Entity Matching for Person Records in Authority Files

A Case Study-Based Comparison of Approaches

Master's Thesis

in Computing in the Humanities
of the Faculty of Information Systems and Applied Computer Sciences
at the Otto-Friedrich-Universität Bamberg

Chair of Media Informatics

Author: Maximilian Hebeis

Supervisor: Prof. Dr. Andreas Henrich

Bamberg 2026

URN: urn:nbn:de:bvb:473-irb-112660x

DOI: https://doi.org/10.20378/irb-112660

Abstract

Authority control is of increasing importance in multiple research fields. To ensure reusability as promoted by the FAIR principles, authority files provide persistent identifiers to named entities such as persons, organisations, or geographical locations. Initially conceived as controlled vocabularies in the library context, in recent years authority files have evolved into broader knowledge bases with additional information on the entity referred to in each authority record. Reconciling authority files which contain records referencing the same entity remains a challenge. This thesis represents a case study into applying learning-based entity matching to person records from two large authority databases, namely the German national Integrated Authority File (GND) and the crowd-sourced open knowledge base Wikidata. A workflow is built to extract the person records from the data dumps of both authority files in the RDF/N-Triples format and store them in MongoDB collections. The person records with existing link attributes to a record from the respective other database are then used as source data to train, validate and test learning-based classifiers on. These classifiers are subsequently compared based on their performance classifying person record pairs correctly as matching (referring to the same person) or non-matching. The two approaches compared are classifiers based on classical machine learning and classifiers based on deep learning, i.e. a multilingual variant of the language model BERT. The results show that learning-based entity-matching architectures achieve high F_1 scores on GND and Wikidata authority data; despite their semantic capabilities, BERT-based approaches don't manage to outperform ML classifiers on the data used in this study.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The data collected, managed and provided access to by the cultural institutions commonly known as *GLAM*[1] institutions is often annotated with a multitude of metadata. This is a consequence of the nature of cultural heritage data: Datasets in the humanities are "collected rather than generated" [Tóth-Czifra, 2020], and thus are semi-structured or unstructured in nature, just like their source material. Sources in the humanities rarely exist isolated from one another – literary works, archival records and artworks might contain explicit or implicit references to related *entities* such as other works, persons, geographical locations or organisations. Because they're not *born digital*, annotating cultural datasets with rich metadata is especially important in order to provide researchers with a representation of the digitised object which is as undistorted as possible [Edmond and Nugent Folan, 2017]. Apart from aiding researchers, adding metadata to cultural heritage data also helps them being more machine-readable: A scanned image of a historical manuscript can benefit from being accompanied with a transcript which highlights relevant entities appearing in the text through embedded metadata, for example.

The shareability of data is of growing interest to different research communities. The FAIR Data Principles, originally published by Wilkinson et al. [2016], expressly address the importance of research (meta-)data being findable, accessible, interoperable and reusable. For data in the humanities, this holds especially true: Tóth-Czifra [2020] argues that providing cultural heritage data with metadata is not just an optional improvement, but that omitting contextual information leads to cultural datasets being next to unusable; the "thick description" is not just a supplementary feature, but an integral part of each dataset stemming from the humanities [Tóth-Czifra, 2020; Edmond and Nugent Folan, 2017]. To be FAIR, the metadata annotations have to be consistent, however: Two datasets containing references to Julius Caesar might still not be found by end users or web crawlers if one refers to the entity "Caius Iulius Caesar", while the other records the referenced entity as "J. Cäsar". To standardise entity references, the concept of **authority control** was conceived: Each record in an **authority file** defines a persistent access point associated with an entity that can be referred to by the metadata of a given dataset. There exists a broad range of authority files, ranging from simple controlled vocabularies to entire knowledge bases with additional informational attributes available for each entity [Fischer et al., 2025].

Because of the multitude of existing small- and large-scale authority files, the problem

[1] Galleries, Libraries, Archives and Museums.

of possible inconsistent or ambiguous entity identifiers still persists: How can datasets be guaranteed to be interoperable, if authority records for some entities only exist in some authority files, or if it isn't clear that some authority records across different authority files refer to the same entity? To solve this problem, reconciling authority databases has been a focus of research into authority control in recent years. Most of the efforts to classify pairs of authority records into matching (referring to the same entity) and non-matching records have required considerable human effort, however. Automated techniques to find records referring to the same entity, known as **entity matching (EM)**[2], exist. While early efforts were rule-based and assigned fixed weights to the comparisons between the records' attributes, modern approaches are increasingly based on supervised learning, leveraging methods from machine learning (ML) and, more recently, deep learning (DL) [Binette and Steorts, 2022]. DL-based entity matchers in particular often make use of fine-tuning pre-trained language models (LM) such as BERT for the entity matching task [Devlin et al., 2019].

Despite the active research into authority control and entity matching, there haven't been efforts into applying modern learning-based entity matchers to authority data, to the best of my knowledge. Therefore, this thesis will investigate how learning-based entity matching on authority records can be implemented and how it performs on said data – the **feasibility** and **effectiveness** of applying learning-based entity matchers will be at the centre of this study. To evaluate the performance of different learning-based approaches, one ML-based entity approach and one DL-based (BERT-based) approach will be compared with each other. In order to not exceed the scope of this thesis, two restrictions will be placed on the study. Firstly, it will be limited to a case study between two authority(-like) databases, namely the **Integrated Authority File (GND)** and **Wikidata**. Secondly, the case study will focus on performing entity matching on authority records referring to natural persons, from here on called **person records**.

In the following, the current state of research in authority control and entity matching will be outlined along with prior interdisciplinary efforts to reconcile authority files, combining both areas of study (chapter 2). Subsequently, the motivation, general approach and objectives will be further explained in chapter 3, along with a preliminary analysis of the RDF authority data dumps to be processed. The implementation of both approaches is then detailed in chapter 4. With a workflow for matching authority data implemented, the performance will afterwards be evaluated through a series of experiments, the setup of which is outlined in chapter 5, while the experimental results along with statistics about the corpus of person records built from both authority files' data dumps can be found in chapter 6. Chapter 7 recapitulates the findings of this thesis and evaluates whether learning-based entity matching could be a viable tool for authority file reconciliation. Potential directions for future research based on the results are also discussed.

[2] Also known under a great number of different terms such as *record linkage*, *entity resolution*, *data matching*, *data linkage* or *instance matching*. In this thesis, the moniker **entity matching** will be used.

Chapter 2

Background

2.1 Authority control and open knowledge bases

As stated in the introduction, data from the Humanities often contains references to named entities. These references might be shared by multiple datasets within one database or across multiple databases. To ensure the consistency and integrity of an annotated record across similar datasets, the unambiguity of the record's metadata references is of particular importance to a dataset's *FAIRness*. The same entities might be referred to under lexically very heterogenous name variants, spellings, and formats: A study by On et al. [2014] couldn't find a single author without at least one name variant (e.g. "Jeffrey D. Ullman" vs. "Jeff Ullman") across three bibliographic digital libraries (DBLP, ACM and CiteSeer), with the majority of variants stemming from abbreviation or omission of parts of the names. Additionally, data points being recorded in different languages (e.g. "München" vs. "Munich") or erroneous content caused by spelling mistakes at data entry (e.g. "Archeologist" vs "Archaeologist") can also be sources of ambiguity. Hence, providing unambiguous identifiers to entity references is valuable when working with named entities; the infrastructure and process doing this is known as **authority control**. At the centre of authority control lie centralised databases called **authority files**, which assign a persistent and unique identifier to (named) entities [Busch and Müller, 2023].

Different standards exist for the structure of traditional authority records, such as Resource Description and Access (RDA) or the German *Regeln für die Schlagwortkatalogisierung* (RSWK); Wiederhold and Reeve [2021] distinguish between five components a traditional authority record should include: An **authorised access point** describing the preferred label for an entity (e.g. "Einstein, Albert"), *variant access points* recording alternative labels (e.g. "Einstein", "Aïnstaïn, Almpert", or "爱因斯坦, 阿尔伯特"), **related access points** being references to other authority records related to the entity (like references to Einstein's birth place Ulm or his second wife Elsa Einstein), **associated attributes** denoting further information on the entity not referencing another authority record (e.g. Einstein's birth and death date) and **source information** documenting information on how the information on the entity was obtained and which decisions were made in its creation. Figure 1 shows an excerpt of the GND authority record describing Albert Einstein[1] as an example.

Authority files thus describe their entities not only through their attributes, but also

[1] `https://d-nb.info/gnd/118529579/`

```
@prefix gndo: <https://d-nb.info/standards/elementset/gnd#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .

<https://d-nb.info/gnd/118529579> a gndo:DifferentiatedPerson;
                                  owl:sameAs <https://dbpedia.org/resource/Albert_Einstein>,
                                             <http://viaf.org/viaf/75121530>,
                                             <http://www.wikidata.org/entity/Q937> .

<https://d-nb.info/gnd/118529579> gndo:preferredNameForThePerson "Einstein, Albert" .

<https://d-nb.info/gnd/118529579> gndo:variantNameForThePerson "Einstein";
                                  gndo:variantNameForThePerson "Aïnstaïn, Almpert";
                                  gndo:variantNameForThePerson "爱因斯坦, 阿尔伯特"@zh-Hans .

<https://d-nb.info/gnd/118529579> gndo:familialRelationship <https://d-nb.info/gnd/118688405> .
<https://d-nb.info/gnd/118529579> gndo:familialRelationship <https://d-nb.info/gnd/116425741> .

<https://d-nb.info/gnd/118529579> gndo:placeOfBirth <https://d-nb.info/gnd/4061529-7>;
                                  gndo:placeOfDeath <https://d-nb.info/gnd/4103300-0> .

<https://d-nb.info/gnd/118529579> gndo:dateOfBirth "1879-03-14"^^xsd:date;
                                  gndo:dateOfDeath "1955-04-18"^^xsd:date .

<https://d-nb.info/gnd/118529579> gndo:professionOrOccupation _:node1immkqk89x575184477 .

_:node1immkqk89x575184477 a rdf:Seq;
                          rdf:_1 <https://d-nb.info/gnd/4045968-8>;
                          rdf:_2 <https://d-nb.info/gnd/4137384-4>;
                          rdf:_3 <https://d-nb.info/gnd/4066567-7> .
```

Attribute	**Value**
class	`DifferentiatedPerson`
sameAs	→ `Albert_Einstein` (DBPedia) → `75121530` (VIAF) → `Q937` (Wikidata)
preferredNameForThePerson	“Einstein, Albert”
variantNameForThePerson	“Einstein” “Aïnstaïn, Almpert” “爱因斯坦, 阿尔伯特”
familialRelationship	Einstein-Marić, Mileva (→ `118688405`) Einstein, Elsa (→ `116425741`)
placeOfBirth	Ulm (→ `4061529-7`)
placeOfDeath	Princeton, NJ (→ `4103300-0`)
dateOfBirth	“1879-03-14”
dateOfDeath	“1955-04-18”
professionOrOccupation	Physiker (→ `4045968-8`) Pazifist (→ `4137384-4`) Wissenschaftler (→ `4066567-7`)

Figure 1: An excerpt of the GND authority record with the ID `118529579` in the RDF/Turtle format and in tabular representation.

through their relationship to other entities contained in the database. Initially, authority control was conceived to improve and facilitate the cataloguing of books and subsequently facilitate the information retrieval process for library patrons. By enriching bibliographic records with identifiers from authority files, their findability, interoperability and reusability is improved, thereby allowing library users to locate resources belonging or related to their area of interest – regardless of naming variants of entities associated with the record [Wiederhold and Reeve, 2021; Sardo and Bianchini, 2022]. Consequently, early authority control systems were mainly concerned with providing controlled vocabularies to single institutions (like libraries). Through the increased linking of entities to other related entities within authority files, authority files have undergone an evolution towards general entity and identity management and can nowadays be understood as providers of **linked data** [Wiederhold and Reeve, 2021]. Originally proposed by Berners-Lee [2006], linked data is defined as structured data which specifies an HTTP **Unique Resource Identifier** (URI) for each data point, with these data points including attributes further describing the entity and references to other data points (links to their URIs). The **Resource Description Framework** (RDF) standard is of special importance to the structure of Linked Data objects: RDF files contain triples of **subject–predicate–object**, semantically describing a relationship (predicate) between two entities (subject and object). Subject, predicate, and object can be of three types: URIs identifying resources, literals consisting of a string and optionally a language tag and/or datatype, and blank nodes which represent resources without a URI and are used for temporary referencing [Cyganiak et al., 2014].

The resulting network of entities and the relationships between them can be represented as a **knowledge graph** [Hawkins, 2022; Fischer et al., 2025]. An example for a knowledge graph representation of an authority record is shown in figure 2. References to other knowledge bases can be represented through either the `owl:sameAs` predicate, denoting that the current entity is the same as the one referenced by the record linked to by the URI contained in the object [Beek et al., 2018] or predicates expressly created to link to certain authority files; `P227` is the Wikidata predicate reserved for pointing to GND record identifiers, for example.[2]

Authority files have evolved from being locally restricted to single institutions as well. While local authority databases still exist, **cooperative authority files** have been established to be able to persistently identify entities across local institutions on a national or international scale [Wiederhold and Reeve, 2021; Sardo and Bianchini, 2022]. Such cooperative files as the German **Integrated Authority File** (*Gemeinsame Normdatei*, **GND**)[3], the French **Autorités** of the *Système universitaire de documentation* (SUDOC)[4] or the American **Library of Congress Name Authority File** (LCNAF)[5] are managed by authoritative cultural heritage organisations (in the case of the GND, the German National Library (DNB)). Adding new records to these cooperative authority files is usually only possible through one of the affiliated institutions [Busch and Müller, 2023].

While this centralised "intellectual curation" [Menzel et al., 2021] guarantees high-quality authority datasets, restricting data entry to institutional actors also restricts the amount and informational depth of the authority records. Apart from institutional cooper-

[2] `https://www.wikidata.org/wiki/Property:P227`

[3] `https://www.dnb.de/DE/Professionell/Standardisierung/GND/gnd_node.html`

[4] `https://www.idref.fr/`

[5] `https://id.loc.gov/authorities/names.html`

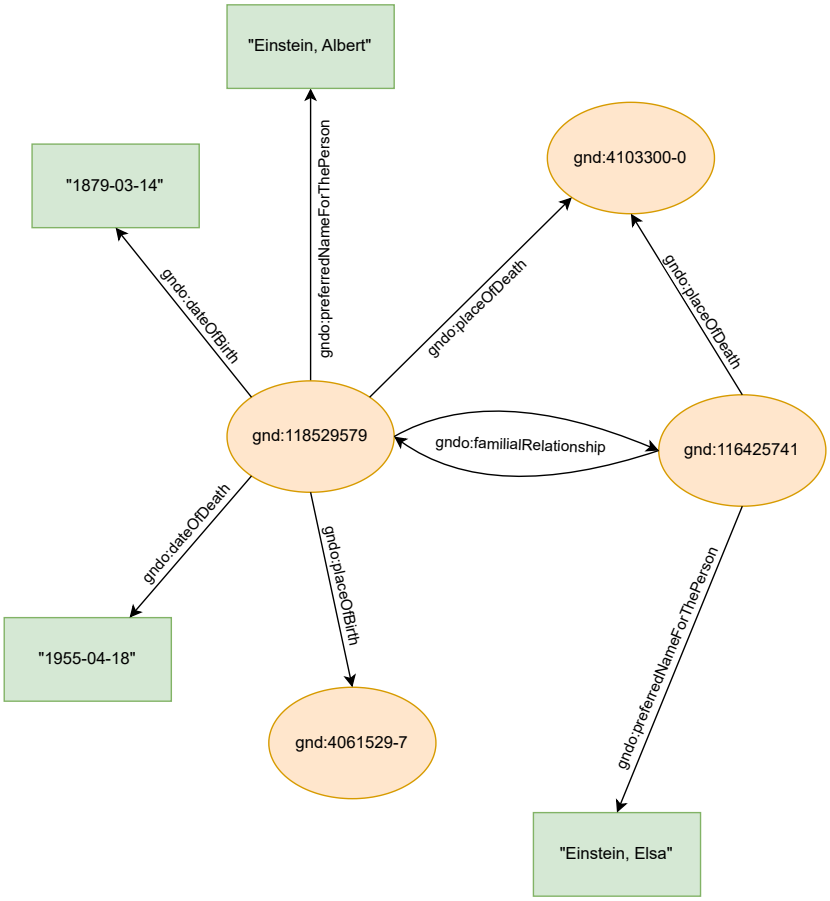


Figure 2: An example knowledge graph based on the GND authority record `118529579` for "Einstein, Albert"

ative authority files, open non-institutional knowledge bases such as **Wikidata**[6], **GeoNames**[7], or **DBPedia**[8] can serve as sources of authority-like data as well, despite them not being authority files in the traditional sense. The Wikidata project in particular is noteworthy here: As a collaborative, *bottom-up* knowledge base, it allows anyone to add new records or modify existing ones. While traditional authority files rely on an underlying fixed ontology, Wikidata allows users to define new data fields for each record as they see fit. This allows Wikidata to grow much quicker than *top-down* authority files and enables contributors to possibly describe entities more precisely than under the constrictions of a fixed data schema. On the other hand, this also means that the quality of the records can vary considerably [Heng et al., 2022; Bianchini et al., 2021]. Nevertheless, because of the openness of Wikidata's crowdsourcing model and its fast growth, there has been growing interest by GLAM institutions into more collaborative data management models: The aim

[6] `https://www.wikidata.org/wiki/Wikidata:Main_Page`
[7] `https://www.geonames.org/`
[8] `https://www.dbpedia.org/`

of the "GND meets Wikibase" project by the German National Library, for example, was to test migrating the GND to Wikibase, the underlying software framework of Wikidata. The project sought to evaluate its suitability for the collaborative creation and maintenance of authority data, assess its usability, and explore if it facilitates reconciling GND records with other authority files and knowledge bases [Fischer and Hartmann, 2019].

Today, authority control is used in non-librarian contexts as well. The benefits of linking their metadata to authority files has been recognised by other GLAM institutions such as archives: As the number of digitally-available archival resources has greatly increased in the last decades, efforts have been made by archives to enrich these digital archival records with authority data and thus to enhance their interoperability and reusability [Hawkins, 2022; Hoinkis, 2023; Koch et al., 2024]. Researchers have recently begun to use authority data to semantically enrich their data, especially in the Digital Humanities. One classic use case is the enrichment of research datasets or entire databases with links to associated authority records [Koch et al., 2024; Thompson and Mukhopadhyay, 2021; Bartalesi et al., 2022]. Digital editions can benefit from linkage to authority files as well by including URIs to authority records in the metadata [Galka, 2023; Menzel et al., 2021]. Authority data is also often used to build specialised search engines by leveraging the high information density of authority records [Yang et al., 2022]. Additionally, open knowledge bases are often used to power domain-specific search engines: Adams [2021] use Wikidata, DBPedia and other gazetteers to provide temporal and spatial information to a historical search interface; Yang et al. [2022] extract textual descriptions of geographical features from Wikidata in order to feed this data to their search engine of hydrographic datasets. Finally, visualisation projects using cultural heritage data often benefit from linkage with authority data and knowledge bases as well. Rantala et al. [2022] use Wikidata to visualise battles from the Finnish Civil War on an interactive map; the *Dehmel Digital*[9] project uses information from the GND to visualise the network of correspondents of German authors Ida and Richard Dehmel. Apart from these few examples, there are a plethora of other interdisciplinary use cases benefitting from semantic enrichment with the help of authority data [Busch and Müller, 2023; Zhao, 2023].

Person records, often making up the largest share of the records contained in authority files, are especially attractive for the use in non-librarian contexts. In contrast to other types of entities, most domains and disciplines agree more or less on the attributes necessary to unambiguously identify persons. Working with person data as the intersection of different research disciplines isn't just an incentive for researchers to work with authority files originating from outside their domain, but also to further contribute data to these files [Fischer et al., 2025].

Although open knowledge bases such as Wikidata aren't authority files in a traditional sense, they serve a similar purpose as "authority-like" databases – providing entities with a unique, persistent identifier. Interlinkage to traditional authority files is encouraged in Wikidata with thousands of properties linking to authority databases [Fagerving, 2023]. Furthermore, institutional authority files have also evolved towards accommodating use cases beyond bibliographic references by incorporating links to other databases' vocabularies; the GND for example continually extended its ontology over its history to include the capability to add linkage to authority files in other languages or specialised in other domains, such as social sciences or economics [Fischer et al., 2025].

[9] `https://dehmel-digital.de/`

2.2 The classical entity-matching workflow

The process of identifying records across multiple datasets referring to the same entity is called **entity matching**. These datasets can either be structured, semi-structured or unstructured, although the traditional entity matching approaches focussed on structured data. Entity matching takes two data sources $D = \{r_1, r_2, ..., r_n\}$ and $D' = \{r'_1, r'_2, ..., r'_m\}$ containing n and m records, respectively. Each record consists of a series of key-value pairs $r = \{(attr_i, val_i)\}$ where $attr_i$ designates the attribute name and val_i its value. The goal of the entity matching process is to find a set $M \subseteq D \times D'$ where each pair (r, r') refers to the same real-world entity [Christen, 2012; Li et al., 2020; Wu et al., 2017]. Early research into entity matching focussed on structured data organised in a tabular manner and often referred to the problem as **record linkage**.

For special cases, separate monikers exist which are sometimes mistakenly applied to the broader task of entity matching as well: **Deduplication** or **(near-)duplicate detection** refer to the process of finding records corresponding to the same entity within a *single* database [Christophides et al., 2021; Papadakis et al., 2020]. Related problems going beyond (semi-)structured data include **coreference resolution** which aims at finding mentions in a full text referencing the same entity, and **entity alignment** which explicitly focusses on knowledge graphs [Barlaug and Gulla, 2021]. The latter, which uses graph embeddings to compare two entries with each other, is also known as **instance matching** [Subagdja et al., 2024].

In classical entity matching (or record linkage), a distinction is made between deterministic and probabilistic linkage. In deterministic linkage architectures, a set of fixed rules is specified which two records have to fulfil in order to be classified as a match (e.g. an exact match on three attributes `family_name`, `occupation` and `birth_place`). Although deterministic linkage is a relatively simple approach, it is still widely used in practice with adequate results when dealing with relatively clean data [Harron et al., 2015; Gross and Mueller-Smith, 2021; Padmanabhan et al., 2019; Schnell, 2022]. However, decision rules in deterministic linkage can prove to be too rigid, especially if there's no attribute uniquely identifying the record's entity (such as a social security number). Probabilistic linkage, first proposed and formalised by Fellegi et al. [1969], uses weights for each comparison between two attributes when considering a potentially matching record pair. The original **Fellegi-Sunter framework** assumed conditional independence between the attributes, using the Bayesian theorem to find a decision rule classifying record pairs according to their probability of being definite matches, possible matches, or definite non-matches. This theoretical framework was later extended to incorporate methods that allow dynamic learning of the feature weights directly from the datasets to be matched or from a subset of those datasets [Binette and Steorts, 2022].

The classical entity-matching workflow consists of **data preprocessing**, **indexing**, **record pair comparison** and **classification**, as depicted in figure 3. Although this workflow originates from the conception of entity matching in the 1960s, most modern EM systems (and the workflow developed as part of this thesis) are still based on an implementation of at least a subset of this process [Christen, 2012]. For this reason, each step shall be briefly explained in the following.

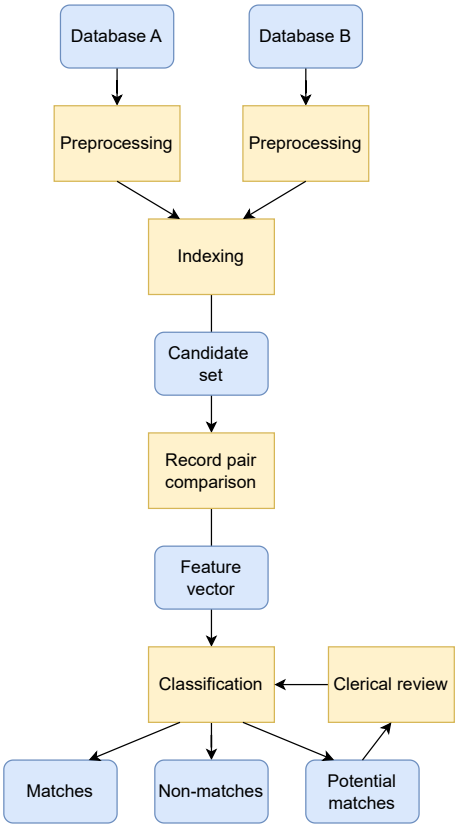


Figure 3: The classical entity-matching workflow, cf. [Christen, 2012]

2.2.1 Preprocessing

The entity-matching process begins with multiple data sources with (possibly) heterogeneous data formatting and differing underlying ontologies and database schemata. **Data preprocessing** is necessary in order to ensure consistent formatting for the following steps. This includes data cleaning, i.e. normalising the encoding of each records' attributes as well as their values; date formats or categorical variables might need to be normalised to make them comparable across data sources [Christen, 2012].

To get semantically related attributes in the source datasets which can later be compared with each other (i.e. `occupation` in one database and `works_as` in another both referring to a person's job), their database schemata have to be aligned. By applying a schema mapping to each record, we arrive at a number of output data fields semantically shared by all source databases; this process is also called **segmentation** or **schema matching** [Christen, 2012; Barlaug and Gulla, 2021]. Segmentation is mostly done manually, leveraging domain knowledge of the underlying databases; (semi-)automatic methods to identify matching attributes exist, however [Barlaug and Gulla, 2021].

2.2.2 Indexing/Blocking

The resulting number of normalised records with shared well-defined output fields can only in rare cases be directly compared with each other. Finding matching records between two databases of size M and N would necessitate $|M| \times |N|$ comparisons; the number of possible comparisons grows quadratically with the combined size of the source datasets. To reduce the search space, candidate pairs which are likely not corresponding to the same entity are preliminarily filtered out through **indexing** or **blocking** [Christen, 2012; Papadakis et al., 2020]. Complexity and granularity of this filtering process depend on the source data. Blocking refers also to a widespread indexing technique, during which every record is being put in a block according to their **blocking key value**. A person record being blocked according to the birth year would thus be inserted in a block or cluster containing only records of persons born in the same year. By then only having to compare records inside each block with each other, the search space can be reduced considerably [Christen, 2012]. For highly structured databases with many categorical variables blocking according to whole values may suffice; to block on attributes which can assume many different values such as personal names, different techniques exist such as tokenising attributes into q-grams and taking these as blocking keys (**Q-Gram Blocking**) or blocking on the suffixes of attribute string values (**Suffix-Array based Indexing**) [Deo et al., 2023; Papadakis et al., 2020]. To account for spelling variations in such attributes as personal names, phonetic algorithms are often used to block strings on their phonetic encodings such as Soundex or Metaphone [Vykhovanets et al., 2020]. Departing from indexing by putting records into blocks, the **sorted neighbourhood approach** sorts merged datasets according to a sorting key, generating sets of record pairs to be compared by use of a sliding window [Papadakis et al., 2020].

Apart from string-based blocking, modern indexing schemes also include signature-based methods, relying on locality-sensitive hashing (LSH) of the whole record [Borthwick et al., 2020]. The **recall** metric (i.e. the proportion of actual matching pairs contained in the possibly matching pairs found during indexing) is of central importance here. A "catch-all" indexing function with large block sizes however inflates the number of comparisons in each block and negates the reduction of the search space. The trade-off between high recall and an as small as possible search space must thus be considered when specifying an indexing procedure [Christen, 2012]. Indexing results in a set of potentially matching record pairs, the indexed **candidate set**.

2.2.3 Record pair comparison

Each pair of records possibly referencing the same entity then has to be compared "in-depth" according to the similarity of the records' attributes. In **record pair comparison**, the similarity of the equivalent output data fields of both records is scored through a comparison function. The easiest method to measure attribute similarity, applied in earlier methods using deterministic linkage, is checking for exact matches between both attributes, their pre- or suffixes, or their phonetic encodings [Christen, 2012]. While guaranteeing high precision, this approach also often causes many missed matches in case of misspellings or missing values [Harron et al., 2015].

Most probabilistic entity matchers calculate the similarity of two data fields using an approximate similarity function, usually yielding a similarity score between 0 (totally dis-

similar) and 1 (exact match). The employed functions differ depending on the fields' data type: For strings, edit-distance-based similarity functions like the **Levenshtein distance**, the **Jaro** and **Jaro-Winkler distance** or the **Smith-Waterman distance** measure similarity according to the number of edits necessary to convert one string into another [Christen, 2012]. Token-based similarity functions split the strings to be compared with each other into sets of tokens (e.g. through whitespace or q-gram tokenisation), and measure the overlap between those sets; the **Jaccard coefficient** and **Dice coefficient** are two examples able to score the similarity of tokenised strings this way [Christen, 2012]. The **Monge-Elkan algorithm** can be seen as a hybrid between edit-distance-based and token-based similarity functions: It compares two sets of tokens A and B using a secondary similarity function, e.g. Jaro-Winkler, to compare each member of the first token set with each member of the second one. The resulting similarity score is the sum of the highest similarity score found for each token in set A with another token from set B. As in indexing, signature-based methods have been used in record pair comparison to encode strings to then measure the distance between the signatures: An example is Wu et al. [2017] using simhash [Charikar, 2002] on document titles and abstracts to then measure the Levenshtein distance between the resulting simhash values.

While these similarity functions mainly operate on a syntactic level, vector space models for **word embedding** such as **Word2Vec**, **GloVe**, or **fastText** capture (at least word-level) semantics as well; the similarity between strings can then be measured by calculating the cosine similarity between two embeddings [Krüger, 2024; Chen et al., 2019]. Chen et al. [2019] were able to achieve higher performance in a classical entity matching workflow by using word embeddings to compare semantically related string attributes as compared to using edit-distance-based similarity functions. It has to be noted, however, that word-embedding algorithm still only capture semantic meaning on the word level; in case of attribute values consisting of sentences or whole paragraphs, large language models like **BERT** are better suited to create contextually enriched embeddings [Jurafsky and Martin, 2025].

For purely numerical attributes, like a person's salaries or savings, the (normalised) absolute difference or percentage difference between two values suffices [Christen, 2012]. Depending on the domain of the databases, there might be a large number of special cases for which custom comparison functions may be necessary. Dates and geographical locations are two such examples: For both a maximal tolerated (day or kilometre) difference might have to be specified to get a normalised similarity score between 0 and 1 [Christen, 2012].

Record pair comparison results in a comparison vector or feature vector of similarity scores for each record pair in the indexed candidate set [Christen, 2012; Konda et al., 2016]. To gain a comprehensive understanding, multiple features are often extracted from each attribute pair, leveraging the strengths of multiple similarity functions; for two output data fields `ltable_occupation` and `rtable_occupation` the Levenshtein distance, Jaccard coefficient, and Monge-Elkan score might be included into the comparison vector as distinct features, for example.

2.2.4 Classification

In the final step, the resulting set of feature vectors must be categorised into matches and non-matches, a process known as **classification**. As mentioned before, most of the developments in EM research in recent years has centred on the classification process:

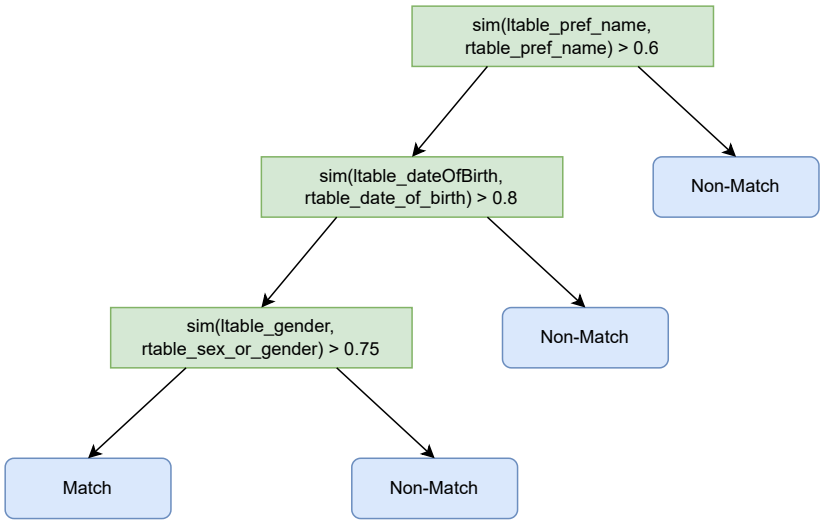


Figure 4: An example decision tree for classifying record pairs based on three features. $sim(a, b)$ denotes a similarity score between two attributes a and b, with the tree classifying the given record pair according to thresholds for each feature learnt during training, cf. [Christen, 2012].

Traditionally, in deterministic linkage, the comparison vectors resulting from record pair comparison would be classified as matches or non-matches according to predefined rules or, if approximate similarity functions were used, according to a predefined threshold applied to the sum of all similarity scores. Early probabilistic EM architectures expanding on the ideas by Fellegi et al. [1969] used the **Expectation-Maximization** algorithm to estimate the probabilities that certain attribute similarity scores belong to matching records; each probability was then taken as a weight for the respective attribute pair (feature). Subsequently, these weights had to be adjusted in clerical reviews until the number of possible matches were minimised [Binette and Steorts, 2022; Gross and Mueller-Smith, 2021].

More recently, supervised learning methods have been leveraged to dynamically learn weights for the comparison vectors. Learning-based classification needs extensive training data to estimate the model parameters. This data is often not available in practice and has to be manually annotated. Nevertheless, machine-learning binary classifiers have been used for the EM classification step since the 1990s because of their superior results on structured data in comparison to other traditional approaches [Binette and Steorts, 2022]. One of the first classifiers used for the entity-matching task were **Naive Bayes** classifiers, which operate under the assumption of conditional independence like the classical Fellegi-Sunter framework and can be seen as an extension of it [Harron et al., 2015; Jurafsky and Martin, 2025]. Conditional independence can impact matching performance, however. In real-life applications, attributes often aren't independent from each other: An attribute containing a person's home town will in most cases co-occur with certain values of an attribute containing the home country (e.g. "Stuttgart" likely co-occurs with "Germany"). Although classification based on conditional independence is a "good enough"

approximation in many cases, other ML classifiers departing from it outperform Naive Bayes [Sayers et al., 2016; Wu et al., 2017; Harron et al., 2015]. Such classifiers include **logistic regression**, **support vector machines** (SVM), and **decision trees**. SVMs learn an optimal hyperplane best separating data points according to their class; by mapping data into a higher-dimensional feature space, non-linear classification can be done as well [Murphy, 2022]. Decision trees are widely used in ML-based entity matching: They partition the feature space recursively, resulting in a tree-shaped model with each node representing a decision rule based on a feature; when passing a feature vector to a decision tree, the vector is evaluated at each node according to the node's decision rule, moving down the tree until a leaf node is reached, which specifies the predicted outcome. Decision trees, or classification and regression trees (CART), are fit to training data by iteratively growing the tree feature-by-feature, making splits at each node based on the feature that best separates the data [Christen, 2012]. A simplified example of a decision tree trained on comparison vectors for classification is shown in figure 4. To reduce the instability of the decision trees, decision tree learning is often combined with ensemble learning techniques: **Bagging** tree models like **random forests** work by fitting a number of decision trees to random subsets of the training data, basing the splits on a random subset of the features at each node. **Boosting** tree models like **XGBoost** take this one step further by iteratively training decision tree models, in each step fitting a new decision tree to the residual errors of the previous step's model, thereby minimising the classification errors [Murphy, 2022].

In general, decision-tree models and SVMs outperform other classifiers in recent applications like linking scholarly bibliographic databases [Wu et al., 2017], administrative data from criminal case defendants [Gross and Mueller-Smith, 2021], genealogical records [Buckles et al., 2023], and cancer registries [Röchner and Rothlauf, 2024]. Schnell [2022] compare the classic Expectation-Maximisation algorithm with deterministic linkage on German student and pupil records and report the highest precision and recall for Expectation-Maximisation. Feigenbaum [2016] develops a custom probit classifier which performs best on matching US census records; however, SVMs and logistic regression don't trail far behind.

The performance of an EM workflow's classification scheme doesn't only depend on the classifier, but also on the domain and structure of the matched data. Datasets containing person data might be heavily reliant on few attributes such as the person's name and birth date, while for other domains, feature importance for classification might be more evenly distributed. Many of the recent studies note this: Wu et al. [2017] explain the Naive Bayes classifier underperforming on their data with the attributes not being conditionally independent. Schnell [2022] register a significantly higher error rate when trying to match records of students with a migratory background, which they assume to be due to the data entry of personal names of foreign origin being more error-prone. How the data is preprocessed naturally affects the overall matching performance as well. Studies into entity matching on certain datasets can thus not only tell us about the effectiveness of different preprocessing, indexing, record pair comparison and classification approaches, but about the datasets in question as well.

2.3 Deep Learning in Entity Matching

As in other fields, methods based on deep learning have dominated the research into entity matching in recent years. Methods originally stemming from natural language processing based on **recurrent neural networks** (RNNs) and **transformers** in particular have been leveraged to realise parts of the traditional entity matching process, or even replace it altogether. Most DL-based entity matching systems do record pair comparison and classification in the same step, for example. Parts of the preprocessing step like segmentation can similarly be included in neuralised entity matching: Barlaug and Gulla [2021] distinguish between neural EM architectures which need to be presented two records with matching schemas, and methods which are schema-agnostic (**attribute-aligned** vs. **non-attribute-aligned** comparison). In non-attribute-aligned architectures, a schema mapping is no longer needed. Datasets with sparse attributes or attributes missing for some records become easier to handle as well, as there is no need for imputing these attributes. Additionally, a distinction is made in terms of how records are embedded: Architectures employing **independent representation** of the entities, where each record is encoded in isolation in a single embedding, are distinguished from those utilising **interdependent representation**, in which both records are jointly processed to produce contextually cross-aware embeddings [Barlaug and Gulla, 2021].

2.3.1 Early DL-based entity matchers

Early DL architectures relied on combinations of recurrent neural networks, convolutional neural networks and deep neural networks to compare entities. *DeepMatcher*, an attribute-aligned architecture, compares two records by feeding each aligned attribute pair to a summarisation block consisting of gated recurrent units and feedforward neural networks. The resulting attribute comparisons are then concatenated and a classification score is found by feeding them to a final feedforward network. The records are thus embedded interdependently, as all attribute pairs of both records are aggregated into one representation [Mudgal et al., 2018]. Kooli et al. [2018] build a non-attribute-aligned EM architecture by feeding the two records wholly into convolutional neural networks, before aggregating them in a pooling layer and getting a classification score through a fully connected layer. This approach is an example for an architecture using independent representation, as both records are embedded by the CNNs independently from each other.

2.3.2 BERT-based entity matchers

Both of these exemplary early DL-based EM architectures use word embeddings to represent attribute values in latent feature space. While this incorporates word-level context into the attribute representation, the semantic meaning of attribute values consisting of sentences or whole paragraphs might not be sufficiently captured. Pre-trained transformer models such as Bidirectional Encoder Representations from Transformers (**BERT**) are the state-of-the-art method to embed longer textual data [Jurafsky and Martin, 2025]. BERT is a bidirectional transformer encoder which takes a sequence of word embeddings as input, along with two special tokens: A `[CLS]` classification token at the beginning of the input and a `[SEP]` token separating sentences. A stack of transformer blocks subsequently enriches the embeddings with context using self-attention. The last transformer

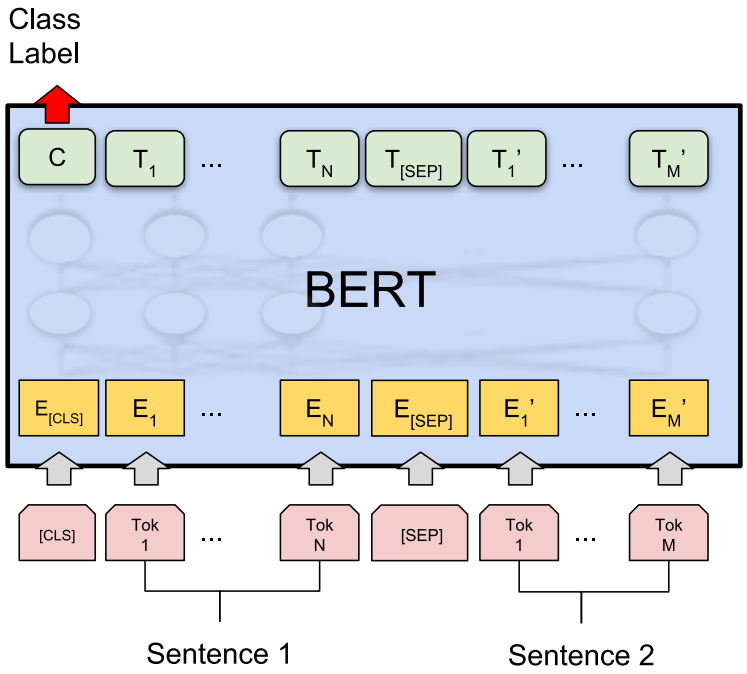


Figure 5: An illustration of a BERT model fine-tuned on classifying sentence pairs [Devlin et al., 2019].

block layer then outputs the contextualised sequence; while each token embedding is now enriched with information about the relationships and dependencies to all other tokens, the output `[CLS]` token serves as a "summary representation" of the entire sentence (see figure 5). The original BERT model was pre-trained on a large corpus from literature and the English-speaking Wikipedia, had an English-only subword vocabulary for word embedding and accepted 512 input tokens at most [Devlin et al., 2019].

Language models like BERT can be used for entity matching: By serialising two records into strings and separating them with the `[SEP]` token, records can be treated like sentences of attribute names and values, which BERT can be trained to embed according to similarity. Paganelli et al. [2024] differ between **pre-trained** and **fine-tuned** EM architectures. Pre-trained BERT entity matchers add two fully connected layers on top of the last layer of the pre-trained BERT model; only these two layers are trained on the entity-matching task. Fine-tuned BERT entity matchers exploit the summarising nature of the `[CLS]` token: A single neural network layer is added on top of the output `[CLS]` embedding and the whole architecture is trained on the EM task, including the underlying BERT model [Paganelli et al., 2024].

Brunner and Stockinger [2020] and Li et al. [2020] first fine-tuned BERT models for entity matching. The concept behind **Ditto**, the architecture by Li et al. [2020], is still considered state-of-the-art for DL-based entity matchers despite of its age [Low et al., 2024; Paganelli et al., 2024]. After serialising the record into a string, Ditto provides functionality to further enrich the data by embedding domain-specific knowledge and to leverage data augmentation strategies to synthesise additional training data from the existing records. As attributes are serialised sentence-pair-wise (as opposed to attribute-pair-wise, which would necessitate a mapping between both records' attributes [Paganelli et al., 2024]),

segmentation can be omitted during preprocessing [Li et al., 2020].

Building on Ditto, most recent research into entity matching is based on feeding fine-tuned BERT models with augmented data: Zhou et al. [2022] use two Sentence-BERT instances for embedding each entity with domain knowledge and then measure the cosine distance between both embeddings. Trabelsi et al. [2022] also embed source data similarly to Ditto, but feed the resulting serializations into multiple domain-specialized BERT models. The resulting ensemble of embeddings is then fed to an attention module for classification. Fang et al. [2023] propose new methods for knowledge augmentation of the source data. Before being fed into the pre-trained language model, the data entries are enriched with semantic information about each attribute label and about the entities referred to in the attribute value; in the latter case, use of Wikidata is made to augment the records with additional information about the mentioned entities. Low et al. [2024] also focus on training classifiers with ensembles; instead of ensembles of language models, ensembles of serialized input strings are built. The different serializations are then sequentially embedded in a modified BERT model with the resulting embeddings then being input to a classification block using self-attention and linear layers. Peeters and Bizer [2021] go beyond data augmentation by modifying the training objective: Instead of solely training the model to output a match probability between two records, they also train it to predict the identifiers of the records. The same authors also experimented with contrastive learning to maximise the difference between embeddings of different entities in product matching [Peeters and Bizer, 2022].

2.4 ML- and BERT-based entity matchers in comparison

As mentioned before, the performance of classical entity matchers depends as much on the data structure and domain of the records to be matched as on the underlying architecture: The accuracy of the found matches and non-matches can vary greatly depending on such factors as the number of the records' attributes, the distribution of distinct attribute values, and the nature of the attributes themselves – whether they are categorical, ordinal, numerical, or textual.

Additionally to pioneering DL-based matching, Mudgal et al. [2018] also introduced the main benchmark datasets used to evaluate EM architectures on in recent years. They distinguish three types of entity matching depending on the structure of the data to be matched (an example for each type is shown in table 1):

Structured EM Records have a tabular structure with a uniform schema, such that each record consists of the same attributes $A_1, ..., A_N$. All records have to be mostly clean, meaning that their attribute values are aligned with the correct attribute label. Attribute values can be textual, but are restricted in length.

Textual EM Compared to structured EM, textual EM records consist of few attributes containing mostly raw, unstructured full-text data.

Dirty EM Records are assumed to have the same schema and structure as records in structured EM, but attribute values might not be aligned to the correct attribute labels. This can happen due to faulty data entry or data corruption.

Name	City	Age
Dave Smith	New York	18
David Smith	New York	18

(a) Structured record.

Name	Brand	Price
Adobe Acrobat 8		299.99
Acrobat 8	Adobe	299.99

(b) Dirty record.

Description
Kingston 133x high-speed 4GB compact flash card ts4gcf133, 21.5 MB per sec data transfer rate, dual-channel support, multi-platform compatibility.
Kingston ts4gcf133 4GB compactflash memory card (133x).

(c) Textual record.

Table 1: Examples for the three types of data encountered in entity matching according to Mudgal et al. [2018].

In general, classical ML classifiers fare especially well in EM applications on **tabular** or **structured data**. Random Forests and boosted tree models in particular are a prevalent approach, as they provide good interpretability, and relative robustness to outliers [Murphy, 2022]. However, real-world data is rarely completely clean and structured, especially when dealing with big sets of scraped data. Textual EM involves the challenge of potentially having to infer the record's attribute values semantically. While there are defined data fields, the record's attributes might be contained in some of the attribute values' full text (e.g. the storage size of the flash drive in the examples in table 1c); they might be completely omitted from other records altogether. Similarly, in dirty EM, an entity matcher might be able to mitigate the data corruption by recognising that a wrongly aligned attribute value really belongs to another attribute label (e.g. in the first example record in table 1b "Adobe" really belongs to the attribute "Brand" instead of the attribute "Name"). The strength of DL-based and BERT-based entity matchers in particular lies in their ability to take the whole record's semantic context into account during the classification process: Language models like BERT are pre-trained on vast corpora of text, thus making them very suitable for extracting information from attributes containing long strings, sentences or whole paragraphs. Additionally, as these architectures don't look at a record's attribute labels and values in isolation from another, but in one serialised "sentence", the embeddings of values wrongly attributed to an attribute label are nonetheless related to all other attributes. Regardless, language models are very sensitive to the structure of their input as well – as mentioned, recent attempts to improve the performance of BERT-based entity matchers mainly centred on serialisation and augmentation of the input [Hegselmann et al., 2023]. Moreover, classical ML architectures are considered to yield comparably good performance on structured data. Compared to EM architectures based on deep learning, training ML classifiers also has significantly lower computational cost [Shwartz-Ziv and Armon, 2022; Grinsztajn et al., 2022].

Type	Name	Domain	# Pairs	# Pos.	# Attr.
Structured	BeerAdvo-RateBeer	beer	450	68	4
	iTunes-Amazon$_1$	music	539	132	8
	Fodors-Zagats	restaurant	946	110	6
	DBLP-ACM$_1$	citation	12,363	2,220	4
	DBLP-Scholar$_1$	citation	28,707	5,347	4
	Amazon-Google	software	11,460	1,167	3
	Walmart-Amazon$_1$	electronics	10,242	962	5
	Clothing$_1$	clothing	247,627	105,608	28
	Electronics$_1$	electronics	249,904	98,401	28
	Home$_1$	home	249,513	111,714	28
	Tools$_1$	tools	249,317	96,836	28
Textual	Abt-Buy	product	9,575	1,028	3
	Company	company	112,632	28,200	1
	Clothing$_2$	clothing	247,627	105,608	3
	Electronics$_2$	electronics	249,904	98,401	3
	Home$_2$	home	249,513	111,714	3
	Tools$_2$	tools	249,317	96,836	3
Dirty	iTunes-Amazon$_2$	music	539	132	8
	DBLP-ACM$_2$	citation	12,363	2,220	4
	DBLP-Scholar$_2$	citation	28,707	5,347	4
	Walmart-Amazon$_2$	electronics	10,242	962	5
	Home$_3$	home	249,513	111,714	3
	Tools$_3$	tools	249,317	96,836	3

Table 2: The *DeepMatcher* benchmark datasets for EM performance evaluation, cf. [Mudgal et al., 2018].

The datasets by Mudgal et al. [2018], from here on referred to as the *DeepMatcher datasets*, are considered to be the reference benchmark for modern entity matching architectures [Paganelli et al., 2024]. As shown in table 2, they're stemming from diverse domains. The DeepMatcher datasets consist of lists of record pairs, with the total number of pairs (*# Pairs*), the number of *positive* pairs referring to the same entity (*# Pos.*) and the number of attributes (*# Attr.*) varying for each dataset. While the methodology of recently introduced EM architectures usually traces back to Ditto [Li et al., 2020], the datasets they're evaluated on usually either originate from the DeepMatcher datasets or the WDC product matching datasets by Primpeli et al. [2019]. **Magellan** [Konda et al., 2016] was used by Mudgal et al. [2018] to compare the performance of classical EM to DL-based architectures; as these scores were likewise used by subsequent studies evaluating on the DeepMatcher datasets, Magellan has become a kind of standard benchmark architecture for classical ML-based entity matching.

Table 3a, 3b and 3c show the performance of different recent BERT-based entity matching architectures on structured, textual and dirty data, respectively. The compared entity matchers include Ditto [Li et al., 2020], JointBERT [Peeters and Bizer, 2021], SupCon [Peeters and Bizer, 2022], REMS [Zhou et al., 2022], DAME [Trabelsi et al., 2022], KAER [Fang et al., 2023] and AttendEM [Low et al., 2024]; Magellan [Konda et al., 2016] serves to compare these results to an ML-based entity matcher. Separate results are shown for Ditto with and without data augmentation and domain knowledge (*baseline* and *full*).

Dataset	**Model F_1 score**								
	Magellan	Ditto (baseline)	Ditto (full)	JointBERT	SupCon	REMS	DAME	KAER	AttendEM
Amazon-Google	49.1	74.1	75.58	–	76.14	65.3	73.53	74.02	**77.67**
Beer	78.8	84.59	94.37	–	–	**96.65**	87.58	–	91.04
DBLP-ACM$_1$	98.4	98.96	98.99	–	–	98.18	98.66	98.43	**99.12**
DBLP-Scholar$_1$	92.3	95.84	95.6	93.99	–	91.74	94.64	**96.11**	95.85
Fodors-Zagats	**100**	98.14	**100**	–	–	**100**	**100**	–	**100**
iTunes-Amazon$_1$	91.2	92.28	97.06	–	–	98.18	95.24	86.21	**98.9**
Walmart-Amazon$_1$	71.9	85.81	86.76	–	–	71.34	82.26	–	**86.81**

(a) Structured EM.

Dataset	**Model F_1 score**								
	Magellan	Ditto (baseline)	Ditto (full)	JointBERT	SupCon	REMS	DAME	KAER	AttendEM
Abt-Buy	43.6	88.85	89.33	83.44	**94.29**	67.4	84.1	88.19	90.11
Company	79.8	41.00	**93.85**	91.40	–	80.73	–	–	92.5

(b) Textual EM.

Dataset	**Model F_1 score**					
	Magellan	Ditto (baseline)	Ditto (full)	REMS	KAER	AttendEM
DBLP-Scholar$_2$	82.5	95.44	95.75	91.76	95.75	**95.89**
DBLP-ACM$_2$	91.9	98.92	99.03	98.19	–	**99.08**
iTunes-Amazon$_2$	46.8	92.92	95.65	94.74	81.82	**96.45**
Walmart-Amazon$_2$	37.4	82.56	85.69	65.74	–	**86.29**

(c) Dirty EM.

Table 3: Reported F_1 scores of recent BERT-based entity matching architectures on different types of data. For each dataset, the best-performing F_1 score is highlighted in **bold**, and the second-best is underlined.

For DAME, the results after training on the full dataset are shown; for KAER, results using entity linking and prompting with a slash ($R + EL + /$) were chosen. All of the comparison results were taken from the respective architectures' papers, except for Magellan, whose results were obtained from Mudgal et al. [2018]. Ditto's results are split into the model with and without data augmentation and domain knowledge injection (*full* and *baseline*, respectively). JointBERT, SupCon and DAME are omitted from table 3c, as they weren't tested on dirty data. Although the results from the original paper on Ditto by Li et al. [2020] are replicated here, it should be noted that multiple authors have reported not being able to reproduce these results [Low et al., 2024].

The general performance advantage of BERT-based methods is clearly visible. Depending on the data, there are marked differences, however: On structured data, the performance gap between Magellan and the other architectures is significantly smaller, with Magellan achieving an optimal F_1 score on the Fodors-Zagats dataset. Some modern entity matchers are even outperformed by Magellan in some cases, like REMS on DBLP-ACM$_1$ and DBLP-Scholar$_1$, and KAER on iTunes-Amazon$_1$. The Amazon-Google dataset is a notable exception here, with Magellan only achieving an F_1 score of 49.1. Mudgal et al. [2018] attribute this to the product title attribute which are often semantically similar but can have large string similarity distances from each other, depending on the string comparison metric (an example would be `corefx three level` and `sos aggregation company corefx three level-core learning ltd.`, in the matching records with the IDs `1203` and `590`[10]). On textual records, BERT-based entity matchers can fully leverage their advantage in semantic comparison over classical EM classifiers. On datasets with particularly informative attributes containing a lot of relevant information with little noise, Magellan performs rather well, however [Mudgal et al., 2018]. Similarly, neural entity matching architectures outperform Magellan on dirty data. The results of the baseline Ditto model and the full Ditto model being not far apart is also worth mentioning: With the exception of the textual Company dataset, the full model only performs slightly better then the baseline, with the baseline's F_1 score even surpassing the full model on the DBLP-Scholar$_1$ dataset.

2.5 Authority file reconciliation and entity matching in authority files

The evolution of authority files from locally-used controlled vocabularies to open knowledge bases makes methods to integrate multiple authority databases necessary. Further "opening up" authority files to wider audiences or complementing records in one authority file with the data from another one is a central motivation behind **authority file reconciliation**. Reconciling authority databases can be understood as a form of entity matching; in the following, the methods employed for matching authority records from different sources and their similarity or difference to classical and DL-based entity matching will be explored.

As mentioned before, traditional authority files are managed by national cultural heritage institutions. The Virtual International Authority File (VIAF) is a **federated authority database** aggregating records from a multitude of national and subnational au-

[10] The dataset in question can be found at `http://pages.cs.wisc.edu/~anhai/data1/deepmatcher_data/Structured/Amazon-Google/amazon_google_exp_data.zip`

thority files, assigning each entity a cluster, consisting of a persistent identifier and links to all corresponding authority records [Wiederhold and Reeve, 2021]. The VIAF originated in 1998 as a joint project by the United States Library of Congress, the German National Library, and OCLC; in 2012 it fully transitioned to being run by OCLC [Putnam, 2022]. The records clustered by the VIAF continue to grow – a clustering algorithm is being run every month on the updated authority file data. Due to OCLC being a private company, the methods employed for clustering are proprietary: Publications by OCLC Research scientists point to the clustering mostly focussing on comparing record labels, with date attributes like date of birth and death also playing an important role for person records. Learning-based methods seem not to be involved in the classification process [Hickey and Toves, 2014; Toves and Hickey, 2014]. With authority files and authority-file-like knowledge bases themselves including links to other authority databases, the boundaries between federated authority databases and authority files become increasingly blurred. Wikidata itself can be seen as a federated authority database as well, with the distinction of working *bottom-up* rather than *top-down* like the VIAF, where new data is only accepted and existing datasets can only be modified by OCLC [Bianchini et al., 2021].

The VIAF represents a transnational effort to reconcile multiple cooperative institutional authority files in a top-down manner. Most authority reconciliation efforts stem from the need to link smaller, often domain-specific databases, to large, well established authority files, however [Heng et al., 2022]. Two principal software solutions have become widely adopted for linking authority records across different databases: Mix'n'Match[11] and OpenRefine[12]. Mix'n'Match allows end users to collaboratively match lists of outside records to Wikidata entries; it divides records into fully matched, preliminarily matched, unmatched, records not on Wikidata, and not applicable records [Magnus, 2013]. Users can import new datasets to be matched against Wikidata entries as CSV or TSV files: To be able to be automatically matched and thus be designated as a preliminary match, these files' header row must include an ID and preferred label for each record; other attributes such as dates of birth and death (for person records) or descriptions can be optionally supplied [Mix'n'Match, 2024]. Mix'n'Match finds probable matches through approximate string matching on the record's labels with Wikidata entries; users with a Wikimedia account can then verify preliminary matches or manually specify a Wikidata entry matching to a given unmatched record [Mix'n'Match, 2024].

Mix'n'Match is designed with Wikidata as the fixed target database in mind. In contrast, OpenRefine offers the functionality to reconcile structured datasets to any bigger knowledge base. The reconciliation process hinges on the latter knowledge base's implementation of its reconciliation service [OpenRefine, 2022]: The manner in which records are matched to each other can range from simple approximate string matching to more complex architectures akin to the classical EM process discussed earlier. The Wikidata reconciliation service[13], for example, aggregates the similarity scores of different types of attributes into a weighted matching score for each record pair; these weights are fixed and not dynamically learned, however [Delpeuch, 2022]. As for Wikidata, there exist a large number of (partially third-party) implementations of reconciliation services for

[11] `https://mix-n-match.toolforge.org/`
[12] `https://openrefine.org/`
[13] `https://wikidata.reconci.link/`

other knowledge bases and classical authority files such as the GND[14] or federated authority files like the VIAF[15]. After automated matching, the end user can then review the matches OpenRefine is confident about and verify or discard lower-scored candidate matches [OpenRefine, 2022].

There exists a multitude of research reports on reconciling authority files or authority-like knowledge bases. Erlinger [2019] seek to reconcile records from the Digital Historical Gazetteer of Saxony (*Historisches Ortsverzeichnis von Sachsen*) with Wikidata, either creating new Wikidata entries or complementing existing ones; for reconciliation, OpenRefine, Mix'n'Match, and a custom Python script crawling Wikipedia pages which already reference gazetteer records are discussed. Myntti et al. [2020] use OpenRefine to reconcile the smaller Western Name Authority File (WNAF), containing person and corporate authority records, with the Library of Congress Name Authority File (LCNAF), to then manually add or update records in the LCNAF with the help of student researchers. The Italian National Library Service (SBN), managing Italy's national authority files, have likewise initiated efforts to link their authority records to Wikidata: Since 2013, SBN records have been linked to Wikidata entries through a dedicated Wikidata property (`P396`) by importing SBN records into Mix'n'Match, making it possible for them to be linked through crowdsourcing [Girolamo, 2024]. Said SBN catalogue was itself a result of integrating multiple authority files into one integrated research system through a manual process of double-checking [Ravelli and Mataloni, 2022]. Such integrated search engines across knowledge bases represent one of the other few use cases within authority control where entity-matching techniques are applied. Jegan et al. [2023] propose an integrated search system for queries on geographical data by downloading authority file dumps and indexing each geographical location according to their coordinates; user queries are then transformed using schema mappings to retrieve relevant records from all indexed authority files. Finally, Mihindukulasooriya [2024] builds a workflow for researchers to add their scholarly articles to Wikidata; this workflow consists of first querying the relevant publications' records from the DBLP[16], before then reconciling the found entities with Wikidata through OpenRefine. The results are then used to create entries for authors and publications which didn't exist in Wikidata before [Mihindukulasooriya, 2024].

[14] `https://lobid.org/gnd/reconcile/`
[15] `https://refine.codefork.com/`
[16] `https://dblp.org/`

Chapter 3

Methodology

3.1 Motivation

Although elements of the entity matching workflow like segmentation, record pair comparison or classification are employed when trying to reconcile multiple authority databases, studies applying the whole workflow to authority data are still missing. In recent years, institutional information curators trend towards linking authority files together and transforming rigid controlled vocabularies into open knowledge bases; nonetheless, recent publications about authority file reconciliation efforts have been mostly limited to praxis reports about how existing software solutions like OpenRefine can be applied to certain databases, often with a lot of manual human effort required. To the best of my knowledge, learning-based classification making use of classical machine learning classifiers or, more recently, deep learning, hasn't been evaluated on authority data at all.

This thesis will therefore evaluate how well learning-based entity-matching architectures perform on authority data: The performance comparison of classical ML-based entity matching and a DL-based (BERT) entity matching architecture on authority data will be of particular interest here. Learning-based classifiers require extensive training data. Modern authority records, which often include links to other authority files via predicates like `owl:sameAs`, can be leveraged by using these links as gold-standard labels to indicate whether two records refer to the same entity.

3.2 Approach: Scope and Limitations

To not exceed the scope of this thesis, the analysis will be focussed on certain facets of the entity-matching workflow and specific domains of authority records. The performance of entity-matching architectures depends both on design decisions made during the implementation of the entity-matching workflow and on external factors like the structure and quality of the records. Regarding the implementation, design decisions of each step of the workflow can result in a gain or loss of accuracy. The data not being properly cleaned during preprocessing, too many matching record pairs being dropped during indexing, or an inadequate schema mapping can affect precision or recall of the entity-matching procedure. The comparison of both entity-matching approaches will therefore focus on the classification step: An entity-matching workflow for authority records will be developed, with ML- and BERT-based classification then being evaluated on the indexed record pairs.

In addition, the comparison will be limited to authority records referring to natural persons, from here on referred to as **person records**, as they constitute a large share of many authority files and typically contain a diverse set of attributes. Restricting the datasets to a particular domain has multiple advantages: Only comparing the approaches on person records ensures that classification models are trained on a homogeneous set of features; a too small random sample set across multiple domains might otherwise cause over- or underfitting on certain domains and distort the evaluation. Furthermore, taking all available domains into account would require a significantly larger schema mapping effort.

Regarding external factors affecting the performance of entity-matching architectures, matching accuracy depends on the structure and quality of the source data. As outlined in chapter 2.4, despite deep-learning based entity matchers generally outperforming classical ML-based entity matching architectures, ML matchers don't fall far behind on discovering links in structured datasets. In order to evaluate different entity-matching architectures on authority records, the structure and integrity of said datasets thus needs to be taken into account as well.

3.3 Datasets

3.3.1 GND and Wikidata

Two authority databases were selected for this case study: The GND, a traditional institutional authority file managed by the German National Library, and Wikidata, a widely-used collaborative open knowledge base. Internal data exchange in the GND uses a customised variant of the MARC 21 data format for bibliographic data. GND authority records provided as linked data follow a fixed and well-defined ontology with fixed classes each entity can belong to and fixed relationships entities can have with each other, mapping concepts from MARC 21 to a linked data context [Haffner, 2024]: GND records referring to individual persons are assigned the type[1] `DifferentiatedPerson`. `DifferentiatedPerson` is a subclass of `Person` and itself has the subclasses `CollectivePseudonym`, `Gods`, `LiteraryOrLegendaryCharacter`, `Pseudonym`, `RoyalOrMemberOfRoyalHouse` and `Spirits` [Haffner, 2024].

Wikidata doesn't follow a fixed ontology – as users can define new entities as they see fit, Wikidata doesn't enforce person records to have a well-defined structure. Entities in Wikidata are also called items, while the entries describing relationships are called properties. Only the items `Q16889133` (*class*) and `Q35120` (*entity*) and the properties `P31` (*instance of*) and `P279` (*subclass of*) are predefined, which "structure the ontology" [Wikidata, 2024]; `P31` serves to designate an entry as having a particular class, with the class being another entity of type `Q16889133`. Wikidata thus considers the items and properties its users define as its underlying ontology [Wikidata, 2024]. While the GND ontology specifies the types of entities and the attributes an entity of each type can have, Wikidata can only recommend entries to follow certain data schemata: Wikidata recommends entries referring to humans to be an instance of `Q5` (*human*), for example. For structuring individual entries, recommendations in the form of user-defined schemata

[1] From here on, *types* and *classes* will be used synonymously for the categories of entities referred to by authority records; the GND uses the former moniker, while Wikidata uses the latter.

exist as well.[2] These recommendations aren't mandatory, however. Additionally, in contrast to traditional authority files like the GND, Wikidata puts no restrictions on the number of classes a Wikidata entry can have: The Wikidata entry for Jesus Christ[3], for example, is simultaneously an instance of `Q178885` (*deity*), `Q20643955` (*human biblical figure*), `Q4271324` (*mythical character*), `Q18563360` (*Quranic character*), `Q5` (*human*), `Q825` (*God in Christianity*), `Q51625` (*Salvator Mundi*), `Q21070568` (*human whose existence is disputed*), and `Q3375731` (*historical character*).

As the focus lies on records referring to individuals with personal information suitable for comparison, the evaluation will be conducted on pairs of records between GND records of type `DifferentiatedPerson` and `RoyalOrMemberOfRoyalHouse` and Wikidata records of class `Q5`; despite `Q5` having a multitude of subclasses, most person records in Wikidata were found to be instances of Q5 in addition to possible subclasses like `Q7569` (*child*) and `Q106155` (*politically exposed person*).[4]

Both the GND and Wikidata provide downloadable dumps of their databases in various file formats. For this thesis, the RDF/N-Triples dumps of both databases were chosen as the input files to extract and preprocess person records from. The N-Triples format was chosen for its simplicity and ease of parsing, as each line in the file represents exactly one RDF triple.

3.3.2 Record Structure

To obtain a preliminary overview of the structure of person records, the files were parsed with the objective to collect statistics on the frequency of each predicate occurring within a person record, with the top 15 predicates for each database shown in table 4. This already gave some valuable insight on the structure of the datasets. Compared to the total number of available attributes, very few attributes are shared by a majority of the records in both databases. Modern authority files and knowledge bases can be thought of to have a graph structure, as outlined in chapter 2: Serialising these datasets into a tabular format the surveyed entity-matching architectures could work with will result in sparse input tables with many attributes possibly being left empty for a majority of the records.

Authority datasets thus represent a **hybrid** form of the data entity-matching methods in recent years were usually evaluated on: While authority records are highly **structured**, their structure is often graph-based rather than tabular, even more so in the case of open knowledge bases with no restrictions on the attributes a type of entity can have. These records will have to be serialised into a tabular format for classical entity matchers to be able to extract feature vectors from them, making it necessary to **impute** similarity scores for attributes which don't exist in one or both of the authority records. BERT-based entity-matching architectures, treating the records like sentences of attribute names and values and might have an advantage here, as imputation of empty attributes isn't necessary here. Tabular serialisations of authority records can thus be understood as a form of **dirty** data as well, as for many attribute labels, attribute values might be missing; moreover, the crowdsourced model of Wikidata makes erroneous attributes or wrongly aligned attribute

[2] cf. `https://www.wikidata.org/wiki/Wikidata:Database_reports/EntitySchema_directory` for a list of Wikidata schemata.

[3] `https://www.wikidata.org/wiki/Q302`

[4]

Predicate	% records
describedby	100
gndIdentifier	100
gender	100
preferredNameForThePerson	100
descriptionLevel	100
modified	100
geographicAreaCode	82.40
professionOrOccupation	60.62
dateOfBirth	51.58
biographicalOrHistoricalInformation	45.18
variantNameEntityForThePerson	42.99
variantNameForThePerson	42.99
oldAuthorityNumber	42.36
sameAs	25.76
placeOfBirth	20.93

(a) GND.

Predicate	% records
about	100
version	100
dateModified	100
label	100
name	100
description	86.42
P21 (*sex or gender*)	80.93
P106 (*occupation*)	70.50
P735 (*given name*)	61.95
P569 (*date of birth*)	55.70
P734 (*family name*)	43.99
P27 (*country of citizenship*)	41.69
altLabel	30.85
P19 (*place of birth*)	29.64
P570 (*date of death*)	27.76

(b) Wikidata.

Table 4: The top 15 most common predicates in the GND's and Wikidata's RDF/N-Triples data dumps, with those predicates representing internal metadata highlighted in *italics*.

values more likely.

Many attribute values of the records in both authority files consist of links to other entities within the same database. For example, the `professionOrOccupation` attribute in Albert Einstein's GND record refers to the GND record with ID `4045968-8` (see figure 1). The preferred labels of the records they refer to (like "Physiker" in this case) as well as most other attribute values can be thought of as textual attribute values of restricted length; a notable exception are the `biographicalOrHistoricalInformation` attribute in the GND and the `description` attribute in Wikidata which often contain longer string literals, sometimes full sentences or paragraphs.

GND and Wikidata authority records thus possess at least some facets of **textual** data as well in this case: Due to their ability to compare features based on their semantic meaning, BERT-based entity matchers might be more effective than matchers with classical ML classification on records with these textual attributes.

3.4 Objective of this thesis

This thesis addresses the research gap in learning-based entity matching on authority data. While reconciling open knowledge bases like Wikidata with traditional authority files such as the GND is a key focus in authority control research, learning-based classifiers have yet to be applied to authority data. Similarly, from an "entity-matching point of view", modern entity matching architectures have as of yet not been evaluated on authority records which present a hybrid structure combining structured, textual, and dirty data.

The objective of this thesis is to delve into this research gap through a first case study by implementing an entity-matching workflow to match authority records from the GND and Wikidata, detailed in chapter 4. Two learning classifiers will then be compared

on their performance in matching records which refer to the same entity: One classifier based on classical machine learning, and one classifier based on BERT models. Existing links to the respective other authority file in the records (`owl:sameAs` in the GND, `P227` (*GND ID*) in Wikidata) will be leveraged as a ground-truth matching label between two records. This will allow the models to be trained on a fixed proportion of matching and non-matching record pairs and enabling an evaluation of the classifiers' performance on an independent test data set without the need for time-intensive manual annotation. For each experimental setting, outlined in chapter 5, both classifiers will be trained, validated and tested on the same dataset. The experimental results will then be presented in chapter 6. In a final step, the results and possible subsequent research questions will be discussed in chapter 7.

Chapter 4

Implementation

In the following, the implementation of the entity-matching workflow on authority data from the GND and Wikidata will be outlined, with the structure of the classical entity-matching workflow introduced in chapter 2 serving as a guide.

Several third-party libraries were used for the implementation of the different steps of the workflow. The entire implementation was realised in Python[1]. To parse the RDF data, code from the rdflib[2] project was adapted. For storing the person records extracted from the data dumps, the document-oriented database MongoDB[3] was chosen; MongoDB was preferred over a relational solution because "raw" authority records don't adhere to a tabular structure. Because of memory constraints, record labels were stored on-disk as well, using SQLite[4] as the underlying database.

For the classification, Magellan [Konda et al., 2016] and Ditto [Li et al., 2020] were chosen as the two architectures to be evaluated. Magellan was selected because it often served as a benchmark model for classical ML entity matching in the past (see chapters 2.2 and 2.4) and because it's entirely implemented in Python. Magellan, consisting of the Python libraries `py_entitymatching`[5], `py_stringsimjoin`[6] and `py_stringmatching`[7] provides a set of tools for each step of the entity-matching process such as indexing functions, string similarity measures for record pair comparison, and a diverse set of ML classifiers [Konda et al., 2016]. Ditto was selected because of its readily available codebase[8] and because, despite of its age, it is still considered state-of-the-art and performs well compared to more recent BERT-based EM architectures (see chapter 2.4).

To get a uniform dataset to compare the classifiers on, the source data will be indexed using the tools available in Magellan before being separately ingested by the two classifiers. As BERT-based entity matching can be considered to perform record pair comparison and classification simultaneously by embedding records in BERT, steps separate from classification aren't necessary here. The resulting workflow, whose implementation will be outlined in the following, is shown in figure 6.

[1] `https://www.python.org/`
[2] `https://github.com/RDFLib/rdflib`
[3] `https://www.mongodb.com/`
[4] `https://sqlite.org/`
[5] `https://github.com/anhaidgroup/py_entitymatching`
[6] `https://github.com/anhaidgroup/py_stringsimjoin`
[7] `https://github.com/anhaidgroup/py_stringmatching`
[8] `https://github.com/megagonlabs/ditto`

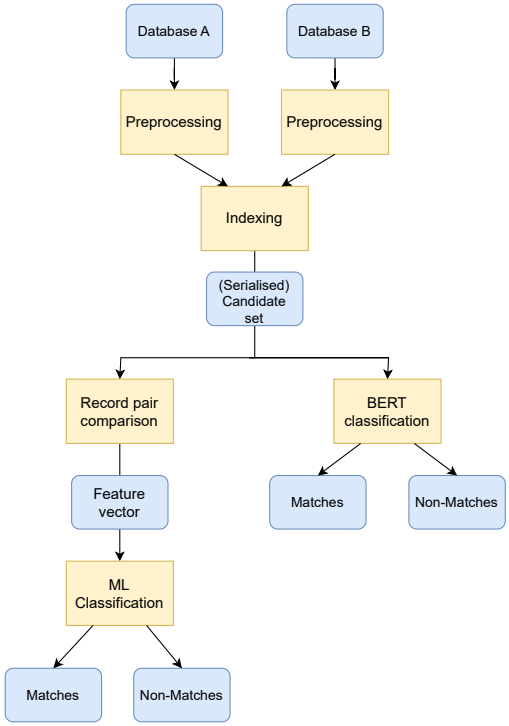


Figure 6: The implemented entity-matching workflow with one ML-based and one BERT-based classifier to be compared according to their performance.

4.1 Preprocessing

4.1.1 Parsing

To perform entity matching on data from both authority databases, the data is preprocessed into a more suitable format. First, the RDF/N-Triples data dumps[9] are parsed, with each person record (either a GND record of type `DifferentiatedPerson`, `RoyalOrMemberOfARoyalHouse` or a Wikidata record which is an instance of `Q5`) being added as a document to a MongoDB collection for each database, with the attribute `_id` containing the record ID. From each triple with a subject URI referring to an authority record an attribute is extracted and added to the corresponding MongoDB document. As predicates are always URIs, their local part is treated as the attribute label: A predicate `https://d-nb.info/standards/elementset/gnd#personalName` results in the attribute label `personalName` and a predicate `http://www.wikidata.org/prop/direct/P108` results in the attribute label `P108`.

The attribute value is then extracted depending on the object's type. Only triples with a subject URI referring to an authority record or a subject blank node identifier

[9]The GND and Wikidata RDF/N-Triples data dumps are available under `https://data.dnb.de/opendata/authorities-gnd_lds_20250313.nt.gz` and `https://dumps.wikimedia.org/wikidatawiki/entities/latest-truthy.nt.gz`, respectively (Accessed: 2025/03/23).

are considered and then parsed according to the type of their object: URIs are added to the collection if they're either pointing to records in the same authority database or to a resource referring to the same entity in the respective other database via `owl:sameAs` in the GND or `P277` in Wikidata. In the latter case, the record ID is extracted from the URI and added to the MongoDB document as attribute `_linked_to`.

RDF literals are stripped of language tag and data type and then added as attribute values, with a special case for preferred labels of person records: They are added as attribute `_pref_name`, with the name's phonetic encoding in **Double Metaphone** [Philips, 2000] added as attribute `_phon_name`. Double Metaphone was selected over other phonetic algorithms because it's not language-dependent and instead accounts for multiple possible pronunciations for each name, making it suitable for finding similar sounding names across different languages [Christen, 2012]. While Wikidata specifies the source language for each literal, literals in the GND dumps mostly don't have language tags; because the GND mainly holds records on persons connected to the German-speaking sphere [Haffner, 2024], (preferred or alternative) record labels are thus mostly in German or neighbouring European languages. The GND Ontology's property names, however, are in English: For this reason, Wikidata literals with the German (`de`), English (`en`) and multilingual (`mul`) tags are added to the collection.

For blank nodes, all associated triples' objects are concatenated and treated like one literal (if all objects are literals) or like a list of attributes (one for each object if objects are URIs as well as literals). After parsing, each person record is stored as a MongoDB document with RDF predicates as the attribute labels and a list of strings extracted from the RDF objects associated with the predicate as the attribute values. At the very least, each person record in the collection now has the attributes `_id`, `_pref_name`, `_phon_name`, and `_linked_to`.

4.1.2 Resolving attributes

During the traversal of the RDF/N-Triples files, each record's (person record or otherwise) preferred labels along with the record ID are saved in a SQLite database (one for the GND and Wikidata records, respectively), with supplementary additional SQLite databases for ISO 3166 geographic codes, ISO 639 language codes, and the GND subject categories; the associated GND properties `geographicAreaCode`, `languageCode` and `gndSubjectCategory` always refer to URIs whose preferred labels aren't in the provided RDF dumps. The second step of preprocessing then consists of iterating through the MongoDB collections to resolve the attribute values and labels (in the case of Wikidata) consisting of record IDs to these records' preferred labels: The key-value pair `P19 : Q494413` is resolved to *place of birth* : *Westmoreland County*, for example. Additionally, the attributes are normalised during this step.

4.1.3 Scrubbing unneccessary attributes and filtering linked records

In a third step, the records are then cleaned of unnecessary attributes. These consist mostly of internal metadata about the record such as `modified` or `deprecatedUri`, or of references to knowledge databases other than the GND or Wikidata. Finally, both MongoDB collections are then traversed once again to filter out records which are mutually linked to one another, with those records being saved in two final MongoDB collections.

GND		**Wikidata**	
Attribute	% records	Attribute	% records
`_pref_name`	100	`_pref_name`	100
`gender`	65.86	`sex_or_gender` (P21)	96.40
`geographicAreaCode`	82.82	`country_of_citizenship` (P27)	55.03
`professionOrOccupation`	80.25	`occupation` (P106)	87.67
`professionOrOccupationAsLiteral`	5.67		
`dateOfBirth`	79.61	`date_of_birth` (P569)	84.36
`biographicalOrHistoricalInformation`	50.46	`description`	80.69
`variantNameForThePerson`	50.42	`altLabel`	42.32
`dateOfDeath`	44.91	`date_of_death` (P570)	50.16
`gndSubjectCategory`	33.97	`field_of_work` (P101)	12.72
`placeOfBirth`	32.80	`place_of_birth` (P19)	53.99
`placeOfActivity`	18.85	`residence` (P551)	1.94
`placeOfDeath`	17.22	`place_of_death` (P20)	29.24
`affiliation`	17.01	`employer` (P108)	17.59
		`member_of` (P463)	8.37
		`member_of_political_party` (P102)	5.32

Table 5: The handcrafted schema mapping. The attribute names for Wikidata are the resolved labels; where applicable, the Wikidata property ID is indicated in parentheses.

These resolved, normalised, cleaned, and mutually linked collections of records can be used to create sets of gold-standard labelled record pairs in the next step. These sets of record pairs will then act as training and testing datasets for classification.

4.1.4 Segmentation

The normalised datasets are then analysed on the proportion of records each attribute label appears in and the number of times each attribute occurred in total. Furthermore, statistics on the average length of each list of attribute values and of each attribute value itself are gathered. Based on these statistics, a schema mapping is handcrafted mapping pairs of semantically related attributes of both databases, with at least one attribute of each pair occurring in more than 15% of total records in the respective database. This limitation was set in order not to bloat the schema mapping: As missing values have to be imputed, the comparison vectors would mostly consist of imputed values if we were to include all attributes, despite most of the attributes only being filled in for a small fraction of the records. The resulting schema mapping is shown in table 5.

On average, Wikidata person records have more of the mapped attributes assigned to them. This is especially apparent in relation to the persons' sex or gender, longer-form descriptions of the person, and places of birth and death: Wikidata records have these attributes filled in substantially more often than GND records. On the other hand, the GND assigns subject categories and countries of origin more frequently to the persons referred to in its records than Wikidata. In general, the importance of preferred personal names is underlined again, as `_pref_name` is the only attribute for both databases to be present in every person record, with variant names additionally available for many of the records.

4.2 Indexing

Subsequently, indexing procedures have to be specified and implemented to get an indexed candidate set from both datasets. Three blockers on personal name, phonetic encodings and birth years and genders were implemented:

Blocking on `_pref_name` On each records pairs' preferred name labels, Q-gram blocking was applied with $q = 4$ and an overlap size of 4: Each name label was tokenised into a list of 4-grams; if two records' name labels share 4 or more 4-grams, the record pair is added to the candidate set.

Blocking on `_phon_name` To account for spelling variations not remedied through the previous blocker, blocking with $q = 4$ on the phonetic encodings of each record pairs' preferred names was additionally applied; if a record shares at least one of its `_phon_name` values with another, the record pair is added to the candidate set.

Blocking on birth years and genders The birth date and gender attributes are used as blocking key values in a third blocker: To account for variations in how birth dates are recorded (if only the birth year of a person is known, the GND usually solely lists the birth date as *YYYY*, while Wikidata sporadically lists it as *YYYY-01-01*), only the birth years are considered. A record pair is added to the candidate set if there's an exact match between birth date and gender of both person records.

4.3 Record Pair Comparison

For classical ML classification, the record pairs in the indexed candidate set have to be converted into comparison vectors to be able to be fed into the classifiers. Except for `_pref_name`, the values of all attributes in the schema mapping consist of sets of strings (i.e. `occupation` : ["Physiker", "Pazifist", "Wissenschaftler"]). The Magellan library provides the functionality to automatically assign similarity functions to attributes based on their data types; however, this automatic feature extraction is only implemented on string-based and numerical attribute values. For `_pref_name`, the feature extraction inbuilt into Magellan's `py_entitymatching` was used: From each attribute having strings consisting of more than one word and less than or equal to five words on average as the attribute value, eight features are extracted by the inbuilt extractor using the Jaccard coefficient, the cosine similarity, the Monge-Elkan algorithm, the Levenshtein distance/similarity, the Needleman-Wunsch measure, and the Smith-Waterman edit distance. For the rest of the attributes, the Magellan code was modified to be able to automatically extract features of sets as well. The similarity functions used for this will be presented in more detail in the following:

Monge-Elkan algorithm The Monge-Elkan algorithm compares two sets of tokens A and B using a secondary similarity function to compare each token pair [Christen, 2012]. The resulting similarity score is calculated as:

$$sim_{monge\_elkan}(A, B) = \frac{1}{|A|} \sum_{i=1}^{|A|} \max_{j=1}^{|B|} sim'(A_i, B_j), \tag{1}$$

where sim' is the secondary similarity function [Christen, 2012]; this implementation uses the Jaro-Winkler distance.

Cosine similarity The cosine similarity measures the angle between two vectors a and b in a high-dimensional space [Jurafsky and Martin, 2025] and is calculated as:

$$sim_{cosine}(a, b) = \frac{a \cdot b}{|a| \cdot |b|} \tag{2}$$

The Magellan library uses a variation of the cosine similarity called the **Ochiai coefficient** [Romesburg, 1984], which operates on the sets of tokens A and B:

$$sim_{cosine\_ochiai}(A, B) = \frac{|A \cap B|}{\sqrt{|A| \cdot |B|}} \tag{3}$$

Dice coefficient The Dice coefficient or Dice score also calculates the similarity between two sets of tokens A and B by considering their overlap while putting an emphasis on shared elements [Christen, 2012]. It is calculated as:

$$sim_{dice}(A, B) = \frac{2 \cdot |A \cap B|}{|A| + |B|} \tag{4}$$

As outlined in chapter 3, some attributes of authority databases can be considered to contain textual data with rich semantic information; in the case of the GND and Wikidata, this holds true for attributes `biographicalOrHistoricalInformation` and `description`. The similarity function selection was thus designed to assign functions depending on the average number of words separated by whitespace the attribute value sets contain. This way, sets consisting of longer full-text strings could be specifically handled. As already mentioned in chapter 2, higher performance in record pair comparison might be achieved by using word embeddings for comparing semantically rich attributes [Chen et al., 2019]. For this reason, two variants of comparing the similarity of textual attributes were implemented to later compare in the evaluation: One variant compares all attributes in a purely "syntactic" way. Another one optionally compares especially long textual attribute values using the cosine distance between both attributes' word embeddings. To embed these attributes, Sentence-BERT (SBERT)[10] was chosen, a BERT model trained specifically for embedding sentences [Reimers and Gurevych, 2019]; to account for Wikidata person record descriptions possibly not being available in German, the multilingual SBERT model `paraphrase-multilingual-MiniLM-L12-v2`[11] was used. The employed similarity functions on attribute value sets depending on the length of the sets' elements are shown in table 6. This results in a pandas `DataFrame` with one row for each record and the extracted features as columns.

4.4 Classification

In a final step, the record pairs are classified as matches or non-matches by either feeding the `DataFrame` of feature vectors extracted through record pair comparison to an ML classifier or by directly handing over the serialised pairs of person records to a BERT model.

[10] `https://sbert.net/`

[11] `https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2`

Similarity function	Tokenisation
sim_{monge_elkan}	Whitespace
sim_{dice}	Q-Gram
sim_{cosine_ochiai}	Whitespace

(a) "Syntactical" variant

Similarity function	Tokenisation
SBERT + sim_{cosine}	–
sim_{dice}	Q-Gram
sim_{cosine_ochiai}	Whitespace

(b) "Semantic" variant

Table 6: The features extracted from attribute value sets. To get the tokens similarity values are calculated for, attribute value sets are either converted into a bag of words (splitting each element at whitespace) or into a bag of q-grams (tokenising each element with $q = 3$).

4.4.1 ML classification

Before classification, the feature `DataFrame` must first be imputed, as the feature vectors may contain `NaN` values resulting from comparisons between attributes that one or both records in a pair lack; most machine-learning models require numerical input and cannot handle missing values. Omitting record pairs with missing attributes is not feasible, as the vast majority of records contain missing data. Instead, `NaN` values are imputed using the mean similarity score of that feature across all other available record pairs. This approach ensures that features involving missing attributes are treated in a neutral, non-biased manner during classification.

The imputed `DataFrame` of feature vectors is split into training, validation, and test sets. Various classifiers are trained on the training set and validated on the validation set, with the best-performing classifier (in terms of the classifier's F_1 score on the validation set) subsequently being evaluated on the test set. The classifiers considered include decision tree (DT), random forest (RF), XGBoost, support vector machine (SVM), Naive Bayes (NB), and logistic regression (LR). The default parameters for these matchers, implemented in Magellan as wrappers around `sklearn` classifiers, weren't altered.

4.4.2 BERT classification

BERT-based entity matchers directly take the record pair as input after serialising both records as strings. As mentioned in chapter 2, BERT models have a fixed number of input tokens they accept; the original BERT model accepts 512 input tokens at most [Devlin et al., 2019]. For authority records with particularly many attributes, this makes summarisation necessary in order to not exceed the maximal number of input tokens. Ditto uses TF-IDF-based summarisation, condensing record pair serialisations into the most frequent words in the sequence. While effective for textual records with few attributes and long raw-text attribute values, this approach struggles with person records: As the records have many attributes with only a few of them being textual, unique attribute values possibly helpful for classification tend to be omitted. Additionally, as Ditto's serialisation module doesn't differentiate between attribute labels and values, attribute labels are often truncated. To accommodate the specific characteristics of person records, a simple custom summarisation method was implemented. This method iterates over each record's attributes, selects the first value from each attribute's value set, and greedily appends it to the serialised record string, provided that the maximum input token limit is not exceeded.

Ditto provides functions to augment the serialised data before embedding it with a BERT model: Domain knowledge can be injected into the attribute values by leveraging a named-entity recognition (NER) model; Li et al. [2020] use spaCy's `en_core_web_lg` model[12] specifically trained on English-language source data. Additionally, Ditto can augment training data by randomly deleting attributes, shuffling the order attributes appear in, or swapping values between attributes. The original source data is then interpolated with the augmented examples using the MixDA technique and the model is trained on the resulting interpolation, making it more robust against overfitting on certain attributes [Li et al., 2020].

This case study is concerned with comparing classical ML classifiers with BERT-based classifiers on a conceptual level. Augmenting the data or injecting extra domain knowledge could distort the comparison, as the datasets used as input to the ML-based approach are completely unaltered. Therefore, only "baseline" Ditto without data augmentation and domain knowledge injection was chosen as the BERT classifier implementation. Li et al. [2020] achieved the best performance using the RoBERTa language model [Liu et al., 2019]. The `roberta-base` model[13] was specifically trained on five English-language source corpora, however; as the GND and especially the Wikidata authority records can contain attribute values in multiple languages (mainly German and English), `distilbert-base-multilingual-cased`[14], a multilingual BERT variant, was used as the language model in this implementation. For the training process, a lower batch size and number of epochs was used compared to Ditto's default hyperparameters, as the BERT models proved to be prone to overfit much quicker than expected. The used hyperparameters are shown in table 7.

Hyperparameter	**Value**
Maximal sequence length	256
Learning rate	3e-5
Batch size	32
# of epochs	5

Table 7: The hyperparameters used for fine-tuning the `distilbert-base-multilingual-cased` classifier in Ditto.

[12] `https://spacy.io/models/en#en_core_web_lg`
[13] `https://huggingface.co/FacebookAI/roberta-base`
[14] `https://huggingface.co/distilbert/distilbert-base-multilingual-cased`

Chapter 5

Experimental setup

5.1 Focusses of the experiments

To compare the performance of ML-based and BERT-based entity matchers on authority data, multiple model variants will be trained and evaluated on the preprocessed GND and Wikidata authority records. This chapter will outline the experimental settings; the experimental results will then be evaluated and discussed in the following two sections. To assess which approach is best suited to reconcile authority files, the evaluation and discussion of the experimental results will be conducted across five key areas of focus:

Matching performance First and foremost, the performance of the two entity-matching approaches naturally must be the focus of the evaluation. Three performance metrics widespread in EM literature are **precision**, **recall**, and the **F-measure** or **$\mathbf{F_1}$ score** [Christen, 2012]. **Precision** is used as a metric for the quality of found matches. It is defined as the fraction between the number of matches correctly designated by the classifier (**true positives**) and the total number of designated matches (**false positives**):

$$precision = TP/(TP + FP). \tag{5}$$

Recall informs about the proportion of total existing matches found, calculated as:

$$recall = \frac{TP}{TP + FN}. \tag{6}$$

Finally, the **$\mathbf{F_1}$ score** is the harmonic mean between precision and recall, defined as:

$$f_1 = 2 \cdot \frac{prec \cdot rec}{prec + rec}. \tag{7}$$

Despite it not affecting the matching performance comparison between both approaches (as the same indexed candidate sets will be used to train, validate and test both approaches), indexing quality will also be measured. Recall and precision are known in this context as **pair completeness** and **pair quality**, respectively.

Importance of name attributes As evidenced by the traditional structure of authority records which assign special importance to the authorised access point containing the preferred label for an entity (see chapter 2.1) and the statistics collected on the frequency of certain predicates in records from the GND and Wikidata (see table 4), personal names are of special importance for disambiguating person records. This can possibly cause name attributes to have a far greater influence on the outcome of the classification step in comparison to other attributes; in this case, the overfitting of the model on personal names is a danger that has to be considered. For each approach, two variant models will be trained and evaluated to estimate the importance of person record attributes containing personal names in both authority databases: One including name attributes such as `_pref_name`, `variantNameForThePerson` and `altLabel`, and one excluding them. For ML classifiers, `sklearn` provides a function to evaluate feature importances through randomly permuting a feature column of the input dataset. The decrease of the estimator's performance (measured using a given metric such as the F_1 score) on the shuffled dataset is then compared with its performance on the unshuffled one, with a high performance degradation signifying an important feature for the classification process. This is repeated for each feature; the resulting feature importances can be plotted in a graph.[1] Highly correlated or collinear features can distort the measured feature importances, however: In this case, one feature column being randomly shuffled doesn't affect the estimator's performance much, as the model can get the required information from another correlated feature. This makes clustering correlated features necessary to get meaningful performance degradations for each permutation on the dataset.[2]

BERT models are a lot more opaque to analyse: An entire research field named **BERTology** is dedicated to investigating the inner workings of BERT architectures. Paganelli et al. [2024] undertook an in-depth study into BERT-based entity matchers. As part of this study, feature importances are inspected as well by analysing the attention giving by the `[CLS]` token towards the tokens for each attribute and performing gradient analysis for each attribute [Paganelli et al., 2024]. As Paganelli et al. [2024] only do this on the tabular structured and dirty DeepMatcher datasets, however, they can assign a fixed position to each attribute in the input string. Because of the graph-like nature of authority data, authority records can consist of very different sets of attributes, with the position of few attributes in the serialised string guaranteed. Additionally, BERT learns contextual relationship across all attributes on a sentence-level, and not just between pairs of attributes, making a measure of feature importance through randomly shuffling them less meaningful.

As multiple features are extracted from each attribute pair in the schema mapping, multiple features are highly correlated (e.g. the Monge-Elkan score, cosine distance and Dice score for `professionOrOccupation` and `occupation`). Measuring the permutation importance for authority record attributes would entail either shuffling the attribute columns before feature extraction or clustering the features, requiring manual effort. As mentioned, feature importances can't be measured in a robust manner for the BERT-based approaches either. For this reason, the importance of

[1] `https://scikit-learn.org/stable/modules/permutation_importance.html`

[2] `https://scikit-learn.org/stable/auto_examples/inspection/plot_permutation_importance_multicollinear.html#sphx-glr-auto-examples-inspection-plot-permutation-importance-multicollinear-py`

name attributes and other features will be observed only indirectly by analysing the entity-matching approaches' performance on the datasets with name attributes in- and excluded.

Amount of necessary human effort As has been described in chapter 2.4, one of the advantages of BERT models for entity matching is that in principle, less preprocessing is required (applying a handcrafted schema mapping to the records is optional) and no features need to be identified and extracted prior to feeding the data to the classifier. In contrast, a considerable amount of preliminary data analysis was necessary in the implementation of this thesis project to convert the "raw" indexed candidate set into a format that the ML classifiers can handle. This applies particularly to the schema mapping, which in theory limits the dimensions along which the machine-learning approach can compare records with each other: BERT-based EMs don't need the input data in a tabular format and can include very rarely occurring attributes in the record pair comparison and classification steps, which were omitted from the schema mapping. To gauge the importance of rare attributes not in the schema mapping, the BERT approach will be evaluated with and without the schema mapping applied.

Amount of training data Somewhat related, classical machine-learning classifiers generally need less training data than language models like BERT [Barlaug and Gulla, 2021]. Consequentially, this means that less labelled ground-truth examples of matching authority record pairs are needed for ML classifiers to fit the data; in the case of authority databases which don't have records with annotated links (or at least not as many as in the case of the GND and Wikidata), less time-consuming manual labelling would be required. Therefore, multiple subsets of the full cross-linked datasets of different sizes will be used for training and validating both approaches to assess how much labelled data is sufficient for training a robust classifier.

Difference in training time Finally, the time taken for training is also a factor not to be neglected when assessing the suitability of both approaches. ML classification needs to extract feature vectors through record pair comparison before classification, while BERT only needs to serialise the indexed candidate set; to compare training times, the time taken for record pair comparison on the training set will be added to the time elapsed training the ML classifiers, while for the BERT classifier the time taken for serialisation will be counted towards training time.

In addition to these five key focuses, the ML classifier is also trained and evaluated using the alternative feature configuration for long descriptive attributes outlined in chapter 4.3. Following Chen et al. [2019], in doing so it shall be assessed if a "hybrid" approach between syntactical and semantic comparison and classification yields better results compared to pure syntactical feature extraction while not needing to fine-tune an entire language model like the "purely semantic" BERT-based approach.

In summary, four variants of each approach are respectively trained and evaluated: ML classification is tested with and without name attributes included in the source data, and with either purely syntactic record pair comparison or hybrid feature extraction partially making use of SBERT. This results in four models, referred to as `ml_full`,

`ml_nonames`, `ml_sbert_full`, and `ml_sbert_nonames` from this point forward. Experiments with BERT-based classification are conducted with and without name attributes as well, and with and without a schema mapping applied, resulting in four models designated as `bert_full_map`, `bert_full_nomap`, `bert_nonames_map`, and `bert_nonames_nomap`.

5.2 Training, validation and testing data

These models are trained and validated on progressively larger subsets of an indexed candidate set built from the GND and Wikidata. Cursory analysis of the preprocessed authority data in MongoDB revealed a problem, however: The cross-linked person records' `_pref_name` attributes proved to be too similar in most cases. In fact, out of 1,623,141 linked record pairs in the GND and Wikidata, 1,319,198 have the exact same preferred name. This constitutes a possible trade-off in using linked authority records as ground-truth data: Authority records which have been manually annotated with links to other authority files have a high probability of having already been enriched with data from the linked records. Hence, in our case, a Wikidata entries' preferred name attribute might have been supplemented from the corresponding GND record or vice versa. While linked authority records being strongly aligned after reconciliation ensures consistency across authority files, it increases the risk of learning classifiers overfitting on the `_pref_name` attribute even more. With `_pref_name` exactly matching for a great majority of the records, overfitting is all but guaranteed.

To be able to train more robust classifiers and to assess the performance of learning-based entity matchers on a more realistic diversity in attributes, a synthetic dataset was built from the full set of linked authority records: From the linked GND and Wikidata records, all record pairs with a Monge-Elkan score of less than or equal to 0.75 were extracted, ensuring some disparity between the `_pref_name` attributes. This resulted in two MongoDB collections `diff_gnd_75` and `diff_wd_75`, including 62,268 GND and Wikidata records respectively, from which the indexed candidate set used for training, validating and testing the models could then be sourced.

This happened in multiple steps, visualised in figure 7: First, two samples were taken from `diff_gnd_75` and `diff_wd_75` containing 6,000 records respectively with 25% of the records in each sample matching with a record from the respective other sample. This ratio was chosen to approximately reflect the real-life match ratio in the datasets, ensuring a balanced yet realistic sample composition. These samples were then indexed, resulting in an indexed candidate set. This was done twice, once for the test set, using the resulting indexed candidate set directly, and once for the training and validation sets which were further subdivided: The training/validation candidate set was downsampled into 10 subsets containing between 1,000 and 10,000 record pairs in increments of 1,000. Each of these subsets was sampled to maintain a 10% match and 90% non-match ratio. This high imbalance reflects the very high imbalance of the source candidate set (see the indexing results in chapter 6) and thus prevents the models from being biased towards positive examples. However, the match ratio was kept slightly higher than in the original candidate set to ensure sufficient positive pairs for model training. The subsets were then each split into training and validation sets with a ratio of 4:1.

Each of the eight model variants discussed above was trained on each of the training/validation subsets and evaluated on the test set. For the ML-based approach, this

entailed training each of the classifiers listed in chapter 4.4.1 and evaluating them on the validation set; the classifier with the best F_1 score for each subset was then selected and evaluated on the test set. Using the BERT-based approach, each model is trained using the hyperparameters detailed in chapter 4.4.2 and evaluated on the validation set after each epoch. Compared to the approach by Li et al. [2020], which used 10, 15, or 40 epochs depending on dataset size, a smaller number of epochs (5) was chosen. This adjustment was made because loss converged more quickly than anticipated on the authority data, and a reduced number of epochs helped prevent overfitting.

The results of indexing `diff_gnd_75` and `diff_wd_75`, along with the outcomes of training, validating and testing for the eight model variants will be presented in the following chapter.

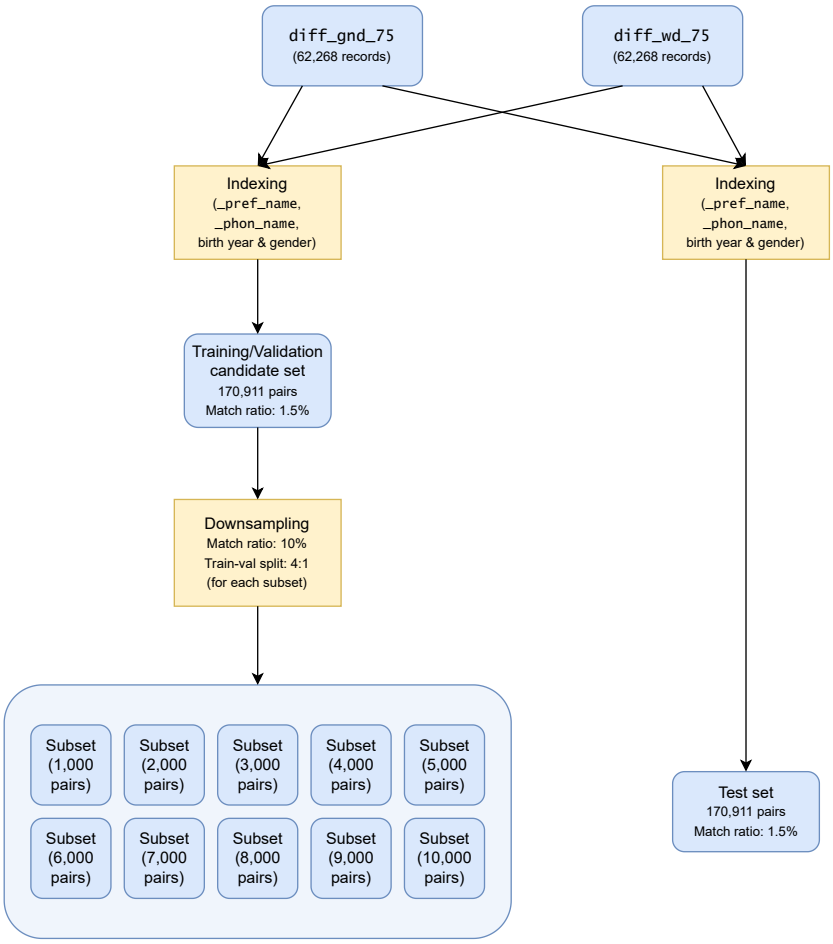


Figure 7: The workflow for constructing the datasets used to train, validate, and test the classifiers.

Chapter 6

Evaluation

6.1 Evaluation of indexing

Two samples of the databases `diff_gnd_75` and `diff_wd_75` are indexed to obtain indexed candidate sets for training and validation and for testing. Out of the 1,250 positive pairs contained in both sample sets of 6,000 total pairs, the implemented blockers find 1,135 and 1,134 respectively, resulting in a moderately high pair completeness (recall), very high reduction ratio, and low pair quality (precision). Modifying the parameters for indexing (such as the q-value or overlap size for indexing on `_pref_name`) resulted in higher pair completeness, but lower pair quality and vice versa; the current settings were chosen to ensure comparatively high recall, a precision not too far off from the target match ratio for the training/validation sets (10%), and a high reduction ratio.

	Candidate set	
	Train/Val	Test
# pairs (matching)	63,901 (1,125)	64,833 (1,132)
Reduction ratio	99.82	99.82
Pair completeness	75	75.47
Pair quality	1.76	1.75

Table 8: Indexing statistics for training/validation and test candidate sets.

The contribution of the three individual blockers to the combined candidate set is markedly heterogeneous in indexing quality: Indexing `diff_gnd_75` and `diff_wd_75` for the test set reveals highly disparate performances of the three blocking procedures for example, detailed in table 8. Blocking on `_phon_name` achieves a very high reduction ratio and results in pairs with high pair quality compared to candidate sets resulting from other blockers. The vast majority of truly matching record pairs contained in the combined candidate set are found by blocking on `_pref_name`, however. While this indexing procedure "casts a wider net" compared to the others, resulting in lower pair quality, the importance of the preferred name attribute is further underlined by the amount of unique true matches each blocker contributes, visualized in figure 8: This reveals an even greater disparity between the blockers' performances, with blocking on `_phon_name` and birth year and gender only accounting for 22 and 6 uniquely found matches which are not also found by blocking on `_pref_name`. While the two other blockers thus provide

complementary coverage, blocking on the `_pref_name` attribute is the primary driver of recall during indexing.

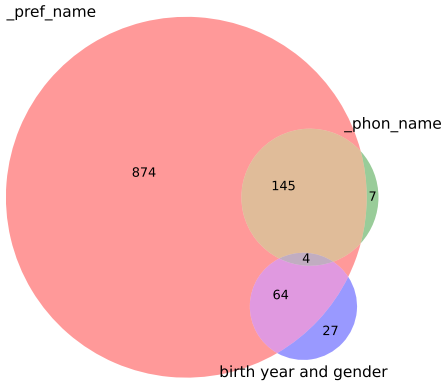


Figure 8: Contribution of the true-positive matches found by each blocker to the total number of matching candidate pairs in the indexed set.

6.2 Evaluation of training and validation

During each training run, each of the ML classifiers included in the comparison (DT, RF, XGB, SVM, NB, and LR) were trained and validated on a given subset of the training/validation candidate set. For the ML-based models, the best-performing ML classifiers along with their F_1 scores on the respective validation sets are listed in table 9, while the F_1 scores achieved by the BERT-based models on each validation set are shown in table 10.

# Train/Val pairs	Model name			
	`ml_full`	`ml_nonames`	`ml_sbert_full`	`ml_sbert_nonames`
1,000	92.31 (RF)	78.79 (XGB)	92.31 (DT)	78.79 (XGB)
2,000	94.74 (RF)	88.31 (XGB)	94.87 (LOG)	92.11 (XGB)
3,000	97.44 (RF)	84.11 (RF)	97.44 (RF)	88.5 (RF)
4,000	93.42 (XGB)	89.19 (RF)	93.51 (RF)	85.14 (RF)
5,000	97.44 (RF)	83.8 (XGB)	97.96 (RF)	84.78 (XGB)
6,000	95.73 (RF)	83.41 (RF)	95.65 (RF)	83.18 (RF)
7,000	94.81 (RF)	86.82 (RF)	94.16 (RF)	87.02 (RF)
8,000	95.82 (XGB)	88.51 (RF)	95.82 (XGB)	89.86 (RF)
9,000	96.05 (XGB)	86.59 (RF)	96.02 (XGB)	87.8 (RF)
10,000	97.46 (XGB)	87.67 (RF)	97.45 (XGB)	87.98 (RF)

Table 9: F_1 scores of the ML-based model variants on the validation sets, along with the best-performing classifier for each combination of model and validation set size.

# Train/Val pairs	Model name			
	bert_full_map	bert_nonames_map	bert_full_nomap	bert_nonames_nomap
1,000	58.82	23.4	26.92	20
2,000	92.11	81.69	87.5	49.54
3,000	89.43	83.76	70.49	72.31
4,000	92.9	80.46	85.71	68.24
5,000	95.38	90.1	93.47	86.46
6,000	97.02	91.77	94.35	88.98
7,000	97.14	89.61	92.68	90.18
8,000	96.30	91.26	92.9	91.08
9,000	95.18	90.45	94.05	91.27
10,000	95.96	89.67	90.54	90.39

Table 10: F_1 scores of the BERT-based model variants on the validation sets.

As in the literature, classifiers based on tree ensembles seem to perform best on the tabularised authority data. The Naive Bayes classifier consistently yields low F_1 scores compared to the other classifiers. For instance, validation on the training/validation set of 8,000 pairs results in an F_1 score of just 18.33, for example (precision: 10.09; recall: 100). This performance pattern is consistent with the findings of Chen et al. [2019]; as previously noted, multiple features are extracted from each attribute pair in the schema mapping, leading to conditional dependencies among features that violate the independence assumption of Naive Bayes.

Despite lowering the number of epochs, the BERT models continued to exhibit signs of potential overfitting. Table 11 illustrates this with the initial loss and the F_1 score on the validation set after each epoch during a representative training run of the `bert_full_map` model. For bigger training set sizes in particular, both loss and F_1 score converge rapidly, with only marginal gains in later epochs.

Epoch	Dev F_1 score	Loss at step 0
1	82.79	0.75084
2	90.64	0.10109
3	92.31	0.03287
4	91.71	0.01156
5	90.86	0.00057
6	93.13	0.00023
7	92.78	0.03100
8	92.98	0.00008
9	93.77	0.00010
10	93.77	0.00010

Table 11: Example training run of `bert_full_map` with a training/validiation set of size 8,000.

As expected, the BERT-based models required substantially more training time than the ML-based models. Table 12 shows the average time needed to fit the ML classifiers compared to the average time needed to fine-tune `distilbert-base-multilingual-cased` across increasing training/validation set sizes: Using `ml_full` and `bert_full_map` as rep-

resentative examples, training a traditional ML classifier (random forests or XGBoost in this case) on the subsets takes between 0.33 and 3.19 seconds. In contrast, fine-tuning `distilbert-base-multilingual-cased` takes between 47.20 and 169.58 seconds, highlighting the considerable computational overhead of the BERT-based entity matchers.

# Train/Val pairs	Model type	
	ML-based	BERT-based
1,000	0.29	48.07
2,000	0.44	77.89
3,000	0.67	98.75
4,000	0.9	104.37
5,000	1.17	117.61
6,000	1.58	128.39
7,000	1.97	139.22
8,000	2.37	158.99
9,000	2.82	168.01
10,000	3.34	184.20

Table 12: Average time taken (in seconds) for training and validating each model type.

6.3 Evaluation of testing

6.3.1 Overall results

The F_1 scores of each classification model on the test set depending on the size of the training/validation set used during the training process are listed in tables 13a and 13b and visualised in figures 9a and 9b. Overall, there's a notable difference in performance between models trained on datasets with naming attributes included and those without: Models including attributes containing preferred or alternative names perform better on average than the models excluding them. ML-based models almost consistently outperform BERT-based models. Like during validation, the BERT-based classifier performs rather poorly for smaller training/validation set sizes, largely due to insufficient training data and overfitting. Early overfitting is especially evident in the `nonames` variants: `bert_nonames_map` trained on the training/validation set of size 2,000 exhibits an absolute F_1 score difference between validation and test of 30.97, for example. The performance drop between `bert_full_nomap` trained and validated on 2,000 and 3,000 record pairs is an especially striking example of the BERT model struggling to fit to the training data, visible from an already comparatively low F_1 score of 70.49 on the validation set. Interestingly, only the `full` variant exhibits this behaviour – the `nonames` variant doesn't show a similar performance drop across the same data sizes. This suggests that the overfitting is tied specifically to the name attributes in this case, likely due to ambiguous or inconsistently serialized name data in this portion of the training set.

As training/validation set sizes increase, `bert_full_map` eventually matches the performance of `ml_full` and `ml_sbert_full`, for both `full` and `nonames` variants. Surprisingly, however, neither the BERT-based models nor the ML-based models enhanced with SBERT to compare longer textual attributes manage to substantially outperform `ml_full`

and `ml_nonames`, which rely purely on syntactic record pair comparisons. Despite their considerably higher training cost – in terms of the necessary amount of training data and training time – BERT-based models fail to leverage BERT's semantic capabilities to achieve a meaningful performance gain and ultimately converge with the performance of the simpler ML-based models.

In the following, select models will be evaluated by comparing and interpreting their performance on the test set. This analysis will proceed along three axes: **ML-based vs. BERT-based**, **"syntactic" vs. "semantic" ML-based** and **mapped vs. non-mapped BERT-based**. To not exceed the scope of this thesis, this will be done by trying to interpret the classification outcome on sample records which are correctly classified by only a part of the respective compared models.

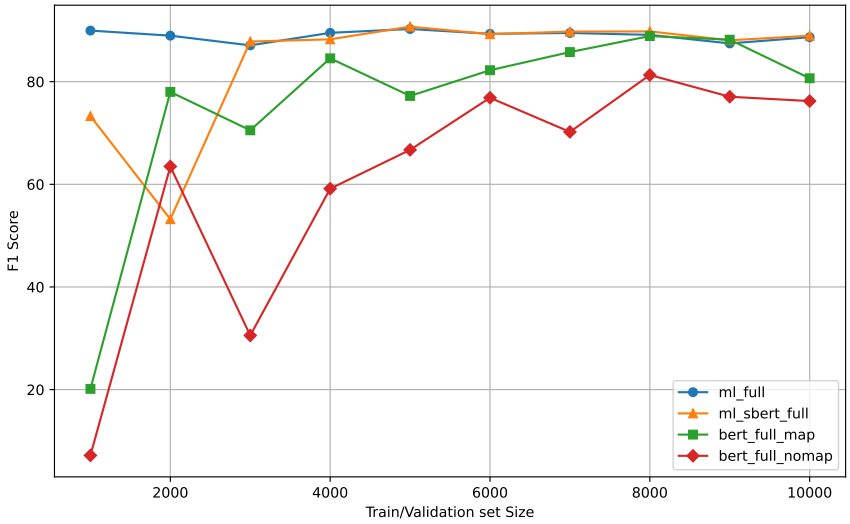
# Train/Val pairs	Model name			
	`ml_full`	`ml_nonames`	`ml_sbert_full`	`ml_sbert_nonames`
1,000	89.95	72.3	73.3	72.93
2,000	88.97	70.58	53.2	71.5
3,000	87.08	76.81	87.82	76.7
4,000	89.51	76.62	88.25	77.87
5,000	90.27	76.99	90.72	76.23
6,000	89.32	75.32	89.25	76.55
7,000	89.49	77.9	89.76	78.42
8,000	89.12	76.11	89.79	79.08
9,000	87.45	79.58	88.06	80.06
10,000	88.66	76.87	88.95	78.47

(a) ML-based models.

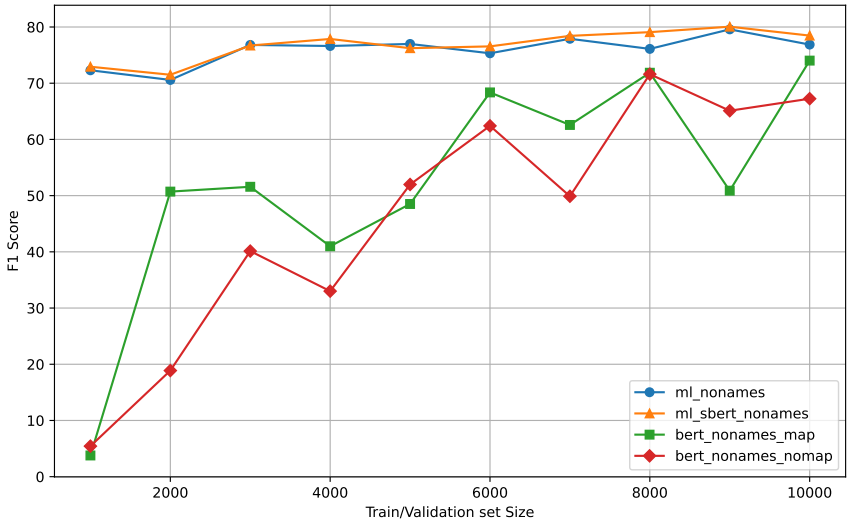
# Train/Val pairs	Model name			
	`bert_full_map`	`bert_nonames_map`	`bert_full_nomap`	`bert_nonames_nomap`
1,000	20.15	3.77	7.17	5.43
2,000	78.01	50.72	63.48	18.88
3,000	70.52	51.57	30.55	40.14
4,000	84.55	40.99	59.15	33.01
5,000	77.22	48.52	66.71	51.98
6,000	82.23	68.36	76.87	62.41
7,000	85.76	62.58	70.21	49.89
8,000	88.89	71.86	81.3	71.61
9,000	88.17	50.88	77.07	65.10
10,000	80.66	74.02	76.22	67.24

(b) BERT-based models.

Table 13: F_1 scores of all model variants on the test set.



(a) With all attributes.



(b) Without name attributes.

Figure 9: F_1 score of all models on the test set depending on training/validation size.

6.3.2 Analysis of BERT-based vs. ML-based model variants

Comparing the performance of BERT-based and ML-based entity matchers proved challenging: As already mentioned in chapter 5.1, BERT classifiers in particular are highly opaque and difficult to interpret; as we can't assign a fixed position to each attribute in the BERT input strings (due to authority records differing a lot in terms of their attributes),

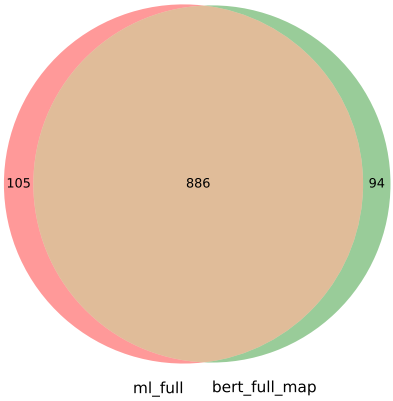


Figure 10: The overlap between the true positive matches found by `ml_full` and `bert_full_map` trained and validated on a set with 8,000 record pairs.

the influence of individual attributes can't be reliably measured. Furthermore, Paganelli et al. [2024] found that the reliance of BERT models on semantic similarity between corresponding attributes actually decreases with fine-tuning on the entity-matching task and contextual knowledge on a sentence level gains in importance to the classification outcome.

When trained and validated on 8,000 record pairs, `ml_full` and `bert_full_map` achieve high F_1 scores of 89.12 and 88.89, respectively. Although both models identify a substantial and largely overlapping set of true-positive matches (991 and 980, respectively), their outputs are not completely identical, as illustrated in figure 10: `ml_full` and `bert_full_map` respectively correctly identify 105 and 94 record pairs as matching that the respective other classifier misses. Two example true-matching record pairs are shown in table 14: The record pair referencing the ancient Greek historian Phylarchus is correctly classified as a match by the `bert_full_map` model, but not by `ml_full`. This differing classification is relatively easier to interpret, as both records contain non-empty values for only three attribute pairs. It is likely that the GND record is too sparse – lacking, for instance, information on Phylarchus' date of birth, gender, or profession – for the ML-based classifier to confidently identify the pair as a match. BERT appears to leverage contextual information more effectively in this case, for example by recognizing that the term *Historiker* (historian) occurs in multiple fields that are not explicitly aligned in the schema mapping.

The record pair referring to the Austrian writer Eduard Silesius shown in table 15 on the other hand illustrates the difficulty of interpreting the BERT classifier: It is correctly classified by `ml_full`, but not by `bert_full_map`. In this case, both records are rich in information, which likely enabled the ML-based classifier to recognize the match. However, the reasoning behind `bert_full_map`' s misclassification remains opaque. Interestingly, while `bert_full_map` and `bert_nonames_nomap` incorrectly classify the pair as non-matching, `bert_full_nomap` and `bert_nonames_map` do predict a match. This inconsistency further underscores the limited interpretability of these models.

_id	10019494X
_pref_name	Phylarchus Atheniensis
variantNameForThePerson	Phylarchus, Historicus
	Phylarchos, von Athen
	Phylarchos, von Sikyon
	Phylarchus
	Phylarchos, von Naukratis
	Filarco, di Atene
	Phylarchos, Historiker
	Phylarchos
	Filarco, di Naucrati
	Phylarchus, of Athens
biographicalOrHistorical Information	FGrH 81
gender	NaN
professionOrOccupation	NaN
professionOrOccupationAs Literal	NaN
geographicAreaCode	Griechenland (Altertum)
dateOfBirth	NaN
dateOfDeath	NaN
gndSubjectCategory	Personen der Geschichtswissenschaft (Historiker, Archäologen)
placeOfBirth	NaN
placeOfActivity	NaN
placeOfDeath	NaN
affiliation	NaN

_id	Q609147
_pref_name	Phylarchos
altLabel	Phylarchos von Athen
	Phylarchus of Athens
description	griechischer Historiker
	3rd-century BC Greek historian
sex_or_gender	männlich
occupation	Historiker
	Schriftsteller
	Mythograph
country_of_citizenship	NaN
date_of_birth	-0254-01-01
date_of_death	NaN
field_of_work	NaN
place_of_birth	Antikes Athen
residence	NaN
place_of_death	NaN
employer	NaN
member_of	NaN
member_of_political_party	NaN

Table 14: An example true-matching person record pair correctly classified as matching by `bert_full_map`, but not by `ml_full`.

_id	11603047X
_pref_name	Eduard Silesius
variantNameForThePerson	Silesius, Eduard
	Badenfeld, Czeike von
	Badenfeld, Eduard von
	Badenfeld, Eduard Karl Franz Heinrich Eusebius Johann Sarkander Freiherr Czeike von
	Czeike von Badenfeld, Eduard
	Badenfeld, Eduard von, Freiherr
	Badenfeld, Eduard
	Badenfeldt, ... von
	Badenfeld, Eduard Freiherr von
biographicalOrHistorical Information	Freiherr
gender	männlich
professionOrOccupation	Schriftsteller
	Lyriker
	Dramatiker
	Jurist
professionOrOccupationAs Literal	NaN
geographicAreaCode	Deutschland
	Österreich
	Tschechische Republik
dateOfBirth	1800-08-14
dateOfDeath	1861-12-06
gndSubjectCategory	Personen zu Literaturgeschichte (Schriftsteller)
	Personen zum Recht
	Personen zu Bau, Boden
placeOfBirth	Troppau
placeOfActivity	Wien
	Dresden
placeOfDeath	Slezské Rudoltice (Bruntál)
affiliation	NaN

_id	Q55847382
_pref_name	Eduard Freiherr von Badenfeld
altLabel	Eduard Karl Franz Heinrich Eusebius Johann Sarkander Czeike von Badenfeld
	Eduard von Badenfeld
	Eduard Silesius
description	(1800-1860)
	1800 bis 1860 Geburtsort Troppau Beruf/Funktion Schriftsteller ; Jurist Konfession katholisch Namensvarianten Badenfeld, Eduard Karl Franz Heinrich Eusebius Johann Sarkander Freiherr von Czeike von Badenfeld, Eduard Freiherr von Czeike von Badenfeld,
sex_or_gender	männlich
occupation	Philosoph
	Dichter
	Dramaturg
	Prosaist
	Journalist
	Jurist
country_of_citizenship	Kaisertum Österreich
date_of_birth	1800-08-14
date_of_death	1860-12-06
field_of_work	NaN
place_of_birth	Opava
residence	NaN
place_of_death	Slezské Rudoltice
employer	NaN
member_of	NaN
member_of_political_party	NaN

Table 15: An example true-matching person record pair correctly classified as matching by `ml_full`, but not by `bert_full_map`.

6.3.3 Analysis of "syntactic" vs. "semantic" ML-based model variants

The ML-based models perform rather consistently on both datasets with the naming attributes included and excluded alike; interestingly, the ML model variants using semantic feature extraction for long textual attributes using SBERT don't significantly outperform ML model variants trained on purely syntactical comparison vectors, given enough training data. This may have multiple reasons: On the one hand, the textual values of the `biographicalOrHistoricalInformation` and `description` attributes differ a lot in length and complexity; measuring the distance between the embeddings of these textual attributes does not offer a particular advantage over traditional string metrics if one or both of the attribute values is too short or differs too much in content from the other. A few examples are shown in table 16: For Rosa Luxemburg both the GND and the Wikidata record contain descriptive attributes consisting of shorter strings with similar meaning. The descriptions of the records referring to Albert Einstein have different foci, with the GND record's description being far longer and giving a brief summarisation of Einstein's professional career, while the Wikidata `description` focusses on Einstein's key scientific discovery, the theory of relativity. Both descriptions still roughly have the same subject (Einstein's scientific career). The descriptions for German TV presenter Sandra Maischberger, meanwhile, are shorter strings with completely different subject matters: The GND record's `biographicalOrHistoricalInformation` solely focusses on Maischberger having received an order of merit, while her Wikidata `description` shortly states her profession and nationality.

GND		**Wikidata**	
`118575503`	war Reichstagsabgeordnete der Deutschen Demokratischen Partei	`Q7231`	deutsche Politikerin (DDP, FDP), MdR/MdB und Frauenrechtlerin
`118529579`	1894-1913 überwiegend in der Schweiz, 1901 Schweizer Staatsbürger; 1914-1933 in Deutschland, 1921 Nobelpreisträger für Physik; emigrierte 1933 in die USA, ab 1940 amerikanischer Staatsbürger. Ehrenmitglied des Physikalischen Vereins. Ehrendoktor der Universität Rostock. Ehrendoktor der Universidad Nacional Mayor de San Marcos. Ehrendoktor der Princeton University. Ehrendoktor der ETH Zürich.	`Q937`	theoretischer Physiker, Entwickler der Relativitätstheorie (1879–1955), Nobelpreisgewinner 1922
`123685192`	Trägerin des Bundesverdienstkreuzes am Bande.	`Q106538`	deutsche Journalistin und Fernsehmoderatorin

Table 16: Examples for textual values of descriptional attributes of matching person record pairs.

Additionally, as shown by the analysis of the attribute frequencies across the person records contained in the GND and Wikidata MongoDB collections (see table 5), below half of the matching record pairs between `gnd_linked` and `wd_linked` have both the `biographicalOrHistoricalInformation` and `description` attributes. This further lowers the importance of features extracted from these attributes for classification. Overall, the differences in performance between `ml_full` and `ml_sbert_full` are very small and

inconsistent. For a training/validation dataset size of 7,000, `ml_sbert_full` slightly outperforms `ml_full`, while `ml_full` demonstrates marginally better performance on dataset sizes 9,000 and 10,000. These discrepancies can likely be attributed to variations in the specific training/validation subsets, rather than to inherent strengths of one model over the other. Comparing `ml_nonames` and `ml_sbert_nonames`, `ml_sbert_nonames` exhibits a small advantage over `ml_nonames` in F1 score when training and validating on bigger subsets. In table 17, four example record pairs with non-empty description fields that are correctly classified by `ml_sbert_nonames` but missed by `ml_nonames`, at a training/validation size of 8,000, are shown. Notably, the GND and Wikidata records for Bulgarian geneticist Robert Penchovsky (with IDs `124534015` and `Q58804342`, respectively) are correctly classified as a match, even though the Wikidata description consists solely of his ORCID ID. This suggests that `ml_sbert_nonames` is able to capture latent semantic cues from sparse or indirect descriptions – in this case, associating research-related context across both records. While naming attributes generally carry the most discriminative power, this example highlights the added value of incorporating rich textual fields, particularly for edge cases.

GND		**Wikidata**	
`1028555997`	Professor of Criminal Law, Tilburg University	`Q30127772`	niederländischer Rechtswissenschaftler
			Dutch lawyer
`130291404`	Russ. Künstlerin, Grafikerin, Buchillustratorin	`Q112414055`	ruská grafička, malířka, ilustrátorka
`140530231`	spielt zahlreiche Instrumente	`Q21755460`	deutsche Singer-Songwriterin und Geigerin
	Bayerische Sängerin		German singer-songwriter and violinist
`124534015`	geb. in Sofia, Diss. Köln 2003	`Q58804342`	researcher ORCID ID = 0000-0001-9502-3421
	Diss. Fachbereich Genetik		

Table 17: The descriptional attribute values of four record pairs correctly classified as matches by `ml_sbert_nonames`, but missed by `ml_nonames`.

The performance of ML classifiers based on syntactical comparisons only differs substantially from models using semantical comparisons on two occasions: For training/validation sets of size 1,000 and 2,000, `ml_sbert_full` achieves a considerably lower F_1 score than `ml_full`. This can be explained by the underlying classifiers selected for these training/validation runs: As visible in table 9, the decision tree and logistic regression classifier achieve the highest F_1 scores on the respective validation sets. As mentioned in chapter 2.2.4, ensemble learning techniques prevent overfitting due to their ability to combine the predictions of multiple models. These are the only cases where non-ensemble models were used and the F_1 score difference between validation and test sets is notably larger than for other ML models, suggesting overfitting in these training runs. For example, `ml_sbert_full` with a logistic regression classifier trained and validated on 2,000 pairs shows an absolute F_1 score drop of 41.67 points from validation to test, whereas the same model with a random forest classifier trained on 3,000 pairs exhibits a smaller drop of just 9.62 points. This supports the notion that tree-based ensemble models, such as random forests and XGBoost, are preferable for classifying authority data due to their lower tendency to overfit.

6.3.4 Analysis of mapped vs. non-mapped BERT-based model variants

In comparison to the ML-based models, the BERT-based models initially perform poorly. This is to be expected, as deep learning architectures usually need more training data to learn features compared to classical machine-learning models. The priorly discussed possible advantages of BERT-based classifiers are not very visible in the performance results. Despite not having to rely on a schema mapping, the BERT-based models using the manually crafted mapping (`bert_full_map` and `bert_nonames_map`) during the serialisation of the person records frequently outperform those that not using it (`bert_full_nomap` and `bert_nonames_nomap`). For some records, this might be caused by the serialisation process itself: As detailed in chapter 4.4.2, if including all attribute values into the serialised string yields a string exceeding the maximum number of input tokens of 512, only the first element of each attribute value set is added to the string resulting from serialisation. This approach ensures that as much information as possible from the few attributes in the schema mapping is included and as many different attributes as possible in the case of non-schema-mapped serialisation; however, for many records in `diff_gnd_75` and `diff_wd_75` with rich sets of attribute values this means omitting information which would benefit the detection of matching records.

Restricting the attributes to include in the serialisation to those in the schema mapping naturally increases the probability that a serialised record pair might be short enough to be able to be ingested by BERT with all attribute values included: To illustrate this, two example record pairs serialised by `bert_full_map` and `bert_full_nomap` and referencing the Scottish philosopher and theologian Richard of Saint Victor are respectively shown in figures 11a and 11b. Both records have a multitude of attributes, resulting in the record pair serialisation of `bert_full_nomap` only being able to include the first element of each attribute value set in the serialisation of the GND record, thereby omitting 13 of the 14 variant name labels recorded for Richard of Saint Victor (like *Richard, von Saint-Vanne* which is very close to the Wikidata record's preferred name label), one of his recorded professions (*religious order priest*), and a descriptional string referencing a source about his life. The Wikidata record serialisation by `bert_full_nomap` includes additional attributes potentially providing more contextual information like Richard of Saint Victor's held positions, his canonization status, feast day and religious order. Although the `bert_full_nomap` is more balanced in the string length dedicated to each record and it includes broader information, `bert_full_nomap` fails to correctly classify both records as matching, while `bert_full_map` does (both trained on a training/validation set.[1]

The models trained and tested on strings that exclude name attributes – `bert_nonames_map` and `bert_nonames_nomap` – do not exhibit a consistent pattern of one outperforming the other. Incorporating more attributes doesn't result in better performance here either, indicating again that the schema mapping covers most information relevant for classification. Non-mapped serialised record pairs are summarised more often, resulting in a loss of informational depth for particularly important attributes like name attributes. Because the person records differ widely in structure, the amount of relevant information BERT can learn from the summarised training datasets and the resulting performance on the test dataset also varies, with `bert_nonames_nomap` outperforming `bert_nonames_map` for training/validations sets of size 5,000 and 9,000, and the latter outperforming the former for the rest of the evaluated training/validation set sizes.

[1] With both model variants being trained on a training/validation set consisting of 8,000 record pairs.

```
COL _id VAL 103115145 COL _pref_name VAL Richardus de Sancto Vitone COL variantNameForThePerson
VAL Richardus, Abbas ; Richard, von Saint-Vanne ; Richardus, Abbas Sancti Vitoni ; Richard, of Saint Vanne ;
Richard, de Saint-Vannes ; Sancto Vitone, Richardus de ; Richard, de Verdun ; Richardus, Virdunensis ;
Richardus, Sancti Vitoni ; Richardus, von St.-Vanne ; Richard, von Sankt Vanne ; Richard, von Verdun ;
Richard, de Saint-Vanne ; Richardus, Vannensis COL biographicalOrHistoricalInformation VAL OSB;
Abt des Klosters Saint-Vanne (Verdun); Vita et miracula S. Vitoni Episcopi COL gender VAL männlich
COL professionOrOccupation VAL Abt ; Ordenspriester COL geographicAreaCode VAL Frankreich COL dateOfBirth
VAL 0970 COL dateOfDeath VAL 1046 COL affiliation VAL Benediktiner COL gndSubjectCategory VAL Personen zu
Kirchengeschichte, Systematischer und Praktischer Theologie, Kirche und Konfession

COL _id VAL Q3431374 COL _pref_name VAL Richard von St. Vanne COL sex_or_gender VAL männlich COL occupation
VAL katholischer Priester ; katholischer Bischof COL place_of_death VAL Bantheville COL place_of_birth
VAL Bantheville COL date_of_birth VAL 0970-01-01 COL date_of_death VAL 1046-06-20 COL description
VAL French abbot ; lothringischer Mönch, Abt von Saint-Vanne in Verdun COL altLabel VAL Richard von Grandpre ;
Richard of Verdun ; Richard van Saint-Vanne
```

(a) `bert_full_map`.

```
COL _id VAL 103115145 COL _pref_name VAL Richardus de Sancto Vitone COL _phon_name VAL RXRTSTSNKTFTN
COL variantNameEntityForThePerson VAL Richardus de Sancto Vitone COL variantNameForThePerson VAL Richardus,
Abbas COL preferredNameForThePerson VAL Richardus, de Sancto Vitone COL biographicalOrHistoricalInformation
VAL OSB; Abt des Klosters Saint-Vanne (Verdun); Vita et miracula S. Vitoni Episcopi COL gender
VAL männlich COL professionOrOccupation VAL Abt COL geographicAreaCode VAL Frankreich COL dateOfBirth
VAL 0970 COL dateOfDeath VAL 1046 COL affiliation VAL Benediktiner COL gndSubjectCategory VAL Personen
zu Kirchengeschichte, Systematischer und Praktischer Theologie, Kirche und Konfession

COL _id VAL Q3431374 COL _pref_name VAL Richard von St. Vanne COL _phon_name VAL RXRTFNSTFN ; RKRTFNSTFN
COL sex_or_gender VAL männlich COL occupation VAL katholischer Priester ; katholischer Bischof COL given_name
VAL Richard COL place_of_death VAL Bantheville COL place_of_birth VAL Bantheville COL date_of_birth
VAL 0970-01-01 COL date_of_death VAL 1046-06-20 COL label VAL Richard de Saint-Vanne ; Richard von St.
Vanne COL name VAL Richard de Saint-Vanne ; Richard von St. Vanne COL description VAL French abbot ;
lothringischer Mönch, Abt von Saint-Vanne in Verdun COL altLabel VAL Richard von Grandpre ; Richard of
Verdun ; Richard van Saint-Vanne COL position_held VAL Abt ; Bischof COL canonization_status VAL Seliger
COL feast_day VAL 14. Juni COL religious_order VAL Benediktiner
```

(b) `bert_full_nomap`

Figure 11: An example record pair consisting of GND record `103115145` and Wikidata record `Q3431374`, string-serialised for ingestion by two BERT-based models with and without restricting attributes ending up in the serialisation to the schema mapping.

Chapter 7

Discussion and Conclusion

For this thesis, an entity-matching workflow for person records from two authority databases, namely the GND and Wikidata, was implemented. Subsequently, different learning-based approaches for classifying record pairs as matches and non-matches were compared. During preprocessing, some amount of manual configuration was necessary to convert the RDF dumps into a workable format: The record types designating person records in both databases had to be specified for parsing, for example. Additionally, the importance of attributes containing personal names became apparent early on, with the predicate referring to the preferred personal name being shared by most of the records contained in the RDF dumps.

Performing entity matching on real-life data always requires some manual efforts to adjust to the structure of the databases to be reconciled [Christen, 2012]. For traditional classification efforts, this holds especially true for rule-based classification; manually crafting these rules necessitates that the implementers know about which attributes correspond to each other and how important said attributes are for disambiguating matching or non-matching record pairs. In contrast, learning-based classifiers are able to learn from the training data directly, dynamically allocating weights to the comparison vector resulting from record pair comparison. BERT-based models additionally don't necessarily need a schema mapping – architectures like Ditto which compare records sentence-pair-wise don't need to be provided with attribute correspondences.

The dynamic weighting of attributes and potential non-reliance on a schema mapping makes learning-based entity matchers attractive for authority file reconciliation efforts: Learnt weights for the comparison vectors can potentially lead to better performance compared to existing software solutions such as Mix'n'Match (which solely focusses on the preferred record label for classification) or OpenRefine (which requires the implementation of a reconciliation service).

Regarding general **feasibility**, applying existing learning-based entity-matching approaches on the GND and Wikidata proved successful. Despite a significant imbalance in attribute distribution with few attributes being shared by the majority of records, ML-based as well as BERT-based classifiers generally showed strong performance on authority data. The entity-matching approaches demonstrated notable robustness in handling sparse records.

Regarding the **effectiveness** of the compared models, the results varied more strongly. Although BERT-based classifiers were priorly assumed to have an advantage over clas-

sical machine learning because of their semantic understanding capabilities, they didn't manage to substantially outperform the ML-based models. BERT-based classifiers only approached the ML-based models' performance when trained with comparatively larger amounts of training data ($\geq$ 7,000 record pairs) despite needing considerably more time to train. One "disadvantage" of applying learning-based entity matching to authority data is the necessary gold-standard training data, which classifiers must be provided with and, in practice, often require manual annotation. In terms of "cost effectiveness", both computationally as well as in terms of time, ML-based models seem to be the better solution for the person records used in this case study; they achieve high F_1 scores with relatively small amounts of training data and require only a fraction of the training time compared to BERT-based models.

During the experiments conducted as part of this thesis, the dependency of entity-matching performance on the underlying data and thus the limitations of this case study's representativeness became apparent, however. The trade-off between high attribute variance and high-quality gold-standard record pairs became apparent: Authority records from highly interlinked authority files like the GND and Wikidata can serve as high-quality ground-truth source data to train, validate and test classifiers on – the more links there are between records across authority files, the more likely there have already been prior reconciliation efforts. As the GND and Wikidata both are cooperative authority files (or an authority-file-like knowledge base in Wikidata's case) which mainly contain records in German or English, many matching record pairs across both databases have similar attribute values, potentially even integrated from the respective other database. To be able to evaluate learning-based approaches on high-quality data while still getting realistic results for the authority file reconciliation use case, a synthetic dataset of interlinked records had to be built from the full set of linked GND and Wikidata person records. This was done to facilitate a more realistic diversity of attribute values and to prevent overfitting on the preferred name attribute. The latter's importance was further demonstrated during indexing: Blocking on `_pref_name` proved far more effective than blocking on the birth year and gender of a person.

The severe imbalance in attribute distribution became more evident during the experiments, particularly when training, validating, and testing the learning-based entity matchers. The high variance in attribute distribution likely contributed to the inconsistent growth in F_1 scores of fine-tuned BERT models, depending on the dataset used for training and validation. Models that included name attributes almost consistently performed better on the data; when serialisation prioritised breadth of attribute information as in the case of the `nomap` BERT model variants, models that allowed more name variants to be included in the serialised strings performed better. However, attribute detail loses importance when name attributes are omitted. Interestingly, the hybrid approach of incorporating the cosine distance of SBERT embeddings of long textual attributes did not significantly improve matching performance, with only a slight effect when name attributes were excluded.

Although BERT-based approaches didn't outperform models based on classical machine learning in this case study, they shouldn't be discounted completely: The BERT classifiers catching up performance-wise to the ML-based models shows that they can infer relevant patterns from authority data. As they don't need a fixed schema mapping, they can support end users without much knowledge of an authority file's underlying data

schema in reconciliation efforts. Furthermore, as mentioned, both authority files selected for this case study are well integrated and had most information contained in the records available in the same or at least closely related languages (with Wikidata having German- or at least English-language labels for most entries referred to by person records), thus contributing to keep the success rate of syntax-based record pair comparisons high. In use cases where two authority databases in two linguistically more distant languages are to be reconciled, multilingual BERT models like `distilbert-base-multilingual-cased` might still be perform adequately on semantically similar data, while ML-based approaches solely using syntactic string similarity measures might fail. In terms of **scalability**, fine-tuned BERT classifiers thus might be suitable for performing entity matching on linguistically dissimilar authority files.

This thesis serves as a first case study addressing the research gap between learning-based entity matching and authority control (and authority file reconciliation in particular). To further explore the applicability of learning-based entity matching on authority data, future research should focus primarily on evaluating the effectiveness of the presented approaches with yet unlinked authority records. In this context, a performance comparison with prevalent non-learning-based reconciliation solutions such as OpenRefine and Mix'n'Match would also be valuable, especially considering the considerable importance of the preferred name attribute. The advantages (or lack thereof) of learning classifiers on "linguistically close" authority files should also be explored further; as mentioned earlier, BERT-based entity matchers might prove more effective than other approaches when applied to semantically similar data that exhibits significant syntactic and structural dissimilarity, as seen in the case of linguistically distant languages. Additionally, applying data augmentation techniques or incorporating domain knowledge into the input data could further improve the performance of fine-tuned BERT classifiers on authority data. Overall, future investigations into these areas could provide valuable guidance for refining existing authority reconciliation methods and shaping future research in authority control and entity matching alike.

Bibliography

Adams, B. (2021). Chronotopic information interaction: integrating temporal and spatial structure for historical indexing and interactive search. *Digital Scholarship in the Humanities*, 36(3):525–541.

Barlaug, N. and Gulla, J. A. (2021). Neural Networks for Entity Matching: A Survey. *ACM Transactions on Knowledge Discovery from Data*, 15(3):1–37. arXiv:2010.11075 [cs].

Bartalesi, V., Metilli, D., Pratelli, N., and Pontari, P. (2022). Towards a knowledge base of medieval and renaissance geographical Latin works: The IMAGO ontology. *Digital Scholarship in the Humanities*, 37(1):34–50.

Beek, W., Raad, J., Wielemaker, J., and van Harmelen, F. (2018). sameAs.cc: The Closure of 500M owl:sameAs Statements. In Gangemi, A., Navigli, R., Vidal, M.-E., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., and Alam, M., editors, *The Semantic Web*, pages 65–80, Cham. Springer International Publishing.

Berners-Lee, T. (2006). Linked Data. `https://www.w3.org/DesignIssues/LinkedData.html` [Accessed: 2025-03-03].

Bianchini, C., Bargioni, S., and Girolamo, C. C. P. d. S. (2021). Beyond VIAF: Wikidata as a Complementary Tool for Authority Control in Libraries. *Information Technology and Libraries*, 40(2). Number: 2.

Binette, O. and Steorts, R. C. (2022). (Almost) all of entity resolution. *Science Advances*, 8(12):eabi8021. Publisher: American Association for the Advancement of Science.

Borthwick, A., Ash, S., Pang, B., Qureshi, S., and Jones, T. (2020). Scalable Blocking for Very Large Databases. arXiv:2008.08285 [cs].

Brunner, U. and Stockinger, K. (2020). Entity matching with transformer architectures - a step forward in data integration. Technical report, OpenProceedings. Conference Name: 23rd International Conference on Extending Database Technology, Copenhagen, 30 March - 2 April 2020 ISBN: 9783893180837.

Buckles, K., Haws, A., Price, J., and Wilbert, H. E. (2023). Breakthroughs in Historical Record Linking Using Genealogy Data: The Census Tree Project.

Busch, N. and Müller, D. (2023). Normdaten in den Geisteswissenschaften. *Zeitschrift für Literaturwissenschaft und Linguistik*, 53(3):781–796.

Charikar, M. S. (2002). Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, STOC '02, pages 380–388, New York, NY, USA. Association for Computing Machinery.

Chen, X., Campero Durand, G., Zoun, R., Broneske, D., Li, Y., and Saake, G. (2019). The Best of Both Worlds: Combining Hand-Tuned and Word-Embedding-Based Similarity Measures for Entity Resolution. pages 215–224. Gesellschaft für Informatik, Bonn.

Christen, P. (2012). *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer, Berlin, Heidelberg.

Christophides, V., Efthymiou, V., Palpanas, T., Papadakis, G., and Stefanidis, K. (2021). An Overview of End-to-End Entity Resolution for Big Data. *ACM Computing Surveys*, 53(6):1–42.

Cyganiak, R., Wood, D., and Lanthaler, M. (2014). RDF 1.1 Concepts and Abstract Syntax. `https://www.w3.org/TR/rdf11-concepts/` [Accessed: 2025-04-30].

Delpeuch, A. (2022). OpenRefine - Wikibase reconciliation interface. `https://openrefine-wikibase.readthedocs.io/en/latest/` [Accessed: 2025-04-23].

Deo, N., Basak, J., Soliman, A., Weinberg, D., Steorts, R., and Rajasekaran, S. (2023). Novel Blocking Techniques and Distance Metrics for Record Linkage. In Delir Haghighi, P., Pardede, E., Dobbie, G., Yogarajan, V., ER, N. A. S., Kotsis, G., and Khalil, I., editors, *Information Integration and Web Intelligence*, pages 431–446, Cham. Springer Nature Switzerland.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs] version: 2.

Edmond, J. and Nugent Folan, G. (2017). Data, Metadata, Narrative. Barriers to the Reuse of Cultural Sources. In Garoufallou, E., Virkus, S., Siatri, R., and Koutsomiha, D., editors, *Metadata and Semantic Research*, volume 755, pages 253–260. Springer International Publishing, Cham.

Erlinger, C. (2019). Sächsische Ortsdaten in der Linked Open Data Cloud: Teilautomatisierte Anreicherung und Analyse der HOV-ID in Wikidata. ISSN: 2629-5849.

Fagerving, A. (2023). Wikidata for Authority Control: Sharing Museum Knowledge With the World. *development*, 12:97.

Fang, L., Li, L., Liu, Y., Torvik, V. I., and Ludäscher, B. (2023). KAER: A Knowledge Augmented Pre-Trained Language Model for Entity Resolution. arXiv:2301.04770.

Feigenbaum, J. J. (2016). A Machine Learning Approach to Census Record Linking.

Fellegi, I. P., , and Sunter, A. B. (1969). A Theory for Record Linkage. *Journal of the American Statistical Association*, 64(328):1183–1210. Publisher: ASA Website _eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1969.10501049.

Fischer, B. and Hartmann, S. (2019). GND meets Wikibase. https://www.clarin.eu/sites/default/files/sshoc_session-open-source-vocabulary-platforms-1_gndxwikibase_clarin.pdf [Accessed: 2025-04-22].

Fischer, B., Hartmann, S., Scheven, E., Svensson, L. G., and Wiechmann, B. (2025). Normdaten, Linked Data. In Johannsen, J., Mittermaier, B., Schäffler, H., and Söllner, K., editors, *Praxishandbuch Bibliotheksmanagement*, pages 527–548. De Gruyter Saur, Berlin, Boston.

Galka, S. (2023). Digitale Edition und Kommentierung der Tagebücher des Fürsten Christian II. von Anhalt-Bernburg (1599–1656). *RIDE*, (16).

Girolamo, C. C. P. d. S. (2024). The reconciliation of SBN authority records with Wikidata. Progresses and perspectives after a decade of work (2013-2023). *JLIS.it*, 15(1):33–44. Number: 1.

Grinsztajn, L., Oyallon, E., and Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? *Advances in Neural Information Processing Systems*, 35:507–520.

Gross, M. and Mueller-Smith, M. (2021). Modernizing Person-Level Entity Resolution with Biometrically Linked Records.

Haffner, A. (2024). GND Ontology. https://d-nb.info/standards/elementset/gnd_20240806 [Accessed: 2025-04-25].

Harron, K., Goldstein, H., and Dibben, C. (2015). *Methodological Developments in Data Linkage*. John Wiley & Sons, Incorporated, Newark, UNITED KINGDOM.

Hawkins, A. (2022). Archives, linked data and the digital humanities: increasing access to digitised and born-digital archives via the semantic web. *Archival Science*, 22(3):319–344.

Hegselmann, S., Buendia, A., Lang, H., Agrawal, M., Jiang, X., and Sontag, D. (2023). TabLLM: Few-shot Classification of Tabular Data with Large Language Models. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR. ISSN: 2640-3498.

Heng, G., Cole, T. W., Tian, T. C., and Han, M.-J. (2022). Rethinking Authority Reconciliation Process. *Cataloging & Classification Quarterly*, 60(1):45–68. Publisher: Routledge _eprint: https://doi.org/10.1080/01639374.2021.1992554.

Hickey, T. B. and Toves, J. A. (2014). Managing Ambiguity in VIAF. *D-Lib Magazine*, 20(7/8).

Hoinkis, G. (2023). *Wie kommt die GND (Gemeinsame Normdatei) ins Archiv?* BibSpider, Birkenwerder.

Jegan, R., Fruth, L., Gradl, T., and Henrich, A. (2023). Integrating Access to Authority Data for Improved Interoperability of Research Data in the Digital Humanities. https://fis.uni-bamberg.de/handle/uniba/59341 [Accessed: 2025-02-18].

Jurafsky, D. and Martin, J. H. (2025). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd edition.

Koch, I., Ribeiro, C., Poveda-Villalón, M., Rico, M., and Teixeira Lopes, C. (2024). Enriching Archival Linked Data Descriptions with Information from Wikidata and DBpedia. In Antonacopoulos, A., Hinze, A., Piwowarski, B., Coustaty, M., Di Nunzio, G. M., Gelati, F., and Vanderschantz, N., editors, *Linking Theory and Practice of Digital Libraries*, pages 396–412, Cham. Springer Nature Switzerland.

Konda, P., Das, S., Suganthan G. C., P., Doan, A., Ardalan, A., Ballard, J. R., Li, H., Panahi, F., Zhang, H., Naughton, J., Prasad, S., Krishnan, G., Deep, R., and Raghavendra, V. (2016). Magellan: toward building entity matching management systems. *Proceedings of the VLDB Endowment*, 9(12):1197–1208.

Kooli, N., Allesiardo, R., and Pigneul, E. (2018). Deep Learning Based Approach for Entity Resolution in Databases. In Nguyen, N. T., Hoang, D. H., Hong, T.-P., Pham, H., and Trawiński, B., editors, *Intelligent Information and Database Systems*, pages 3–12, Cham. Springer International Publishing.

Krüger, M. (2024). Enhancing Cybercrime Investigations by Integrating RAKE and Case-Based Reasoning with Text Comparison Algorithms. Würzburg.

Li, Y., Li, J., Suhara, Y., Doan, A., and Tan, W.-C. (2020). Deep Entity Matching with Pre-Trained Language Models. *Proceedings of the VLDB Endowment*, 14(1):50–60. arXiv:2004.00584 [cs].

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs].

Low, J. F., Fung, B. C. M., and Xiong, P. (2024). Better entity matching with transformers through ensembles. *Knowledge-Based Systems*, 293:111678.

Magnus (2013). Red link lists on steroids. `http://magnusmanske.de/wordpress/archives/114` [Accessed: 2025-04-21].

Menzel, S., Schnaitter, H., Zinck, J., Petras, V., Neudecker, C., Labusch, K., Leitner, E., and Rehm, G. (2021). Named Entity Linking mit Wikidata und GND –Das Potenzial handkuratierter und strukturierter Datenquellen für die semantische Anreicherung von Volltexten. In Franke-Maier, M., Kasprzik, A., Ledl, A., and Schürmann, H., editors, *Qualität in der Inhaltserschließung*, pages 229–258. De Gruyter Saur, Berlin, Boston.

Mihindukulasooriya, N. (2024). DBLP to Wikidata: Populating Scholarly Articles in Wikidata. In *International Semantic Web Conference: Posters, Demos, and Industry Tracks*.

Mix'n'Match (2024). Manual. `https://meta.wikimedia.org/wiki/Mix%27n%27match/Manual` [Accessed: 2025-04-23].

Mudgal, S., Li, H., Rekatsinas, T., Doan, A., Park, Y., Krishnan, G., Deep, R., Arcaute, E., and Raghavendra, V. (2018). Deep Learning for Entity Matching: A Design Space Exploration. In *Proceedings of the 2018 International Conference on Management of Data*, pages 19–34, Houston TX USA. ACM.

Murphy, K. P. (2022). *Probabilistic machine learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts ; London, England.

Myntti, J., Lewis, N., McCormack, A. M., and Rockwell, K. (2020). Regional Connections to National Authority Files. *Cataloging & Classification Quarterly*, 58(1):76–89. Publisher: Routledge _eprint: https://doi.org/10.1080/01639374.2019.1690087.

On, B.-W., Sang Choi, G., and Jung, S.-M. (2014). A case study for understanding the nature of redundant entities in bibliographic digital libraries. *Program*, 48(3):246–271. Publisher: Emerald Group Publishing Limited.

OpenRefine (2022). User Manual. `https://openrefine.org/docs` [Accessed: 2025-04-21].

Padmanabhan, S., Carty, L., Cameron, E., Ghosh, R. E., Williams, R., and Strongman, H. (2019). Approach to record linkage of primary care data from Clinical Practice Research Datalink to other health-related patient data: overview and implications. *European Journal of Epidemiology*, 34(1):91–99.

Paganelli, M., Tiano, D., and Guerra, F. (2024). A multi-facet analysis of BERT-based entity matching models. *The VLDB Journal*, 33(4):1039–1064.

Papadakis, G., Skoutas, D., Thanos, E., and Palpanas, T. (2020). A Survey of Blocking and Filtering Techniques for Entity Resolution. arXiv:1905.06167.

Peeters, R. and Bizer, C. (2021). Dual-objective fine-tuning of BERT for entity matching. *Proc. VLDB Endow.*, 14(10):1913–1921.

Peeters, R. and Bizer, C. (2022). Supervised Contrastive Learning for Product Matching. arXiv:2202.02098.

Philips, L. (2000). The double metaphone search algorithm. *C/C++ Users Journal*, 18(6):38–43.

Primpeli, A., Peeters, R., and Bizer, C. (2019). The WDC Training Dataset and Gold Standard for Large-Scale Product Matching. In *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19, pages 381–386, New York, NY, USA. Association for Computing Machinery.

Putnam, N. (2022). VIAF and the linked data ecosystem. *Bibliographic control in the digital ecosystem. - (Biblioteche & bibliotecari, 2612-7709 ; 5)*, pages 196–202. Publisher: Associazione italiana biblioteche.

Rantala, H., Jokipii, I., Ikkala, E., and Hyvönen, E. (2022). WarVictimSampo 1914–1922: A National War Memorial on the Semantic Web for Digital Humanities Research and Applications. *J. Comput. Cult. Herit.*, 15(1):8:1–8:18.

Ravelli, E. and Mataloni, M. C. (2022). Integrated Search System: evolving the authority files. *Bibliographic control in the digital ecosystem.-(Biblioteche & bibliotecari, 2612-7709; 5)*, pages 335–346. Publisher: Associazione italiana biblioteche.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084 [cs].

Romesburg, H. C. (1984). *Cluster analysis for researchers*. Lifetime Learning Publ., Belmont, California.

Röchner, P. and Rothlauf, F. (2024). Using machine learning to link electronic health records in cancer registries: On the tradeoff between linkage quality and manual effort. *International Journal of Medical Informatics*, 185:105387.

Sardo, L. and Bianchini, C. (2022). Wikidata : a new perspective towards universal bibliographic control. *JLIS : Italian Journal of Library, Archives and Information Science = Rivista italiana di biblioteconomia, archivistica e scienza dell'informazione : 13, 1, 2022*, pages 291–311. Publisher: EUM-Edizioni Università di Macerata.

Sayers, A., Ben-Shlomo, Y., Blom, A. W., and Steele, F. (2016). Probabilistic record linkage. *International Journal of Epidemiology*, 45(3):954–964.

Schnell, R. (2022). Verknüpfung von Bildungsdaten in einem Bildungsregister mittels Record-Linkage auf Basis von Personenmerkmalen. Technical report, DuEPublico: Duisburg-Essen Publications online, University of Duisburg-Essen, Germany. Publication Title: GRLC Working Paper Series.

Shwartz-Ziv, R. and Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90.

Subagdja, B., Shanthoshigaa, D., Wang, Z., and Tan, A.-H. (2024). Machine Learning for Refining Knowledge Graphs: A Survey. *ACM Computing Surveys*, 56(6):1–38.

Thompson, R. and Mukhopadhyay, T. P. (2021). Digital arts in Latin America: A report on the archival history of intersections in art and technology in Latin America. *Digital Scholarship in the Humanities*, 36(Supplement_1):i113–i123.

Toves, J. A. and Hickey, T. B. (2014). Parsing and Matching Dates in VIAF. *The Code4Lib Journal*, (26).

Trabelsi, M., Heflin, J., and Cao, J. (2022). DAME: Domain Adaptation for Matching Entities. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1016–1024, Virtual Event AZ USA. ACM.

Tóth-Czifra, E. (2020). The Risk of Losing the Thick Description: Data Management Challenges Faced by the Arts and Humanities in the Evolving FAIR Data Ecosystem. In *Digital Technology and the Practices of Humanities Research*, pages 235–266. Open Book Publishers.

Vykhovanets, V. S., Du, J., and Sakulin, S. A. (2020). An Overview of Phonetic Encoding Algorithms. *Automation and Remote Control*, 81(10):1896–1910.

Wiederhold, R. A. and Reeve, G. F. (2021). Authority Control Today: Principles, Practices, and Trends. *Cataloging & Classification Quarterly*, 59(2-3):129–158. Publisher: Routledge _eprint: https://doi.org/10.1080/01639374.2021.1881009.

Wikidata (2024). WikiProject Ontology/Modelling. `https://www.wikidata.org/wiki/Wikidata:WikiProject_Ontology/Modelling` [Accessed: 2025-04-29].

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., Gonzalez-Beltran, A., Gray, A. J. G., Groth, P., Goble, C., Grethe, J. S., Heringa, J., ' t Hoen, P. A. C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S. J., Martone, M. E., Mons, A., Packer, A. L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M. A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J., and Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3(1):160018.

Wu, J., Sefid, A., Ge, A. C., and Giles, C. L. (2017). A Supervised Learning Approach To Entity Matching Between Scholarly Big Datasets. In *Proceedings of the 9th Knowledge Capture Conference*, K-CAP '17, pages 1–4, New York, NY, USA. Association for Computing Machinery.

Yang, M. Y. R., Yang, S., and Lin, J. (2022). Integration of text and geospatial search for hydrographic datasets using the lucene search library. In *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*, JCDL '22, pages 1–5, New York, NY, USA. Association for Computing Machinery.

Zhao, F. (2023). A systematic review of Wikidata in Digital Humanities projects. *Digital Scholarship in the Humanities*, 38(2):852–874.

Zhou, H., Huang, W., Li, M., and Lai, Y. (2022). Relation-Aware Entity Matching Using Sentence-BERT. *Computers, Materials & Continua*, 71(1):1581–1595. Publisher: Tech Science Press.