Knowledge Graph Enrichments for Credit Account Prediction

Michael Schulze[1,2] and Andreas Dengel[1,2]
[1]*Computer Science Department, Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau (RPTU), Germany*
[2]*Smart Data & Knowledge Services Departm., Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Germany*
{*firstname.lastname*}*@dfki.de*

Keywords: Knowledge Graphs, RDF, OWL, Rule-Based Link Prediction, Case-Based Reasoning.

Abstract: For the problem of credit account prediction on the basis of received invoices, this paper presents a pipeline consisting of 1) construction of an accounting knowledge graph, 2) enrichment algorithms, and 3), prediction of credit accounts with methods of a) rule-based link prediction, b) case-based reasoning, and c) a combination of both. Explainability and traceability have been key requirements. While preserving the order of invoices in cross-fold validation, key findings in our scenario are: 1) using all enrichments from the pipeline increases prediction performance up to 12.45 percent points, 2) single enrichments are useful on their own, 3) case-based reasoning benefits most from having enrichments available, and 4), the combination of link prediction and case-based reasoning yields best prediction results in our scenario.
Paper page: https://git.opendfki.de/michael.schulze/account-prediction.

1 INTRODUCTION

A typical task of accountants is the assignment of accounts based on received invoice data. For example, on an invoice where employees visited a restaurant for a business meeting, the accountant needs to first assess the invoicing case correctly, and then needs to find and choose the correct financial account. Through interviews with accountants from our business partner, we learnt that accountants in such situations face two major challenges: On the one hand, as also reported by Bardelli et al. (Bardelli et al., 2020), the search space regarding finding and choosing accounts can be very high. For example, when accountants are responsible for multiple company subsidiaries, accountants need to tap into multiple accounting handbooks where each handbook can be several hundred pages long. On the other hand, to assess the invoicing case correctly, accountants usually need to tap into several heterogeneous sources of information. For instance, in the example of the restaurant visit, the accountant may check a separate participation list to decide whether guests have been present or not. This would then result in different accounts. This list may come from another ERP system or, for example, by e-mail or some other intranet resource. Other sources of information can be, among others, organizational policies (e.g., regarding handling restaurant visits), accounting charts, tax regulations, or other attachments. These challenges can lead to long search times, especially for non-standard cases, for novice accountants, and for new employees. To tackle these challenges, this paper focuses on the outlined problem of credit account prediction on the basis of received invoices. The term credit account is used here in accordance with related work, such as in (Belskis et al., 2021) and (Belskis et al., 2020). In line with insights gained from interviews with accountants, and underlined by experiments reported in Bardelli et al. (Bardelli et al., 2020), throughout account prediction literature (e.g. (Panichi and Lazzeri, 2023; Bardelli et al., 2020)), credit account prediction is considered as especially challenging. However, current research focuses mainly on the goal of automatizing account prediction (Bardelli et al., 2020; Panichi and Lazzeri, 2023; Belskis et al., 2021); also with the explicit goal to exclude knowledge workers (e.g. (Belskis et al., 2021)). In contrast, but in line with forecasts from Jain and Woodcock (Jain and Woodcock, 2017), fully automatization is not seen as realistic. This assessment is also underlined by reported performance numbers in related work (e.g. (Panichi and Lazzeri, 2023; Bardelli et al., 2020; Belskis et al., 2021)), where results do not meet the level of accuracy an automated system would require in practice. Furthermore, through interviews with accountants where guidelines from Lazar et al. (Lazar et al., 2017) have been followed, explainability and

Schulze, M. and Dengel, A.
Knowledge Graph Enrichments for Credit Account Prediction.
DOI: 10.5220/0013181000003890
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 2*, pages 441-452
ISBN: 978-989-758-737-5; ISSN: 2184-433X
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

traceability have been identified as key requirements. While focusing on automatization, state-of-the-art literature does not consider these aspects as requirements in their approaches so far.

Therefore, a different perspective on account prediction research is proposed: Instead of focusing on the goal of full automatization, we propose focusing on the goal of assisting accountants in their daily decision-making for the problem of account assignment. For credit account prediction, knowledge graphs (Ehrlinger and Wöß, 2016) can be on the one hand useful to put prediction results as well as entities occurring in their explanations in the context of an existing accounting knowledge graph (Schulze et al., 2022), thus integrating them with other heterogenous accounting resources, and on the other hand useful to leverage the structure of knowledge graphs for machine learning approaches to predict accounts, such as link prediction as proposed for future work by Panichi and Lazzeri (Panichi and Lazzeri, 2023).

This paper aims to continue these lines of thought in related work and addresses therefore the following research questions: 1) how to predict credit accounts using knowledge graphs with approaches that can provide explanations and traceable predictions, and 2), how to improve predictions by enriching such knowledge graphs based on internal graph data available. Regarding explainability, the scope of the paper is to consider these requirements in the prediction approaches. Because the paper explores the feasibility and performance of such approaches, data-driven evaluation will be conducted. Therefore, the qualitative assessment on the daily usability of different explanation styles is beyond the paper's scope.

Section 2 presents related work on credit account prediction and Section 3 the pipeline to predict accounts. Section 4 presents the evaluation setup and evaluation results which are then compared and discussed in Section 5. Finally, Section 6 draws conclusions and gives an outlook for future work.

2 RELATED WORK

The research field of credit account prediction is getting more and more traction. One reason is the increasing adoption of electronic invoices (Koch, 2019), which is in Europe also politically enforced, and with that the availability of invoice data in a digitized form.

With the goal of automatization, Bardelli et al. (Bardelli et al., 2020) predicted accounts based on electronic invoices. Available to the authors were two datasets from accounting firms consisting of 32k and 34k invoice lines (not invoices) where 61% of invoice lines have been used for account prediction. The rest of the lines could not be assigned back to the respective entries in the account ledger. By considering account prediction for sent and received invoices, identifying the latter as the more difficult case, best results have been generated with decision-tree-based methods combined with Word2Vec (Mikolov et al., 2013).

Panichi and Lazzeri (Panichi and Lazzeri, 2023) focused on the scenario of account prediction on received invoices and presented a semi-supervised approach which manages to increase accuracy by 4% compared to the approach of Bardelli et al. (Bardelli et al., 2020) (with different data in a different setting). The performance increase could be achieved by applying the A* search algorithm (Hart et al., 1968), resulting in a loss of training data (because of the same linkage problem) of only 14% compared to 39% as in Bardelli et al. (Bardelli et al., 2020). Further characterizations of the data used in the experiments were not given except that it was much smaller than the data used in Bardelli et al. (Bardelli et al., 2020).

In contrast, Belskis et al. (Belskis et al., 2021; Belskis et al., 2020) built their approaches not on the basis of invoice data but on postings in ledger accounts. They employed natural language processing classification algorithms and showed that the use of comments in postings, which were manually created by accountants, increased the performance of precision by 2.56 percent (Belskis et al., 2021) compared to not using them (Belskis et al., 2020). The main goal was also automatization without the consideration of explainability.

Schulze et al. (Schulze et al., 2022) conducted in the context of a knowledge service for accountants credit account prediction as a side analysis. The approach was based on an accounting knowledge graph. However, it was restricted to the scope of food- and beverage-scenarios and required manually creation of vocabulary for this particular scope. Still, by learning decision trees, results indicated that knowledge graph enrichments may increase prediction performance.

Table 1 summarizes the comparison of related work. All the approaches use real-world data, which is important for applying insights generated in account prediction research. The downside is that results are not reproducible with the very same data because such data can reasonably not be published. Therefore, no public benchmarks exist (Panichi and Lazzeri, 2023). Given this situation, it is even more important to provide all resources to enable reproduction of presented approaches and evaluations on own invoice data. However, to the best of our knowledge, none of the related work does this so far.

Furthermore, all approaches shuffle the training

Table 1: Comparison of related work on credit account prediction (? = not explicitly mentioned).

	real world data	in-voice based	domain inde-pendent	number of invoices	number of accounts	consider explain-ability	source code public	time-series preserving evaluation
(Belskis et al., 2021)	yes	no	yes	-	?	no	no	no
(Bardelli et al., 2020)	yes	yes	yes	?, 66k inv. lines	130-166	no	no	no
(Panichi and Lazzeri, 2023)	yes	yes	yes	?, < 10k inv. lines	?	no	no	no
(Schulze et al., 2022)	yes	yes	no	1267	6	yes	no	no
(This paper)	yes	yes	yes	7k	161	yes	yes	yes

and test data. This can be problematical because invoices are time dependent and thus can be seen as time-series data as described in Bergmeir and Benítez (Bergmeir and Benítez, 2012). For example, an event such as a celebration may spawn particular types of invoices that may have not been received in such amounts in the training period when there was no similar celebration before. Even when having invoice data over longer periods, then still are business models and supplier relationships respect to change and with this also received invoices. Also, the accounts to predict are adapted over time.

Therefore, we argue that preserving the order of received invoices is important when evaluating predictive models. This also means that reported numbers in related work on predictive performance might be too optimistic. For example, on our data, we reach an accuracy increase of 16.41 percent points when not preserving the order but shuffling the data in cross-validation. Without preserving the order in evaluation, reported performance numbers in related work range from 74% to 89% for invoice-based approaches. However, comparison of prediction performances across approaches is difficult because of different used datasets and different numbers of accounts to predict. The unavailability of open-source code to use an account prediction approach as a baseline also hinders the comparison. Finally, related work focuses on automatization so that the aspects of explainability and traceability are not considered or discussed so far.

This paper aims to address these aspects by 1) focusing on explainable approaches by design using knowledge graphs, 2) preserving the order of invoices in evaluations, and 3) providing resources for applying the presented approach.

3 PIPELINE FOR CREDIT ACCOUNT PREDICTION

Figure 1 summarizes the overall process for account prediction with enriched knowledge graphs. Section 3.1 covers the construction of an accounting knowledge graph from invoice data (Step 1). Section 3.2 elaborates on how the knowledge graph is enriched with internal graph data (Step 2), and Section 3.3 elaborates on account prediction approaches (Step 3).

3.1 Accounting Knowledge Graph Construction from Invoice Data

3.1.1 Invoice Data and Postings

Data used in this study comes from a holding company located in Germany and covers basic information from received invoices and their related processes. To illustrate how we come from invoice data to account predictions with the proposed pipeline in Figure 1, we will use the following fictive example throughout the rest of the paper. The data processing steps with this example including inputs and outputs can also be followed in the demo folder on the paper page. Consider an accountant wants to process an invoice to assign credit accounts. The starting point is the invoice data already represented in an ERP system with which the accountant works, so that for the approach the data is available in the CSV format. This setting is comparable with experiment settings reported in Bardelli et al. (Bardelli et al., 2020) and Panichi and Lazzeri (Panichi and Lazzeri, 2023). For every invoice, the ERP system creates one process and attaches a process number to it. Therefore, whenever in the rest of the paper we speak of a process, we mean this process where data from one invoice is attached to. The booking area can be automatically derived from the recipient listed on the invoice. Let the example process number be `193224` and the booking area `7467`. Furthermore, our example invoice comes from `Chimney Sweep Mrs. Happy`, which is the invoice issuer. The service description on the invoice is `Maintenance service baker street 4`. This is all the data required from the received invoice. However, the approach requires also historical data of booked invoices in form of postings, which is also comparable to experiment settings in related work.

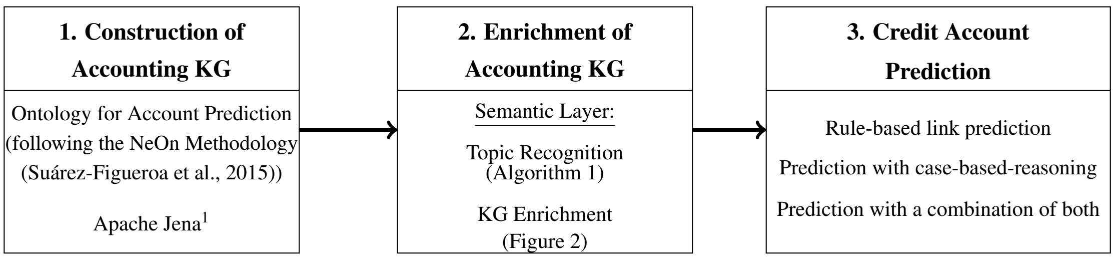
1. Construction of Accounting KG
Ontology for Account Prediction (following the NeOn Methodology (Suárez-Figueroa et al., 2015))
Apache Jena[1]
2. Enrichment of Accounting KG
Semantic Layer:
Topic Recognition (Algorithm 1)
KG Enrichment (Figure 2)
3. Credit Account Prediction
Rule-based link prediction
Prediction with case-based-reasoning
Prediction with a combination of both

Figure 1: Overview of the pipeline for credit account prediction based on enriched accounting knowledge graphs.

There may be multiple similar invoices, however, for illustration, for the rest of the paper consider the following fictive historic invoice with an assigned account (coming from the posting CSV). Process Number: `45564`, Booking Area: `9874`, Invoice Issuer: `Clean Chimney`, Service Description: `Maintenance Service Stand. TI`, Account: `5912`.

The total data used in the experiments amounts to 7k invoices with 12k postings which have been received in a couple of months within a year. In the scope of this study, the 7k invoices represent the usual scenario at the holding company where each invoice has one account as the solution. By following a long-tail distribution, 161 accounts could possibly be predicted. The service description typically consists of one to seven words in German, comes usually from a person who entered the service description manually, and often consists of organization- and business domain-specific vocabulary including abbreviations.

3.1.2 Knowledge Graph Construction

RDF (World Wide Web Consortium, 2014b) was used as the underlying data model of the knowledge graph, SPARQL (Valle and Ceri, 2011) was used as the query language, and OWL (Antoniou and van Harmelen, 2004) was used as the ontology language. This knowledge graph setup is in line with the notion of a knowledge graph provided by Ehrlinger and Wöß (Ehrlinger and Wöß, 2016), which also includes an ontology. For the ontology development, the NeOn Methodology Framework (Suárez-Figueroa et al., 2015) was followed which encompasses the phases of specifying requirements, identifying resources for the ontology, as well as restructuring and evaluating the ontology (Suárez-Figueroa et al., 2015). The main requirement for the ontology was to enable account prediction by downstream prediction approaches. Examples for 'competency questions' (Grüninger and Fox, 1995) serving this goal which also cannot be answered without such an ontology in place are: `Show all processes that have 5912 as an account and have at least one service description topic in common` or `Show all organizations that have 'Chimney' as an invoice issuer topic and where the processes(invoices) where these organizations are listed on as sellers have been booked on the account 5912.` For the ontology development, non-ontological resources were used. These were columns of the CSV file where invoice data was contained (invoice issuer, service description etc.), columns of the posting CSV file (e.g., account number), and headings in accounting handbooks (e.g., account number, short- and long-description). In the restructuring phase, existing vocabulary has been reused, for example the P2P-O ontology (Schulze et al., 2021) for electronic invoices and the Organization Ontology (World Wide Web Consortium, 2014a). Finally, the ontology was checked for consistency, and it was evaluated regarding pitfall patterns from the Oops! (OntOlogy Pitfall Scanner!) (Poveda-Villalón et al., 2014) (which resulted, for example, in more detailed license information). Also, the pre-formulated 'competency questions' (Grüninger and Fox, 1995) have been tested. The ontology and accompanying resources are available on our paper page for reuse.

Technically, by using this ontology, the accounting knowledge graph was constructed with the Java-based framework Apache Jena[1]. With our input data, the accounting knowledge graph had 102k triples. Demo CSVs of the input data are also available on the paper page to facilitate the preprocessing of own invoice data.

The black subgraphs a) and b) in Figure 2 illustrate an excerpt of the constructed knowledge graph with vocabulary from the developed ontology. Subgraph a) shows parts of the example process (invoice) at hand and subgraph b) relevant parts of a similar historic process, which also has a reference to the account `:a5912`. Note that in this excerpt the black subgraphs a) and b) are not connected. In fact, they are connected, for example, with the expression that the instances `:oChimneySweepMrsHappy` and `:oCleanChimney`

[1]https://jena.apache.org

are both of the type org:FormalOrganization. However, it is not shown in Figure 2 because every organization has this relation. Consequently, downstream prediction approaches cannot make use of this non-differentiating information.

3.2 Accounting Knowledge Graph Enrichment

The goal of the enrichment step is to create the kind of additional information in the knowledge graph that downstream prediction approaches can use to make better predictions. In the enrichment approach, such information will be topics about service descriptions of invoices and topics about invoice issuers. Furthermore, the enrichment approach should meet two major requirements: First, the requirements of explainability and traceability should also apply here to meet these requirements throughout the pipeline. Second, the enrichment process should be applicable across domains and across invoice topics. Consequently, the approach should be either unsupervised or semi-supervised in order to manage the wide-ranging topics of received invoices. The latter requirement is contrary to the requirements reported in the study conducted by Schulze et al. (Schulze et al., 2022). There, the domain was restricted to a food- and beverage-scenario so that domain-specific vocabulary could be created upfront. This is not feasible (or too costly) when not restricting to a specific domain within a particular company.

First experimentations have been done with applying classical string-based similarity measures on service descriptions which are implemented in the case-based reasoning framework ProCAKE (Bergmann et al., 2019), such as the Levenshtein-Distance (Levenshtein, 1966). Groups of similar processes were created which would then have been enriched to respective processes. However, analysis of such groups revealed that they had the tendency of being too big with containing many unsimilar processes. One reason may be that for this scenario, semantic similarity cannot be implied by the similarity of the raw strings. Also named entity recognition systems such as RoBERTa (Liu et al., 2019) have been explored; however, at least for German, only broad named entity labels are supported (person, organization, location and misc) which turned out not to be well suited for the very domain specific language in received business invoices.

The presented algorithm in Listing 1 for topic recognition and the enrichment step illustrated with the subgraphs c1) and c2) in Figure 2 are based on the idea of the isolated mapping similarity introduced in ProCAKE (Bergmann et al., 2019)[2]. However, instead of directly comparing tokens of, for example, two service descriptions with each other and then computing a similarity value, as in the isolated mapping similarity, tokens are first gathered (Algorithm 1) and then matched and enriched on the accounting knowledge graph (subgraphs c1) and c2) in Figure 2). In this way, the similarity calculation is pushed downstream to the account prediction approaches.

Algorithm 1 shows how topics are gathered from a set of given service description strings $SD(AKG)$ and invoice issuer names $II(AKG)$ from an accounting knowledge graph AKG. Service descriptions $sd \in SD(AKG)$ and invoice issuer names $ii \in II(AKG)$ are first tokenized (including bi- and trigrams). Next, they are added and counted into a set of tuples M_{SDT} (for service descriptions) and M_{IIT} (for invoice issuer names), where $M_{SDT} = \{\langle t, i\rangle : t \in T_{SD}, i \in \mathbb{N}$ with $i > 0\}$ where T_{SD} represents a set of tokens, and where $M_{IIT} = \{\langle t', i'\rangle : t' \in T_{II}, i' \in \mathbb{N}$ with $i' > 0\}$ where T_{II} also represents a set of tokens. However, a tuple $\langle t_{SD} \in T_{SD}, 1\rangle$ or a tuple $\langle t_{II} \in T_{II}, 1\rangle$ is only added when t_{SD} meets given quality criteria Q_{SD} and t_{II} quality criteria Q_{II}, which are given as an input. By exploring meaningful topics in our data, topics meet quality criteria Q_{SD} and Q_{II} when a) they have more than one character because two characters could already represent important business specific abbreviations, b) they have more than three digits when a topic consists only of numbers because this could be an identifier, and finally c), they are not a general language specific stop word. Finally, Algorithm 1 returns with M_{SDT} a set of counted topics for service descriptions and returns with M_{IIT} a set of counted topics for invoice issuer names. Considering the service descriptions of the two example processes (see also Figure 2), then topics in M_{SDT} would be {maintenance, service, baker, str, maintenance service, service baker, baker str, maintenance service baker, service baker str, stand., ti, service stand., stand. ti, service stand. ti} (an excerpt is shown in Figure 2), and topics in M_{IIT} would be {chimney, sweep, mrs.,happy, chimney sweep, sweep mrs., mrs. happy, chimney sweep mrs., sweep mrs. happy, clean, clean chimney}. An advantage of having such separate sets is that they can be maintained separately. For example, they can be sorted and further refined regarding inappropriate topics. Also, for experimentations with a fixed set, all topics that occur only once can be removed because they do not

[2]see https://procake.pages.gitlab.rlp.net/procake-wiki/sim/collections/

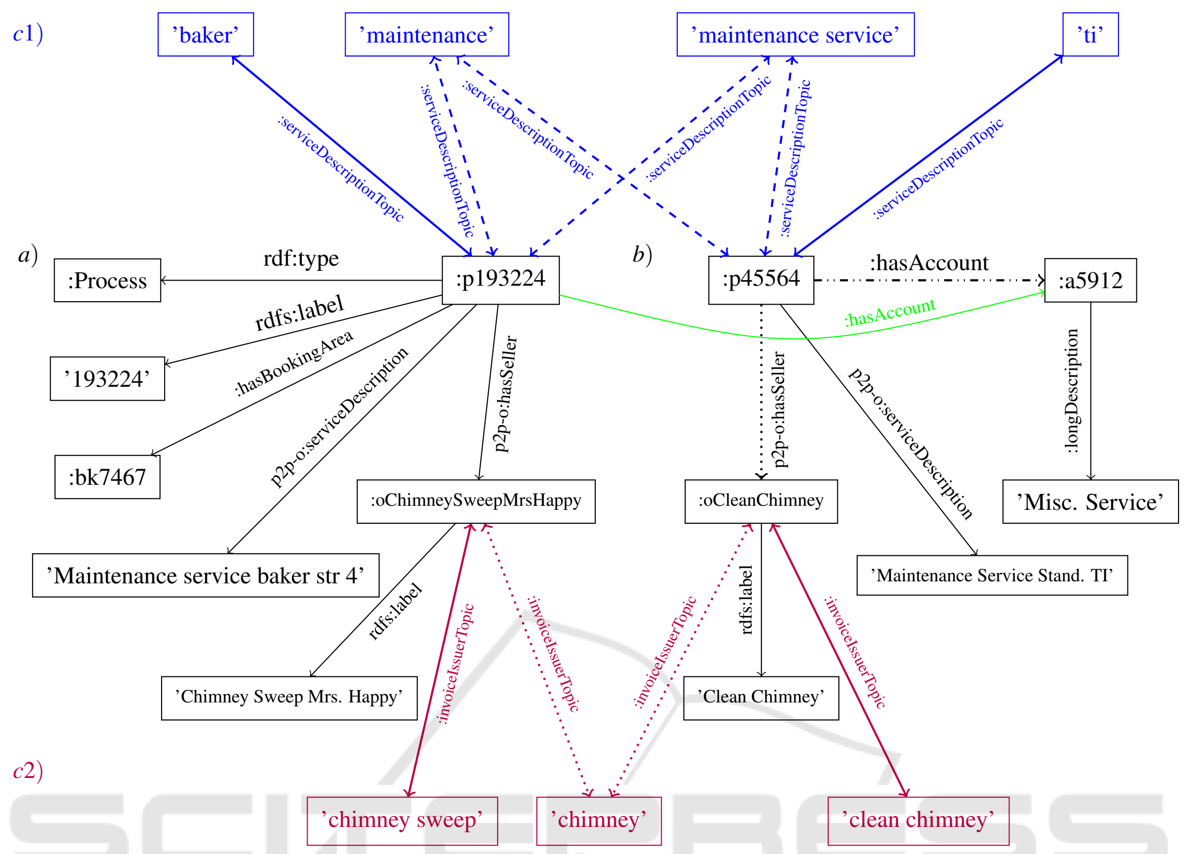

Figure 2: Excerpt of the accounting knowledge graph with fictive example data. The black subgraph a) represents the knowledge graph constructed from the example invoice data at hand, the black subgraph b) represents the knowledge graph constructed from a prior invoice which has a reference to an account, the blue subgraph c1) represents examples for service description enrichments, and the purple subgraph c2) represents examples for invoice issuer enrichments.

connect subgraphs.

The enrichment with the gathered topics from Algorithm 1 is illustrated with the subgraphs c1) and c2) in Figure 2: First, every service description string and invoice issuer name in the accounting knowledge graph is tokenized the same way as described in Algorithm 1. Next, if a token equals a topic in M_{SDT} or M_{IIT}, it is added to the knowledge graph with the predicate `:serviceDescriptionTopic` for service descriptions and `:invoiceIssuerTopic` for invoice issuer names. With our data, 6k of such triples have been enriched resulting in an enriched knowledge graph with 108k triples.

3.3 Account Prediction

As depicted in Figure 1, Section 3.3.1 describes account prediction using a rule-based link prediction (rule-based-LP) approach, Section 3.3.2 presents a case-based reasoning (CBR) approach, and Section 3.3.3 elaborates on how expert knowledge from a CBR System can be combined with rule-based-LP.

3.3.1 Account Prediction with Link Prediction

To enable the application of link prediction algorithms for account prediction, the problem of account prediction is first interpreted and formalized as a link prediction problem: Given is an accounting knowledge graph AKG_{Train}, a test knowledge graph KG_{Test}, a set of nodes N, a set of edges E, and the specific edge $\{:hasAccount\}$ that denotes which process has which account as the solution. Then, AKG_{Train} consists of a set of tuples $\langle n,e,n'\rangle$ where $n,n' \in N$, $e \in E$ and $\exists\langle n,e,n'\rangle \,|\, e = \{:hasAccount\}$. Further, $KG_{Test} = \{\langle n'',\{:hasAccount\},n'''\rangle : n'',n''' \in N\}$. Then, $AKG'_{Train} = AKG_{Train} \setminus KG_{Test}$. With AKG'_{Train}, the goal is to predict tuples of KG_{Test} so that $AKG_{Predicted} = AKG'_{Train} \cup KG_{Test}$.

The enrichment subgraphs c1) and c2) in Figure 2 enable new kinds of rules that rule learners can calculate. With enrichments from the prior step, the dashed and dotted edges can be leveraged by a rule learner for generating new rules which connect subgraphs a) and b). For example, in Figure 2, a

```
Data: SD(AKG),II(AKG), $Q_{SD}, Q_{II}$
Result: $M_{SDT}, M_{IIT}$
$T_{SD} = \emptyset$; $T_{II} = \emptyset$; $M_{SDT} = \emptyset$; $M_{IIT} = \emptyset$;
Service description strings:
foreach $sd \in SD(AKG)$ do
    $T_{SD} = tokenize(sd)$
    foreach $t_{SD} \in T_{SD}$ do
        if $Q_{SD}(t_{SD})$ then
            if $\exists\{\langle t,i\rangle\} \in M_{SDT}$ with $t = t_{SD}$
              then
                $\{\langle t,i\rangle\} = \{\langle t,i+1\rangle\}$
            else
                $M_{SDT} = M_{SDT} \cup \{\langle t_{SD},1\rangle\}$
            end
        end
    end
end
Invoice issuer names:
foreach $ii \in II(AKG)$ do
    $T_{II} = tokenize(ii)$
    foreach $t_{II} \in T_{II}$ do
        if $Q_{II}(t_{II})$ then
            if $\exists\{\langle t',i'\rangle\} \in M_{IIT}$ with $t' = t_{II}$
              then
                $\{\langle t',i'\rangle\} = \{\langle t',i'+1\rangle\}$
            else
                $M_{IIT} = M_{IIT} \cup \{\langle t_{II},1\rangle\}$
            end
        end
    end
end
```

Algorithm 1: Topic recognition given a set of service descriptions, invoice issuer names, and quality criteria.

simple rule can be: `:hasAccount(X,:a5912)` $\Longleftarrow$ `:serviceDescriptionTopic(X,maintenance)`. So, when a process X has the service description topic `maintenance`, it is implied that process X has the account `:a5912`, which is illustrated with the green edge in Figure 2. Imagine that this was the case in 9 out of 10 cases in AKG'_{Train}, then this particular rule would have a confidence value of .90.

To implement this formalization, the bottom-up rule learning algorithm of AnyBURL (Meilicke et al., 2024) was used, which will be more detailed in the Evaluation Setup Section 4.1.1. This state-of-the-art algorithm was chosen because of its good performance recently shown (Meilicke et al., 2024), and because it fulfills the requirements of explainability by generating traceable rules with a symbolic approach.

3.3.2 Account Prediction with CBR

Because historic invoices can be seen as solved cases, and because accountants often use historic cases as the basis for a solution for a current task, case-based reasoning (CBR) approaches (Kolodner, 2014) can provide account predictions based on historic cases where similarity functions can be used for explaining and tracing back such predictions. As a CBR framework, we used ProCAKE (Bergmann et al., 2019) because this framework offers with its focus on processes a suitable integration into the process-oriented accounting domain. A process (with invoice data) is seen as a case which is defined in an aggregate class with the following set of attributes (which is denoted as A): a process number $p \in A$, an invoice issuer name $iin \in A$, a booking area $ba \in A$, an invoice service description $sd \in A$, a set of enriched service description topics $SD_{enriched} \in A$, a set of enriched invoice issuer topics $IIN_{enriched} \in A$, and an account solution $acc \in A$. Considering local calculated similarities between attributes of two cases $s_{local}(iin, iin') \in \mathbb{R}$, $s_{local}(sd, sd') \in \mathbb{R}$, …, the aggregate similarity function $s_{aggregate} : A \to \mathbb{R}$ was defined as follows:

$$
\begin{gathered}
s_{aggregate}(s_{local}(iin,iin'), s_{local}(sd,sd'), s_{local}(ba,ba'),\\
s_{local}(SD_{enriched}, SD'_{enriched}),\\
s_{local}(IIN_{enriched}, IIN'_{enriched}), w_{1,2\ldots5} \in \mathbb{R} : 0 \leq\\
w_{1,2\ldots5} \leq 3)\\
\mapsto\\
(s_{local}(iin,iin') \times w_1) + (s_{local}(sd,sd') \times w_2) +\\
(s_{local}(ba,ba') \times w_3)\\
+ (s_{local}(IIN_{enriched}, IIN'_{enriched}) \times w_4) +\\
(s_{local}(SD_{enriched}, SD'_{enriched}) \times w_5)
\end{gathered}
$$

Note that the attributes process number p and account solution acc are not part of $S_{aggregate}$. This is because in the former case, it has no inherent information value, and in the latter case, this is the attribute to be predicted. However, when retrieving similar cases, these attributes are of high value for the accountant. Local similarity calculations of $s_{local}(iin, iin')$, $s_{local}(sd, sd')$ and $s_{local}(ba, ba')$ were conducted using the string-equals similarity measure in ProCAKE (Bergmann et al., 2019), and calculations of $s_{local}(SD_{enriched}, SD'_{enriched})$ and $s_{local}(IIN_{enriched}, IIN'_{enriched})$ using ProCAKES's (Bergmann et al., 2019) isolated mapping similarity measure. Concrete weights $w_{1,2\ldots5} \in \mathbb{R}$ were found by applying hyperparameter optimization with 'random search' (Rastrigin, 1963), which will be covered in more detail in the Evaluation Setup Section 4.1.2. With respect to the example processes, by following this approach, the attributes (invoice issuer name, service descriptions (as well their topics), etc.) will be compared as reported, and then an aggregate similarity value will be calculated by applying the weights from $S_{aggregate}$. This similarity value indicates the similarity between the process at hand (here

`:p193224`) and the historic process `:p45564`. These calculations are conducted for every historic process in the knowledge graph, and the accounts of the most similar processes (here e.g. `:a5912`) are predicted according to their aggregate similarity values.

3.3.3 Account Prediction with the Combination of Expert Knowledge from CBR and Rule-Based Link Prediction

A strength of rule-learners is that they are unsupervised without hyperparameter optimization required in case of AnyBURL (Meilicke et al., 2024). A strength of CBR is the consideration of human-in-the loop in the refinement phase bringing additional expert knowledge in (Kolodner, 2014). The basic idea of combining these approaches is, on the one hand, to make use of the predictive performance of rule-learners, and on the other hand, to refine such predictions with expert knowledge from a CBR system.

Subsequently, in a refinement phase, the target accounts where additional expert knowledge from CBR should be tailored to, are the accounts where AnyBURL (Meilicke et al., 2024) underperforms. For each such account, the basis for analysis is, on the one hand, the long description of the account in accounting handbooks, and on the other hand, the reachable closure of all invoices in the training knowledge graph AKG'_{Train} that should have yielded the correct tuple in KG_{Test}. On this basis, the expert defines additional topics M'_{SDT} and M'_{IIT} as well as a respective case which can be used for deriving the right account solution. When now the service description or invoice issuer name contains such topics in M'_{SDT} and M'_{IIT}, the respective case and account solution is retrieved and ranks higher than the predictions calculated with $S_{aggregate}$. Further, in the course of account prediction with AnyBURL (Meilicke et al., 2024), when expert knowledge is available, then the account solution coming from the CBR system is put at the first position of the suggested predictions, and all other predictions of AnyBURL (Meilicke et al., 2024) are moved down by one position.

Imagine that in the example service description `Maintenance Service Stand. TI`, the `TI` stands for an abbreviation well-known within the organization meaning a particular type of maintenance (e.g., with dust measurement) which is only typical for the account `:a5912`. When this information is added in the refinement phase of a CBR system, and when `TI` occurs in a service description at hand, then `:a5912` is suggested as the most likely account followed by the predictions of AnyBURL.

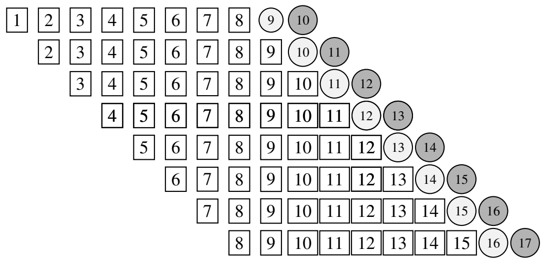

Figure 3: Illustration of how 17 folds have been used for cross-fold validation inspired by Bergmeir and Benítez (Bergmeir and Benítez, 2012) considering the time-series nature of invoice data where rectangles represent folds for training, light circles folds for validation, and darker filled circles folds for testing, resulting in eight datasets.

4 EVALUATION

Section 4.1 demonstrates the applied evaluation setup and developed resources which make the different approaches comparable. Section 4.2 reports computing performances, and Section 4.3 presents the results.

4.1 Evaluation Setup

Although rule-based-LP and CBR do not require mass data for learning, 7k invoices and the resulting 108k triples in the knowledge graph are still a medium-sized sample for evaluation purposes. Therefore, in line with best-practices in related work (e.g.(Panichi and Lazzeri, 2023)), cross-fold validation was applied. However, as introduced in Related Work, invoices can relate to each other and can be time dependent. Therefore, we argue to preserve the received order of invoices in evaluations, also within the folds. Figure 3 demonstrates in more detail how 17 folds of the dataset have been used for training, validation, and testing. As also depicted, this procedure resulted in eight training-, valid- and test-datasets.

The code to enable cross-fold validation on invoice data where the order is preserved and the output is consumable by AnyBURL (Meilicke et al., 2024) and ProCAKE (Bergmann et al., 2019) is published on the paper page. To enable comparative evaluation, it is also considered that training-, valid- and test-datasets can be built with different degrees of semantic enrichment: 1) without any enrichment, 2) with invoice issuer name enrichment, 3) with service description enrichment, and 4) with both, invoice issuer name and service description enrichment (denoted as full enrichment for short).

4.1.1 Evaluation Setup AnyBURL

Accordingly, with AnyBURL (Meilicke et al., 2024), 32 (4×8) models have been trained which then have been evaluated on the 32 test datasets. Because AnyBURL does not require any hyperparameter optimization (Meilicke et al., 2024), and in line with the original AnyBURL evaluation (Meilicke et al., 2019), the valid datasets were also not used here. For experiments, the latest version 2023 was applied[3]. Rules have been learnt for 100 seconds because longer time spans did not result in increased performance. All default parameters have been applied except for the maximum length of cyclic and acyclic rules. This parameter was increased from 2 to 3 because enrichments are this one hop deeper in the graph which otherwise would not have been considered by the rule learner. Furthermore, because we only want to predict `:hasAccount` relations, rules have been learnt for this single relation. Finally, rules have been applied to predict hits @1 to hits @10 for each tuple $\langle n'', \{:hasAccount\}, n''' \rangle$ in KG_{Test}.

4.1.2 Evaluation Setup ProCAKE

For ProCAKE (Bergmann et al., 2019) however, the valid datasets have been used for optimizing hyperparameters. Here, hyperparameters are the weights $w_{1,2...5} \in \mathbb{R}$ of the similarity function $S_{aggregate}$ as described in Section 3.3.2. For optimization, 'random search' (Rastrigin, 1963) was applied where for each training set models have been learnt and then have been evaluated on the respective valid dataset. The hyperparameter configuration with the best performance on the valid dataset was then used for evaluating the trained model on the test dataset. Depending on the enrichment level, weights of other enrichments have been set to zero. Account solutions of the top 1 to top 10 most similar retrieved cases have then been used to calculate hits @1 to hits @10.

4.1.3 Evaluation Setup Combination of ProCAKE and AnyBURL

The evaluation setup of the approach where expert knowledge from the CBR system is combined with AnyBURL (Meilicke et al., 2024) was conducted as follows. First, with custom evaluation code, the performance of the AnyBURL-model of every fold was evaluated on the valid dataset so that the performance for each account was evaluated separately. Then, the top 5 most frequently underperforming accounts have been considered for increasing the performance by integrating expert knowledge into ProCAKE (Bergmann et al., 2019). This was done with the procedure described in Section 3.3.3. Next, CBR with additional expert knowledge was iteratively evaluated on the valid datasets, and with the final set of expert knowledge evaluated on the test datasets. Finally, the predictions of AnyBURL on the test datasets have been evaluated again but with consideration of expert knowledge coming from the CBR system. Because we were interested whether the combined approach is on the one hand feasible and on the other hand is able to increase the overall performance, and because of higher manual effort in providing expert knowledge, the presented procedure was conducted on the fully enriched scenario but on all eight training,- valid- and test-datasets. Also with this procedure, hits @1 to hits @10 have been calculated.

[3]https://web.informatik.uni-mannheim.de/AnyBURL/

4.2 Performance

All experiments were conducted with a laptop having an Intel(R) Core(TM) i7-10750H CPU @2.60GHz and 32GB RAM. As reported, rules have been learnt by AnyBURL for 100 seconds. Applying the learnt models which contained 34420 rules on average on one test fold, which contained ca. 400 `:hasAccount` relations, took on average 37211ms (full enrichment scenario). For CBR, applying models with concrete weights on one fold took on average 1257ms (also full enrichment scenario). Enriching the whole accounting knowledge graph with service description topics (as illustrated in Figure 2) took 3957ms and with invoice issuer topics 775ms. Applying Algorithm 1 for topic extraction took 401ms for the whole knowledge graph (351ms for service descriptions and 50ms for invoice issuer names).

4.3 Evaluation Results

Table 2 summarizes the evaluation results. As shown in Figure 3, eight models have been learnt and validated for each level of semantic enrichment (32 models AnyBURL (Meilicke et al., 2024) and 32 models ProCAKE (Bergmann et al., 2019)). Accordingly, means for hits @1 to hits @5 and hits @10 are depicted. For the full enrichment scenario, results for including expert knowledge from ProCAKE into AnyBURL are depicted as well as the increase of percent points when using full enrichments compared to no enrichments.

Table 2: Evaluation results for credit account prediction with four different degrees of semantic enrichment.

	hits@1	hits@2	hits@3	hits@4	hits@5	hits@10
Without Enrichment						
ProCAKE (Bergmann et al., 2019)	.4143	.4913	.5362	.5627	.5970	.6690
standard deviation	.0686	.0561	.0464	.0470	.0428	.0380
AnyBURL (Meilicke et al., 2024)	.4827	.5789	.6277	.6621	.6863	.7877
standard deviation	.0284	.0291	.0379	.0361	.0376	.0403
Invoice Issuer Enrichment						
ProCAKE (Bergmann et al., 2019)	.4722	.5538	.5910	.6200	.6477	.7200
standard deviation	.0660	.0544	.0498	.0508	.0451	.0329
AnyBURL (Meilicke et al., 2024)	.5027	.6079	.6579	.6885	.7118	.8062
standard deviation	.0414	.0306	.0329	.0435	.0487	.0435
Service Description Enrichment						
ProCAKE (Bergmann et al., 2019)	.5111	.5802	.6193	.6445	.6652	.7270
standard deviation	.0516	.0455	.0465	.0444	.0446	.0408
AnyBURL (Meilicke et al., 2024)	.5273	.6321	.6803	.7083	.7313	.8087
standard deviation	.0368	.0438	.0448	.0478	.0486	.0419
Invoice Issuer Enrichment and Service Description Enrichment						
ProCAKE (Bergmann et al., 2019)	.5387	.6145	.6464	.6786	.6977	.7528
standard deviation	.0518	.0413	.0439	.0405	.0383	.0360
AnyBURL (Meilicke et al., 2024)	.5333	.6462	.6956	.7243	.7446	.8231
standard deviation	.0382	.0428	.0448	.0511	.0525	.0445
Expert Knowledge from ProCAKE (Bergmann et al., 2019) combined with AnyBURL (Meilicke et al., 2024)	.5400	.6522	.7013	.7300	.7504	.8285
Δ percent points between no to full enrichment for ProCAKE (Bergmann et al., 2019)	12.45%	12.32%	11.02%	11.59%	10.07%	8.38%
Δ percent points between no to full enrichment for AnyBURL (Meilicke et al., 2024)	5.06%	6.73%	6.79%	6.21%	5.83%	3.54%

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

5 DISCUSSION

5.1 Comparison Between no and Full Enrichment

According to Table 2, one key-finding is that for both, ProCAKE (Bergmann et al., 2019) and AnyBURL (Meilicke et al., 2024), applying both enrichments increases the performance for hits @1 to hits @5 compared to not having enrichments available. Overall, the effect of using both enrichments is for ProCAKE double as high as for AnyBURL. This may be due to the comparable low performance of the CBR System when having no enrichments available, and the better capability of the rule learner to deal with less useful information by generating still more fine-grained rules. By looking at concrete predictions generated by ProCAKE and AnyBURL in the no-enrichment scenario, it is observable that big groups of three to six accounts with the same confidence values are predicted. When enrichments come in, these big groups disappear, and both approaches successfully handle to make more fine-grained distinctions. This may also explain why the effect of enrichments is less for hits @10 compared to hits @1 to hits @5. However, considering our rationale of assisting accountants by suggesting a range of predictions, especially hits @3 to hits @5 are relevant.

5.2 Comparison Between Invoice Issuer and Service Description Enrichment

As depicted in Table 2, another finding is that for both, CBR and AnyBURL (Meilicke et al., 2024), also single enrichments of invoice issuer names and service descriptions result in a performance increase. This indicates that these enrichments on their own are useful. However, it is notable that service description enrichments have a bigger impact on the performance increase than invoice issuer enrichments. This is interesting because during hyperparameter optimization for the similarity function $S_{aggregate}$ in CBR (see Section 4.1), we observed that when having no enrichments available, then the invoice issuer name was typically weighted higher than the service description, which implies that in this case the invoice issuer name was more useful. However, in line with the observed results, when having both enrichments available, then the service description enrichments typically have been weighted higher than invoice issuer enrichments. The reason may lie on the one hand in the higher variance of service description strings compared to invoice issuer names, which implies that service descriptions can be better used for making distinctions, and on the other hand, in the higher number of tokens in service descriptions which results in more enrichments. This indicates that when applying the enrichments on invoice data, the usefulness may be dependent on the variance of the set of strings, which is typically higher for service descriptions and less for invoice issuer names. In our data, there are twice as many distinct service descriptions as invoice issuer names. To conclude, considering the special characteristics of such strings described in Section 3.1, results in Table 2 indicate that in such cases knowledge graph enrichments may have an impact on predictive performance.

5.3 Combining Expert Knowledge from CBR with Rule-Based LP

By following the procedure described in Section 3.3.3, including expert knowledge in CBR and combining this with AnyBURL (Meilicke et al., 2024) yields overall the best results. With this, one finding is that it is feasible to enhance rule-based-LP by including expert knowledge in a way so that intended account solutions are returned, and at the same time false positives can be mostly avoided.

The following extensions may be beneficial to increase the performance on KG_{Test} further. First, the integration of more fine-grained expert knowledge which is contained in accounting handbooks. In this way, more expert knowledge would be available which may trigger on unseen invoicing cases. Second, the consideration of not only top 5 underperforming accounts but more. Analogously, more accounts would be covered by expert knowledge which increases the chance of coverage on unseen data. We restricted to top 5 underperforming accounts because for cross-fold evaluation, the procedure of bringing in expert knowledge needed to be manually executed 40 times (5×8 because of top five underperforming accounts and eight training-, valid-, and test- datasets). Third, the consideration of more dimensions from the invoice data, such as individual invoice position lines, in order to use them as information for applying expert knowledge.

6 CONCLUSION AND OUTLOOK

The research questions of this paper were how to predict credit accounts using knowledge graphs with approaches that can provide explanations and traceable predictions, and how to improve such predictions by enriching accounting knowledge graphs with internal data. With a three main-step pipeline, it was shown how accounting knowledge graphs can be constructed and leveraged for predicting accounts by using rule-based link prediction, case-based reasoning, and a combination of both. Such approaches have in common that they can explain and trace back predictions. Furthermore, it was shown how accounting knowledge graphs can be enriched from internal data already available in the knowledge graph so that prediction performance increases. Regarding evaluation, it was highlighted to preserve the order of received invoices: on the one hand because of the time-series nature of invoices, and on the other hand, to gain more realistic insights when aiming to deploy such a system. Furthermore, to apply the pipeline and the evaluation on own invoice data, all resources are available at our paper page. Because related work did not publish source code of their approaches so far, with this contribution it is now possible to use the provided resources as a first baseline for future work. The usage of basic invoice information and with that little preprocessing necessary may facilitate this.

For future work, research on the perceived usefulness of different explanation styles provided by CBR and rule-based-LP approaches can be fruitful. This is particularly interesting considering the findings that CBR and rule-based-LP produce similar prediction results when having both enrichments available. Also, while explanations in form of similar cases and corresponding similarity values are in the context of CBR conceptually rather straight forward, in the context of rule learners it poses another challenge to aggregate and represent a set of rules from a prediction in a way so that they are meaningful and useful for daily knowledge work.

ACKNOWLEDGEMENTS

We would like to thank our business partner for the fruitful cooperation and all the support. We thank Christian Zeyen for CBR-related discussions and valuable feedback. We also thank Mahta Bakhshizadeh, Desiree Heim, Christian Jilek, Heiko Maus, and Markus Schröder for fruitful discussions and inspiration. Finally, we would like to thank anonymous reviewers for their valuable feedback.

REFERENCES

Antoniou, G. and van Harmelen, F. (2004). Web ontology language: OWL. In *Handbook on Ontologies*, International Handbooks on Information Systems, pages 67–92. Springer.

Bardelli, C., Rondinelli, A., Vecchio, R., and Figini, S. (2020). Automatic electronic invoice classification using machine learning models. *Mach. Learn. Knowl. Extr.*, 2(4):617–629.

Belskis, Z., Zirne, M., and Pinnis, M. (2020). Features and methods for automatic posting account classification. In *Databases and Information Systems - 14th Intern. Baltic Conf.*, volume 1243 of *Communications in Computer and Information Science*, pages 68–81. Springer.

Belskis, Z., Zirne, M., Slaidins, V., and Pinnis, M. (2021). Natural language based posting account classification. *Balt. J. Mod. Comput.*, 9(2).

Bergmann, R., Grumbach, L., Malburg, L., and Zeyen, C. (2019). Procake: A process-oriented case-based reasoning framework. In *Workshops Proc. for (ICCBR 2019), Otzenhausen, Germany, September 8-12, 2019*, volume 2567 of *CEUR Workshop Proceedings*, pages 156–161. CEUR-WS.org.

Bergmeir, C. and Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Inf. Sci.*, 191:192–213.

Ehrlinger, L. and Wöß, W. (2016). Towards a definition of knowledge graphs. In *Joint Proc. of the Posters and Demos Track of SEMANTiCS2016, Leipzig, Germany, September 12-15, 2016*, volume 1695 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Grüninger, M. and Fox, M. S. (1995). The role of competency questions in enterprise engineering. pages 22–31. Springer US, Boston, MA.

Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.*, 4(2):100–107.

Jain, K. and Woodcock, E. (2017). A road map for digitizing source-to-pay. McKinsey.

Koch, B. (2019). The e-invoicing journey 2019-2025. billentis GmbH.

Kolodner, J. L. (2014). *Case-Based Reasoning*. Morgan Kaufmann.

Lazar, J., Feng, J., and Hochheiser, H. (2017). *Research Methods in Human-Computer Interaction, 2nd Edition*. Morgan Kaufmann.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

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. *CoRR*, abs/1907.11692.

Meilicke, C., Chekol, M. W., Betz, P., Fink, M., and Stuckenschmidt, H. (2024). Anytime bottom-up rule learning for large-scale knowledge graph completion. *VLDB J.*, 33(1):131–161.

Meilicke, C., Chekol, M. W., Ruffinelli, D., and Stuckenschmidt, H. (2019). Anytime bottom-up rule learning for knowledge graph completion. In *Proceedings of IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3137–3143. ijcai.org.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proc. of 27th Annual Conference on Neural Information Processing Systems 2013. Lake Tahoe, Nevada, United States*, pages 3111–3119.

Panichi, B. and Lazzeri, A. (2023). Semi-supervised classification with a*: A case study on electronic invoicing. *Big Data Cogn. Comput.*, 7(3):155.

Poveda-Villalón, M., Gómez-Pérez, A., and Suárez-Figueroa, M. C. (2014). Oops! (ontology pitfall scanner!): An on-line tool for ontology evaluation. *Int. J. Semantic Web Inf. Syst.*, 10(2):7–34.

Rastrigin, L. A. (1963). The convergence of the random search method in the extremal control of a many parameter system. *Automaton & Remote Control*, 24:1337–1342.

Schulze, M., Pelzer, M., Schröder, M., Jilek, C., Maus, H., and Dengel, A. (2022). Towards knowledge graph based services in accounting use cases. In *Proceedings of Poster and Demo Track of SEMANTiCS 2022, Vienna, Austria, September 13th to 15th, 2022*, volume 3235 of *CEUR*. CEUR-WS.org.

Schulze, M., Schröder, M., Jilek, C., Albers, T., Maus, H., and Dengel, A. (2021). P2P-O: A purchase-to-pay ontology for enabling semantic invoices. In *The Semantic Web, ESWC 2021, Virt. Event, June 6-10, 2021, Proc.*, volume 12731 of *LNCS*, pages 647–663. Springer.

Suárez-Figueroa, M. C., Gómez-Pérez, A., and Fernández-López, M. (2015). The neon methodology framework: A scenario-based methodology for ontology development. *Appl. Ontology*, 10(2):107–145.

Valle, E. D. and Ceri, S. (2011). Querying the semantic web: SPARQL. In *Handbook of Semantic Web Technologies*, pages 299–363. Springer.

World Wide Web Consortium (2014a). The organization ontology, http://www.w3.org/tr/vocab-org/.

World Wide Web Consortium (2014b). Rdf 1.1 primer, http://www.w3.org/tr/2014/note-rdf11-primer-20140624/.