1”: { ”Last name”: ”Perrot”,
”First name”: ”Tristan”,
”Local User Id”: ”u1lci1b4”,
”E-mail”: ”tristanp@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
}
},
”Cycle”: ”1”,
”Course code”: ”DA250X”,
”Credits”: ”30.0”,
”Degree1”: {”Educational program”: ”Degree Programme in Computer Science and Engineering”
,”programcode”: ”CDATE”
,”Degree”: ”Masters degree”
,”subjectArea”: ”Computer Science”
},
”Title”: {
”Main title”: ”Advancing Entity Resolution: Creating a Unified Pipeline for Scalable Blocking and Accurate Matching”,
”Subtitle”: ”A Study of Supervised and Unsupervised Methods, Graph-Based Approaches, and Large Language Models”,
”Language”: ”eng” },
”Alternative title”: {
”Main title”: ”Detta är den svenska översättningen av titeln”,
”Subtitle”: ”Detta är den svenska översättningen av undertiteln”,
”Language”: ”swe”
},
”Supervisor1”: { ”Last name”: ”Li”,
”First name”: ”Haibo”,
”Local User Id”: ”u1dclhi4”,
”E-mail”: ”haiboli@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Computer Science” }
},
”Examiner1”: { ”Last name”: ”Hedman”,
”First name”: ”Anders”,
”Local User Id”: ”u1c8661x”,
”E-mail”: ”ahedman@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Computer Science” }
},
”Cooperation”: { ”Partner_name”: ”Wavestone SA”},
"National Subject Categories": "10201, 10208",
”Other information”: {”Year”: ”2025”, ”Number of pages”: ”xxi,87”},
”Copyrightleft”: ”copyright”,
”Series”: { ”Title of series”: ”TRITA – EECS-EX” , ”No. in series”: ”2024:0000” },
”Opponents”: { ”Name”: ”A. B. Normal & A. X. E. Normalè”},
”Presentation”: { ”Date”: ”2022-03-15 13:00”
,”Language”:”eng”
,”Room”: ”via Zoom https://kth-se.zoom.us/j/ddddddddddd”
,”Address”: ”Isafjordsgatan 22 (Kistagången 16)”
,”City”: ”Stockholm” },
”Number of lang instances”: "3",
”Abstract[eng]”: €€€€

```
\generalExpl{Enter your abstract here!}
% Write an abstract that is about 250 and 350 words (1/2 A4-page)  with the following components:
% % key parts of the abstract
% \begin{itemize}
%   \item What is the topic area? (optional) Introduces the subject area for the project.
%   \item Short problem statement
%   \item Why was this problem worth a Bachelor's/'Masters thesis project? (\ie, why is the problem
both significant and of a suitable degree of difficulty for a Bachelor's/'Masters thesis project? Why
has no one else solved it yet?)
%   \item How did you solve the problem? What was your method/insight?
%   \item Results/Conclusions/Consequences/Impact: What are your key
results/\linebreak[4]conclusions? What will others do based on your results? What can be done now
that you have finished - that could not be done before your thesis project was completed?
% \end{itemize}

%
```

€€€€,
”Keywords[eng]”: €€€€
Canvas Learning Management System, Docker containers, Performance tuning €€€€,
”Abstract[swe]”: €€€€

```
\generalExpl{Enter your Swedish abstract or summary here!}
\sweExpl{Alla avhandlingar vid KTH \textbf{måste ha} ett abstrakt på både \textit{engelska} och
\textit{svenska}.\\
```

Om du skriver din avhandling på svenska ska detta göras först (och placera det som det första abstraktet) - och du bör revidera det vid behov.}

\engExpl{If you are writing your thesis in English, you can leave this until the draft version that goes to your opponent for the written opposition. In this way, you can provide the English and Swedish abstract/summary information that can be used in the announcement for your oral presentation.\\If you are writing your thesis in English, then this section can be a summary targeted at a more general reader. However, if you are writing your thesis in Swedish, then the reverse is true - your abstract should be for your target audience, while an English summary can be written targeted at a more general audience.\\This means that the English abstract and Swedish sammnfattning or Swedish abstract and English summary need not be literal translations of each other.}

\warningExpl{Do not use the \textbackslash glspl\{\} command in an abstract that is not in English, as my programs do not know how to generate plurals in other languages. Instead, you will need to spell these terms out or give the proper plural form. In fact, it is a good idea not to use the glossary commands at all in an abstract/summary in a language other than the language used in the \texttt{acronyms.tex file} - since the glossary package does \textbf{not} support use of more than one language.}

\engExpl{The abstract in the language used for the thesis should be the first abstract, while the Summary/Sammanfattning in the other language can follow}

€€€€,
”Keywords[swe]”: €€€€
Canvas Lärplattform, Dockerbehållare, Prestandajustering €€€€,
”Abstract[fre]”: €€€€

Résumé en français.

€€€€,
”Keywords[fre]”: €€€€
5-6 mots-clés €€€€,
}

acronyms.tex

```
%%% Local Variables:
%%% mode: latex
%%% TeX-master: t
%%% End:
% The following command is used with glossaries-extra
\setabbreviationstyle[acronym]{long-short}
% The form of the entries in this file is \newacronym{label}{acronym}{phrase}
%                            or \newacronym[options]{label}{acronym}{phrase}
% see ”User Manual for glossaries.sty” for the details about the options, one example is shown below
% note the specification of the long form plural in the line below
\newacronym[longplural={Debugging Information Entities}]{DIE}{DIE}{Debugging Information Entity}
%
% The following example also uses options
\newacronym[shortplural={OSes}, firstplural={operating systems (OSes)}]{OS}{OS}{operating system}

% note the use of a non-breaking dash in long text for the following acronym
\newacronym{IQL}{IQL}{Independent -QLearning}

% example of putting in a trademark on first expansion
\newacronym[first={NVIDIA OpenSHMEM Library (NVSHMEM\texttrademark)}]{NVSHMEM}{NVSHMEM}{NVIDIA OpenSHMEM
     ↪ Library}

\newacronym{KTH}{KTH}{KTH Royal Institute of Technology}

\newacronym{LAN}{LAN}{Local Area Network}
\newacronym{VM}{VM}{virtual machine}
% note the use of a non-breaking dash in the following acronym
\newacronym{WiFi}{-WiFi}{Wireless Fidelity}

\newacronym{WLAN}{WLAN}{Wireless Local Area Network}
\newacronym{UN}{UN}{United Nations}
\newacronym{SDG}{SDG}{Sustainable Development Goal}

\newacronym{ER}{ER}{Entity Resolution}
\newacronym{EM}{EM}{Entity Matching}
\newacronym{ML}{ML}{Machine Learning}
\newacronym{LLM}{LLM}{Large language model}
\newacronym{POC}{POC}{Proof-of-Concept}
\newacronym{GDPR}{GDPR}{General Data Protection Regulation}
\newacronym{AI}{AI}{Artificial intelligence}
\newacronym{BERT}{BERT}{Bidirectional encoder representations from transformers}
\newacronym{KNN}{k-NN}{k-nearest neighbors}
\newacronym{PCA}{PCA}{Principal component analysis}
\newacronym{NLP}{NLP}{Natural Language Processing}
\newacronym{GPT}{GPT}{Generative pre-trained transformer}
\newacronym{SBERT}{SBERT}{Sentence bidirectional encoder representations from transformers}
\newacronym{EDA}{EDA}{exploratory data analysis}
\newacronym{RNN}{RNN}{Recurrent neural network}
```