HPI Hasso Plattner Institut
IT Systems Engineering
Universität Potsdam

Bachelor Thesis

Evaluation of Entity Linking Models on Business Data

Evaluierung von Modellen für die Entitätsverknüpfung von Unternehmensdaten

Jan Ehmüller
jan@ehmueller.de

July 21, 2017
Information Systems Group
Hasso Plattner Institute
University of Potsdam, Germany

Supervisors
Prof. Dr. Felix Naumann
Toni Grütze
Michael Loster

Abstract

To create a graph of businesses and their relations with each other one can use structured knowledge bases, like Wikidata and DBpedia. But these knowledge bases contain incomplete information about the relations of businesses. To find relations not contained in these knowledge bases they must be extracted from unstructured texts like newspaper articles. To do this, we must first recognize businesses in unstructured texts and link them to entities in our knowledge base.

This thesis proposes a concept that combines both Named Entity Recognition and Entity Linking of businesses in German texts. It uses only publicly available data, i.e., the German Wikipedia and Wikidata, to train a classification model that annotates and links unstructured documents to entities in our knowledge base.

Our approach annotates documents in an efficient way and with a high reliability. With it, we are able to annotate millions of documents in an acceptable amount of time and can, therefore, annotate newly published newspaper articles in real-time. This enables us to update our knowledge base and the business graph with up-to-date relations.

Zusammenfassung

Um einen Graphen von Unternehmen und ihren Beziehungen untereinander zu erstellen, kann man strukturierte Wissensbasen, wie z.B. Wikidata oder DBpedia, benutzen. Diese enthalten aber keine vollständigen Informationen über die Beziehungen von Unternehmen. Um die nicht enthaltenen Beziehungen zu finden, muss man sie aus unstrukturierten Texten wie Zeitungsartikeln extrahieren. Um dies zu tun müssen wir erst Vorkommen von Unternehmen in unstrukturierten Texten finden und diese zu Entitäten in unserer Wissensbasis verlinken.

Die vorliegende Bachelorarbeit stellt ein Konzept vor, dass sowohl die Named Entity Recognition als auch das Entity Linking von Unternehmen in deutschen Texten kombiniert. Es benutzt nur öffentlich zugängliche Daten, Wikipedia und Wikidata, um ein Klassifikationsmodell zu trainieren, dass unstrukturierte Dokumente annotiert und zu Unternehmen in unserer Wissensbasis verlinkt.

Unser Ansatz annotiert die Dokumente effizient und mit einer hohen Zuverlässigkeit. Damit ist es möglich Millionen von Dokumenten in einem akzeptablen Zeitraum zu annotieren. Dadurch kann man neu veröffentlichte Zeitungen in Echtzeit annotieren und die Beziehungen zwischen Unternehmen extrahieren. Dies ermöglicht es uns, unsere Wissensbasis und unseren Unternehmensgraphen aktuell zu halten.

Contents

Introduction[1]

When it comes to economic decisions, uncertainty is a critical issue. Following the approach of the rational choice theory, every market player is constantly trying to maximize his utility and minimize his effort. Uncertainty can be described as a lack of information of how a market - or herein the full German economic system - is constituted and about the future behavior of the market players. Presuming that every market player is acting on a rational basis, all information regarding his situation, resources, plans, and relations makes the results of his decisions more predictable. In this manner, we can state: The more relevant information a market player gathers about other players in the market or economy, the better the foundation of his decisions is. The broad range of that kind of information can lead to a significant competitive advantage. So it should be in a rational player's interest to collect as much relevant information as possible.

In a connected economy, a lot of those uncertainties lie in the relations between corporations[2]. This became evident in the so called *Abgas-Skandal* or *Dieselgate* of the Volkswagen AG in 2015, wherein a lot of external suppliers spun out of control, from a financial perspective [5, 1]. This happened even though most of the suppliers did not take part in the scandal itself. Since there are a lot of other examples like the *Lehmann Brothers bankruptcy* or any other economic shock event, we can state that relations are a significant factor in the economic evaluation of corporations and their financial risks.

The German Corporate Graph Project

Because there are millions of corporations in the German economy[3] and each corporation potentially holds relations to hundreds or thousands of other corporations, collecting and overseeing all those relations becomes a complicated matter. The *German Corporate Graph Project* is one approach to solve this problem. The project's purpose is to extract business entities from multiple structured knowledge bases (e.g., Wikidata and DBpedia), merge them, enrich them with relations extracted from unstructured documents and finally display the graph so that it can be visually explored.

The project consists of a pipeline, which starts with the import and normalization of structured knowledge bases. The next step is the Deduplication, which is the detection and fusion of occurrences of the same entity over multiple knowledge bases. These entities form a graph, whose nodes are businesses and whose edges are the relations between them. This graph is then enriched during the Information Extraction. In this step relations between entities are extracted from unstructured documents using Named Entity Recognition, Entity Linking and Relation Extraction.

[1] This section was authored by Matthias Radscheit [11].

[2] We define as *corporation* any juristic entity that takes part in the German economy. This includes especially businesses but also other entities like public corporations.

[3] The *Federal Bureau of Statistics* notes 3,469,039 businesses in Germany in 2015 [4]. Following our definition of corporations, this number has to be seen as a lower bound for the total number of corporations in Germany.

The results of all these steps can be viewed and curated in the so-called Curation Interface. This is a web-interface, which can be used to control the pipeline itself, view statistical data generated by other pipeline steps and to view and curate the entities and relations of the graph itself. The final graph can be visually explored by using the Corporate Landscape Explorer, which is a web-interface as well.

One Project - Seven Contributions

This thesis is published in the context of a bachelor's project in 2016/2017 at Hasso-Plattner-Institute in Potsdam, Germany. The project's objective was to build the *German Corporate Graph*, as described above, to display Germany's corporate landscape. The project lasted ten months and was accompanied by Commerzbank AG, Germany.

As a result, the following theses were published. Strelow describes the used data model for businesses and their relations with respect to working with Apache Cassandra [19]. Löper and Radscheit evaluate the duplicate detection during the Deduplication [11]. Pabst explores efficient blocking strategies, which are used to increase the performance of the Deduplication [14]. Janetzki explains the creation of our knowledge base and the extraction of features used for the Named Entity Recognition and Entity Linking [9]. This work evaluates the quality of different classification models and of the features used to train them. Schneider evaluates different Relation Extraction methods [17]. Gruner describes methods to extract useful knowledge from the generated business graph [7].

1 Ambiguous Business Aliases

The graph of Germany's corporate landscape contains nodes, which are businesses, and edges, which are the relations between businesses. Information about businesses is extracted from structured knowledge bases such as Wikidata and DBpedia. They describe the businesses themselves but contain incomplete information about the relations between them. These relations are described in unstructured texts like Wikipedia or newspaper articles. They need to be extracted from these unstructured texts to be transformed into an edge of the graph. This extraction needs to be done in an efficient way because there are hundreds of thousands of newspaper articles and other text sources containing information about business relations. To keep the Graph up-to-date, it needs to be possible to extract relations from new newspaper articles in real-time.

For our approach to extract a relation between two businesses, they must first be mentioned in the same sentence. Then both of these mentions need to be found and linked to the entities representing these businesses. Finding these mentions is called *Named Entity Recognition* (NER). They must then be linked to the correct entities in the knowledge base. This step is called *Entity Linking* (EL). The last step is to extract the relation between the two businesses from the sentence, which is called *Relation Extraction* (RE). Our approach combines NER and EL into a single step and transforms it into a classification problem. These resulting two steps, the combination of NER and EL and the RE step, make up the Information Extraction step in the pipeline described in the introduction.

Section 2 covers related work in the field of both NER and EL. Section 3 describes the used approach that combines both NER and EL. After that, different classification models and configurations are compared and evaluated in Section 4. Section 5 evaluates the features used to train the classifier. These features are described in detail by Janetzki [9]. Section 6 discusses the performance and quality of this approach when annotating millions of newspaper articles. Finally, Section 7 concludes this thesis and describes possible improvements that could be done in the future.

2 Related Work

Both NER and EL are widely known and researched problems. Recognizing businesses in texts was one of the first research topics for NER according to Nadeau et al. [13]. This approach by Rau uses rules and heuristics to recognize businesses in texts [15]. Mikheev et al. use a similar approach, also using rules and heuristics and not needing gazetteers, and extend the set of recognized entities by persons and locations [12]. Ritter et al. recognize named entities in Tweets by using part-of-speech tagging and chunking [16]. Approaches to NER in the German language are, e.g., done by Blessing et al., who recognize and ground German geographic proper names by using a three-step model consisting of spotting, typing and referencing [2]. There are also approaches for NER on businesses in German texts. One such approach by Loster et al. uses various dictionaries, regular expressions and text contexts to recognize businesses [10].

There are also many research efforts solving the EL problem in various ways. Some combine the EL with the NER while others assume that the entities were already recognized and tagged. Sil et al. present an approach that combines EL with NER by combining candidate mentions from NER systems and candidate entity links from EL systems and making joint predictions [18]. Brauer et al. also combine NER and EL [3]. They recognize and associate entities in unstructured data with those in structured data. To do so, they disambiguate mappings of entities in the text to entities in the structured data by exploiting relationships from the structured data and the documents' structure. Grütze et al. propose a system that does only the EL task and assumes the NER to be already done [8]. Their focus lies on the reliability and performance of the Entity Linking, similar to the approach this thesis proposes. They annotate millions of documents in an acceptable amount of time and focus on the precision of their EL as well. Additionally, they improve the recall by using a second classifier with a high recall and combining the candidates of both with Random Walks.

3 Named Entity Recognition and Linking

Given an input text, all mentions of businesses should be recognized and linked to the correct business while other mentions of named entities should be ignored. Quote 1 shows two example sentences about automobile businesses. Only the bold named entities are businesses and should be linked to entities in the knowledge base. The underlined entities are also named entities but not businesses. In this case, they are persons, but they could be organizations, like the EU, as well. Since the entities are used to extract relationships between businesses, the non-business entities must be filtered in some way. Finally, the business entities must be linked to entities in the knowledge base. To solve this problem, a classifier is trained using the German Wikipedia and its hand annotated links as knowledge base. This section first describes how the classifier, which solves the combined NER and EL problem, is trained and then explains how documents are annotated.

> Sechs Top-Manager von **GM**, darunter Robert A. Lutz sowie Carl-Peter Forster, welcher als Group Vice President für das Europageschäft zuständig ist, trennten sich Anfang Mai 2009 von ihren gesamten Anteilsscheinen.
> PSA-Chef Carlos Tavares hatte zugesagt, **Opel** als deutsches Unternehmen zu erhalten. Er hatte aber zugleich angekündigt, **Opel** müsse sich im Fall einer Übernahme durch **PSA** weitgehend aus eigener Kraft sanieren.

Quote 1. Two example sentences containing highlighted named entities. Business entities are bold and other non-business entities are underlined.

3.1 Classifier Training

The German Wikipedia has around 3.6 million pages and around 38 million links occurring in the text. Only a fraction of these contains relevant data, which is useful to train a classifier to link business entities. To reduce this massive amount of data to a more relevant dataset the links are filtered with the help of Wikidata. Wikidata is a structured knowledge base derived from Wikipedia. Using its ontology the pages in Wikipedia can be reduced to only the pages about businesses. Let W be the set of all Wikipedia pages and $W_{bsn} = \{w \in W \mid w \text{ is tagged as a business or a subclass of it in Wikidata}\}$. Let E be the set of entities in the knowledge base and let function $f : E \to W$ map each entity in the knowledge base to its Wikipedia page so that $f(e) = w \Leftrightarrow w$ is the Wikipedia page about entity e.

Definition 1. *A **link** $l = (w,i,a,e)$[4] is the textual occurrence of an alias a in position i within Wikipedia page w and linking to the entity e.*

Using the smaller set of business pages W_{bsn}, the relevant links can be reduced as well. Let L be the set of all links occurring in Wikipedia and $L_{bsn} = \{l \in L \mid f(l_e) \in W_{bsn}\}$.

[4] The elements of a tuple will be referred to via an index, i.e., given a link l, l_w stands for the page w on which l occurs.

With this smaller set of links pointing to the pages of businesses, the possible aliases of businesses can be extracted. Let A be the set containing every alias appearing in Wikipedia and $A_{bsn} = \{a \in A \mid \exists l \in L_{bsn} : l_a = a\}$. Next L_{bsn} is expanded by selecting every link with an alias that has the possibility to link to a business. These links $M_{bsn} = \{l \in L \mid \exists a \in A_{bsn} : a = l_a\}$ comprise the relevant links that will be used to train the classifier.

Due to the Wikipedia editing guidelines, however, there are two issues with these links: the mention of an entity on its own page is not linked to the page and usually only the first occurrence of an entity's mention is linked. To solve these two issues the links on a given Wikipedia page w need to be extended. First, all links occurring on w are extracted into the set $N_w = \{l \in M_{bsn} \mid l_w = w\}$. These links are then used to create the set $O_w = \{w' \in W \mid \exists l \in N_w : f(l_e) = w'\} \cup \{w\}$, which contains the page $f(e)$ for every entity e that a link on page w links to. It also contains w itself. Using O_w all links linking to these pages are aggregated into $P_w = \{l \in M_{bsn} \mid \exists w' \in O_w : f(l_e) = w'\}$.

Then P_w is used to create the aliases $A_w = \{a \in A_{bsn} \mid \exists l \in P_w : l_a = a\}$. They are tokenized and these tokens are used to create a trie [6]. In this trie each leaf represents a token of an alias. For example, the alias „*Volkswagen AG*" would be represented by the leaves „*Volkswagen*" and „*AG*". With this trie all occurrences of the aliases A_w on page w are found and transformed into the extended links $Q_w = \{x \mid x_a \in A_w \ \wedge \ \nexists l \in N_w : l_i = x_i\}$.

Definition 2. *An* ***extended link*** *$x = (w,i,a,E_a)$ is the textual occurrence of an alias a in position i within Wikipedia page w. It possibly links to any of the entities E_a, which are defined as follows: $E_a = \{e \in E \mid \exists l \in L : l_e = e \ \wedge \ l_a = a\}$.*

These extended links are then transformed into actual links R_w extending N_w by applying a partial function g. This function transforms an extended link $x = (w, i, a, E_a)$ into a link $l = (w, i, a, e)$ if and only if E_a has only a single element e or there is only one entity $e \in E_a$ for which there are at least two links with alias a and these links make out at least 90% of all links pointing to e. $S_w = R_w \cup N_w$ then makes up all the links on page w used to train the classifier.

The links in S_w all represent true annotations. But the classifier is also supposed to disambiguate true annotations and false, non-business ones, as seen in Quote 1. Thus the training set needs to be extended by false annotations. These are found by searching for the occurrences of known aliases that were not linked by a human. The assumption is that since these entities were not linked by a human that they are false in this context.

Definition 3. *A* ***trie alias*** *$t = (w,i,a)$ is the textual occurrence of an alias a in position i within Wikipedia page w found by using a trie of known aliases.*

To find these occurrences the aliases in A_{bsn} are tokenized and these tokens are again used to create a trie. This trie is then used to find reoccurrences of known aliases. In the case of overlapping occurrences, only the longest occurrence is used. An example of such an overlap would be the sentence „*Die Audi AG sitzt in Ingolstadt.*“. Here the trie would find occurrences of both „*Audi*“ and „*Audi AG*“ but only the longest occurrence will be considered. These trie aliases make up the set $T_w = \{t \mid \nexists l \in S_{t_w} : l_i = t_i\}$ containing the false annotations on page w.

The number of trie aliases found is several times higher than the number of links and extended links per page. This can cause problems if there are so many more false annotations that the classifier just classifies every entry as wrong. It does this because there are so many false annotations that it maintains an accuracy of over 99%. The number of trie aliases is, therefore, reduced by filtering stop words and symbols because they will almost never refer to businesses in newspaper articles.

To train the classifier, the links and trie aliases of the small subset of Wikipedia pages W_{bsn} are used. Every link in S_w and every trie alias in T_w is transformed into a number of feature entries. The possible pages a given alias a can point to are $W_a = \{w \in W \mid \exists l \in L_{bsn} : l_a = a \ \wedge \ f(l_e) = w\}$. For every link or trie alias with an alias a exactly $|W_a|$ feature entries are generated - one feature entry for every possible page that could be linked.

Definition 4. *A **feature entry** y = (w,i,a,e,F) represents an alias occurring on Wikipedia page w in position i and possibly pointing to an entity e with a set of generated features F used to train a classifier.*

This set of features F contains three first order features and six higher order features as described by Janetzki [9]. The higher order features are sometimes called second order features as well. The first order features are the link score, the entity score and the context score. Given a feature entry $y = (w, i, a, e, F)$, the link score $f_1 \in F$ signifies how likely it is, that the alias y_a links to any entity. Therefore, every feature entry generated from the same link or trie alias has the same link score f_1. The entity score $f_2 \in F$ represents the likelihood that, given the alias y_a links to any entity, it links specifically to entity y_e. The context score $f_3 \in F$ shows the similarity of the words around alias y_a on page y_w in position y_i to the words on page $f(y_e)$. The higher order features are only generated for the entity score and the context score because the link score is equal for all feature entries generated from one link or trie alias. They are as described by Janetzki the rank, Δtop and $\Delta succ$ [9]. They describe the relationship between the feature entries generated by a single link or trie alias. The rank is an integer ranking of the values of the feature, which is defined by the partial order $\geq$. This means that the highest value of the feature has the lowest and best rank of 1 while the lowest value has the worst rank of n, with n being the number of feature entries generated for a specific link or trie alias. Δtop describes the difference between the current feature entry to the one with rank 1. In the case of the feature entry with rank 1, its value is ∞. $\Delta succ$ describes the difference to the feature entry with the next higher (worse) rank. In the case of the worst feature entry with rank n its value is also ∞.

Around 545 million feature entries containing these features were used to train the classifier. Different classification models will be evaluated in Section 4 and the first and higher order features will be evaluated in Section 5.

3.2 Annotation of Newspaper Articles

The process of annotating newspaper articles is very similar to the training process. Given an article D, the first task is to find possible mentions of businesses. This is done with the trie built using A_{bsn}. These trie aliases are again filtered by removing stop words and symbols since they are assumed to almost never reference a business. Following that, a number of feature entries is generated from every remaining trie alias found in article D. They are then classified using the trained classification model. Because one trie alias generates multiple feature entries, which are classified independently, it is possible that multiple feature entries generated from one trie alias are classified as links to businesses. When this happens, there is no way to rank the multiple possible entities since the classifier outputs only a boolean decision. Because of that, these collisions are filtered and the alias is not linked to any entity.

4 Evaluation of Classification Models

This section first evaluates different classification models with three quality measures. These are the precision P, the recall R and the F_β-score, which are defined in Definition 5. Specifically, the F_1-score is used, which is the harmonic mean of the precision and the recall and thus favors neither. Then the class thresholds of a Random Forest model are evaluated to see whether or not they can be used to create both a high precision and a high recall model to create seed and candidate alignments as described by Grütze et al. [8]. Finally, this section compares the just introduced quality measures with adjusted quality measures that take the collisions described in the previous section into account. These adjusted quality measures are more realistic for the use case of this approach since the collisions are thrown away.

Let T be the set containing the feature entries to be classified. Each feature entry has a label containing the result (positive or negative) the classifier should predict. A positive label on a feature entry means that its alias links to its entity and that it should be classified as such. A negative label means that the feature entry should not be classified as a link because either it's not a link at all or it links to the wrong entity. The subsets T_{TP}, T_{FP} and T_{FN}, which are used to calculate the quality measures, are defined as follows.

$$\begin{aligned}
T_{TP} \subset T&, \text{ where } \forall y \in T_{TP} : y \text{ is labeled positive} \land y \text{ is classified positive}\\
T_{FP} \subset T&, \text{ where } \forall y \in T_{FP} : y \text{ is labeled negative} \land y \text{ is classified positive}\\
T_{FN} \subset T&, \text{ where } \forall y \in T_{FN} : y \text{ is labeled positive} \land y \text{ is classified negative}
\end{aligned}$$

Definition 5. $P = \frac{|T_{TP}|}{|T_{TP} \cup T_{FP}|}$ $R = \frac{|T_{TP}|}{|T_{TP} \cup T_{FN}|}$ $F_\beta = \frac{(1 + \beta^2) \cdot P \cdot R}{(\beta^2 \cdot P) + R}$

4.1 Model Comparison

The following classification models are tested: Naive Bayes, Logistic Regression, Gradient Boosted Trees and Random Forest. The implementations of the Apache Spark MLlib [5] will be used. Specifically, the Data Frame API of Spark 2.1.0, since it is the primary API. If parameters of a model are not discussed further, then the default parameters of the Spark MLlib were used. The data set used to train and test the models is extracted from the Wikipedia pages W_{bsn} as described in Section 3. 70% of the generated feature entries are used to train the model and the remaining 30% are used to test it. Since the data set contains many more negative entries than positive entries, it is very likely for a model to classify every feature entry as negative. That way the model would still have an accuracy of over 99%. To mitigate this, the training set is filtered by removing every entry having a rank ≥ 10. This rank is the higher order feature of either the link score or the context score.

Figure 1 shows precision, recall, F_1-score and $F_{0.25}$-score of the four tested models. It includes the $F_{0.25}$-score as extra quality measure because it favors the precision. This

[5] https://spark.apache.org/mllib/

mirrors the goal of reliability of this approach. Both the Naive Bayes and the Logistic Regression classified every feature entry as negative, which resulted in a precision, F_1-score and $F_{0.25}$-score of NaN (due to the division by 0). This is caused by too many negative input feature entries with not enough positive ones as just described. Contrary to them, both the Gradient Boosted Trees and the Random Forest classified the data successfully. Only these two models use Decision Trees, showing that such unequal distributions of negative and positive input data are handled better by Decision Tree based models. The results of both are very similar, with the Random Forest having 2% more precision and the Gradient Boosted Trees having 6% more recall. The focus lies on the reliability of the model and thus on the precision. The $F_{0.25}$-score shows that the Random Forest model performs better when the precision is emphasized. It is, therefore, used for the feature evaluation in Section 5.

Most models have class thresholds as parameters as well. With these, a class can be favored. The thresholds can be set to either favor the class positive or the class negative. In the case of favoring the class negative, the classifier needs to be more certain about a feature entry to classify it as a link. In the opposite case of favoring the class positive, the classifier needs to be less certain and thus classifies more feature entries as a link.

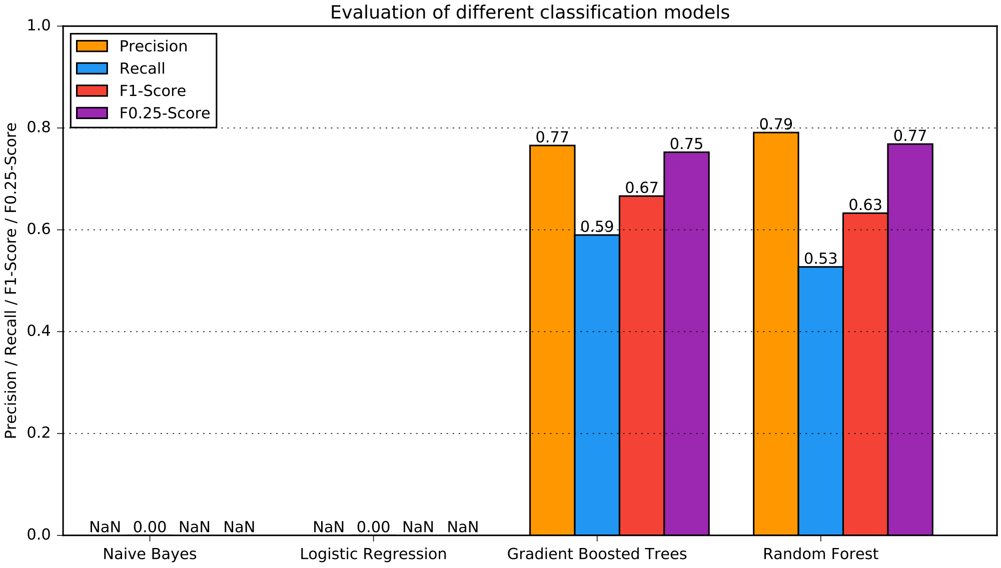

Figure 1: Evaluation of different classification models.

The Naive Bayes and Logistic Regression were both tested with different thresholds for the classes. They were especially tested with thresholds heavily favoring the positive class to see whether or not they would classify anything as positive. Unfortunately, this did not result in a different classification result.

The same parameters concerning the Decision Trees were used for the Gradient Boosted Trees and the Random Forest. These were 20 trees, a maximum depth of 6 and a maximum of 40 bins. The Gradient Boosted Trees were tested with a few different number of iterations. The values displayed in Figure 1 were the best for the tested number of iterations, which were 20 iterations. The other values resulted in a minimally worse precision. The number of trees, the maximum depth and the maximum number of bins were tested using a Random Forest, but none of these parameters changed anything significantly.

4.2 Class Threshold Evaluation

The class thresholds can be, as previously described, used to favor one class over the other. Since the Spark MLlib does not provide a way to change the class thresholds or the loss function for Gradient Boosted Trees the evaluation of the class thresholds is done only with a Random Forest. Figure 2 shows the behavior of the quality measures with a different threshold. A threshold > 1 means that the class positive is favored and a threshold < 1 means that the class negative is favored. The relative distance signifies how strong a class is favored, i.e., a threshold of 0.5 favors the class negative as much as a threshold of 2.0 favors the class positive.

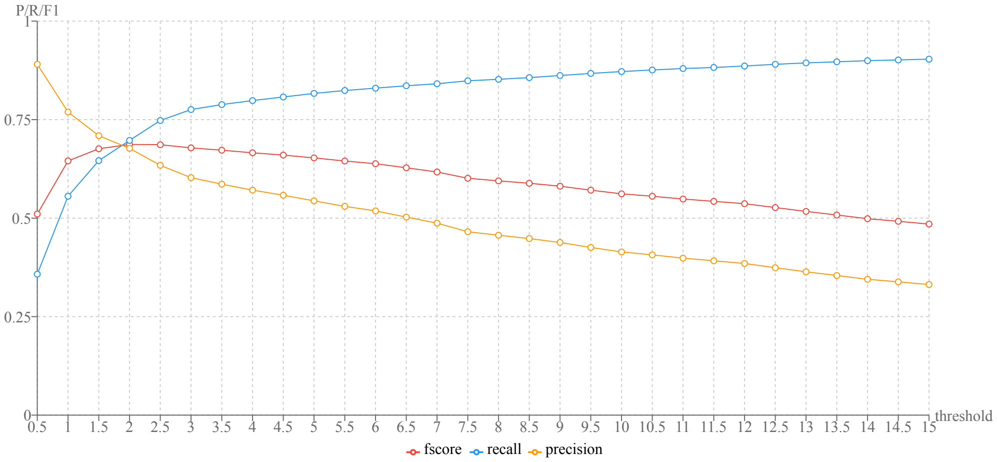

Figure 2: Evaluation of different class thresholds of a random forest model.

It is visible that a low threshold results in a very high precision with a relatively low recall. The threshold of 0.5 results in a precision of 89% and a recall of 36%. A very high threshold leads to a higher recall with a worse precision. An 80% recall is achieved with a threshold of 4, which has a precision of 57%. A 90% recall is achieved with a threshold of 14.5, which has a precision of 34%. This shows that further increasing the recall to very high levels decreases the precision a lot more than it increases the recall.

Adjusting the thresholds indeed produces a high precision and a high recall classifier, which in the future can be used to create the high-quality seed and high-coverage candidate alignments presented by Grütze et al. [8]. The candidate alignments would be used to increase the recall of the seed alignments, e.g., by using Random Walks.

4.3 Comparison with realistic Quality Measures

The adjusted quality measures take the collisions described in Section 3 into account. These are an adjusted precision and an adjusted recall. They are calculated the same way the normal precision and recall are calculated but use different sets for the calculation.

Let the set $U \subset T$ contain all feature entries, which were the only ones classified as positive for its alias, i.e., there is no collision. U is then used to remove all feature entries with collisions. Let $T'_{TP} = T_{TP} \cap U$ and $T'_{FP} = T_{FP} \cap U$. The feature entries producing collisions are then added to T_{FN}: $T'_{FN} = T_{FN} \cup (T_{TP} - U) \cup (T_{FP} - U)$. The sets T'_{TP}, T'_{FP} and T'_{FN} are then used to calculate precision, recall and F_1-score.

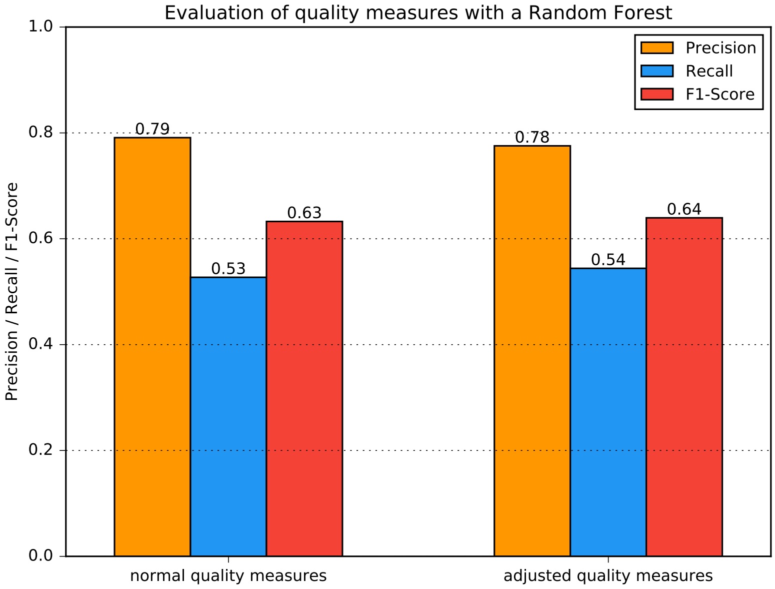

Figure 3: Evaluation of quality measures.

Figure 3 shows that, while taking the collisions into account, the precision drops only by one per cent. This proves that the classifier does not produce a lot of collisions and, therefore, the original quality measures are accurate enough to properly evaluate the classification models in this section and the features in the next section.

5 Feature Evaluation

This section evaluates the first and higher order features, which were described in Section 3. The quality measures used are again precision, recall and F_1-Score, which were defined in Definition 5 and used in Section 4. First, the first order features are evaluated, then the addition of all higher order features and finally every single higher order feature of the context score is evaluated. Of these evaluations, the first and last ones are done with the leave-one-out strategy to avoid a time-consuming exhaustive grid search. In the case of the first order features, there is also no other way to evaluate the link score since it is equal for all feature entries generated from the same link or trie alias. The tests were done with a Random Forest classifier, which was trained as described in Section 4. The parameters used were again a maximum tree depth of 6, a maximum number of bins of 40 and 20 trees.

5.1 First Order Features

The first order features are, as described in Section 3, the link score, entity score and context score. Figure 4 shows a leave-one-out evaluation of these features. Each group of bars, which show precision, recall and F_1-Score, shows the quality of a model trained using the specified first order features. The model trained with all first order features is used to evaluate the other three models by comparing them to it.

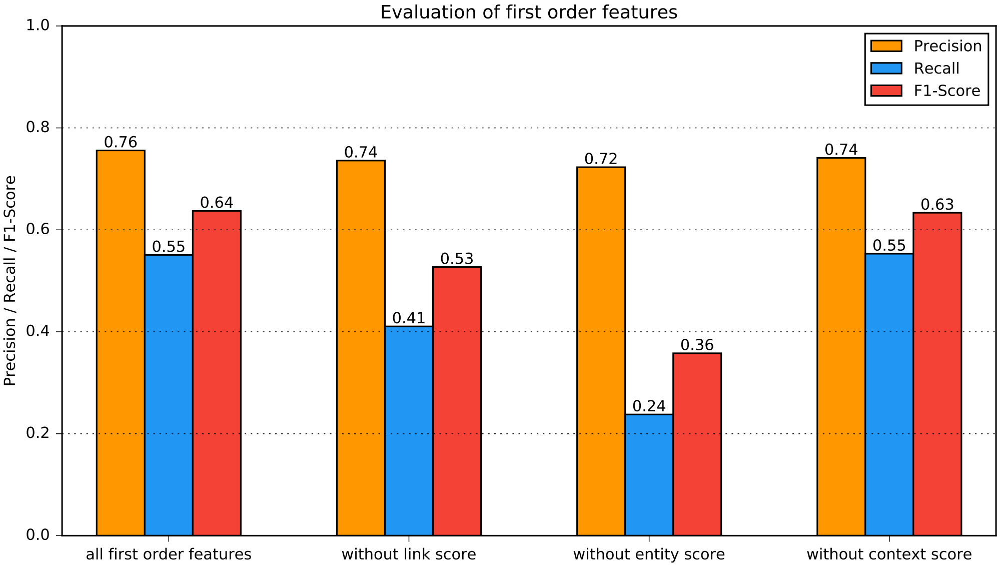

Figure 4: Leave-one-out evaluation of the first order features.

The model without the link score has only 2% less precision and 14% less recall compared to the model with all first order features. This shows that, as intended, the link score mainly signifies whether or not the alias of a specific feature entry is an important word.

The drop in the precision is so small because the link score can't be used to disambiguate between multiple entities one alias could link to since it is the same for all feature entries generated for one link or trie alias.

The removal of the entity score causes, compared to the model with all features, a small precision drop as well. The recall drop is a lot larger with 31%. Because the recall is only at 24% it is not surprising that the precision drop is only relatively small. The small recall signifies that only the obvious feature entries were classified as positives. An example for such an obvious feature entry would be the alias „*Commerzbank AG*“. This alias occurs 125 times, of which 93 are as link, and every time it is linked it links to the page „*Commerzbank*“. Therefore, only one feature entry is generated for the alias and it has a high link score of 0.744, meaning that its likely to be classified as positive. The large drop of the recall shows that the entity score is the most important feature, doing most of the disambiguation and thus entity linking. Since all the feature entries for the same alias have the same link score, only the context score would disambiguate the entities. But, as Janetzki showed, the range of the context score is relatively small, while the range of entity score is very large [9]. The small range of the context score results in a difficult disambiguation between multiple feature entries if it is the only feature used to do so.

The last model has very similar statistics compared to the model with all features. The small precision drop shows that the context score is used to disambiguate between feature entries of an alias with multiple very likely entities. An example would be the alias „*BVB*“. Out of the 340 times, it is linked in the Wikipedia, it links 107 times to the „*Basler Verkehrs-Betriebe*“ and 208 times to „*Borussia Dortmund*“. This means that the entity scores are relatively similar while the context scores will most likely differ due to the entities appearing in very different contexts. E.g., „*Basler Verkehrs-Betriebe*“ is most likely meant if the context of the alias is about traffic and „*Borussia Dortmund*“ is most likely meant if the context is about soccer.

5.2 Higher Order Features

The higher order features, which were described in Section 3, are the rank, Δtop and $\Delta succ$. They are added to the entity score and the context score since they show the relationship between the feature entries generated from a single link or trie alias. Figure 5 shows the quality of four classification models. The first model was trained using only the first order features. The next two were trained with the higher order features for only a single feature. The last model was trained with the higher order features for both the entity and the context score.

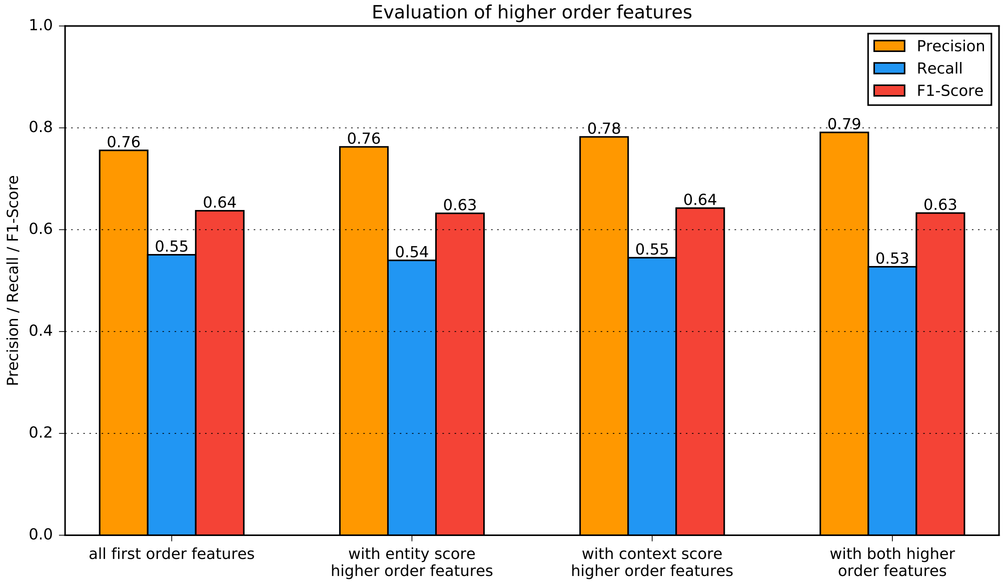

Figure 5: Evaluation of the higher order features.

Adding the higher order features to the entity score did not improve the model by a lot. The precision improved by only 0.7%, which is not visible in Figure 5 due to rounding, and the recall dropped by a per cent. This demonstrates that since the entity score itself already has a very large impact on the model, the higher order features did not do so and made the model only a little more cautious. The relationship between the entity scores of multiple feature entries is in most cases already shown by the scores themselves because for many aliases there are only a few entities with a high entity score. This explains the small impact of the higher order features.

Adding the higher order features to the context score improves the precision by 2% while dropping the recall only by 0.5%. The drop of the recall is not visible due to rounding. This displays that adding the higher order features to the context score further improves the disambiguation with minimal impact on the recall. The higher order features have a higher impact on the context score compared to the entity score because, as previously said, the value range for the context score is relatively small. This causes the higher order features to bring the small values into perspective, e.g., the rank tells the classifier that a value is the best value even though it is small. The disambiguation is improved as feature entries with otherwise high scores might have the worst similarity of the words appearing around the alias and the page of the linked entity. The resulting context score is small, has a bad rank and a large Δtop. Previously the difference of the context score might not have been large enough to matter but now the rank easily shows that the feature entry is a bad candidate.

The last model used the higher order features for both the entity and the context score. The improvement of the precision compared to the model with the higher order features for the context score shows that the higher order features for the entity score indeed improved the precision by a small amount. Likewise, the drop of the recall compared to the model with the higher order features for the entity score shows that the higher order features for the context score dropped the recall by a small amount. Overall the precision is increased by 3% compared to the model without higher order features. The recall dropped by only 2%. This shows that while the higher order features did help improve the classifier they did so only by a small amount unlike the improvement seen by Grütze et al. [8].

The impact of each higher order feature on the entity score is not shown since it is minimal, which is demonstrated by the generally small impact of the entity score's higher order features. The context score's higher order features are evaluated using a leave-one-out evaluation. Figure 6 shows the quality of four models. The first one was trained with all first and higher order features and will be used for comparison. The next three models each show a model without the described higher order feature.

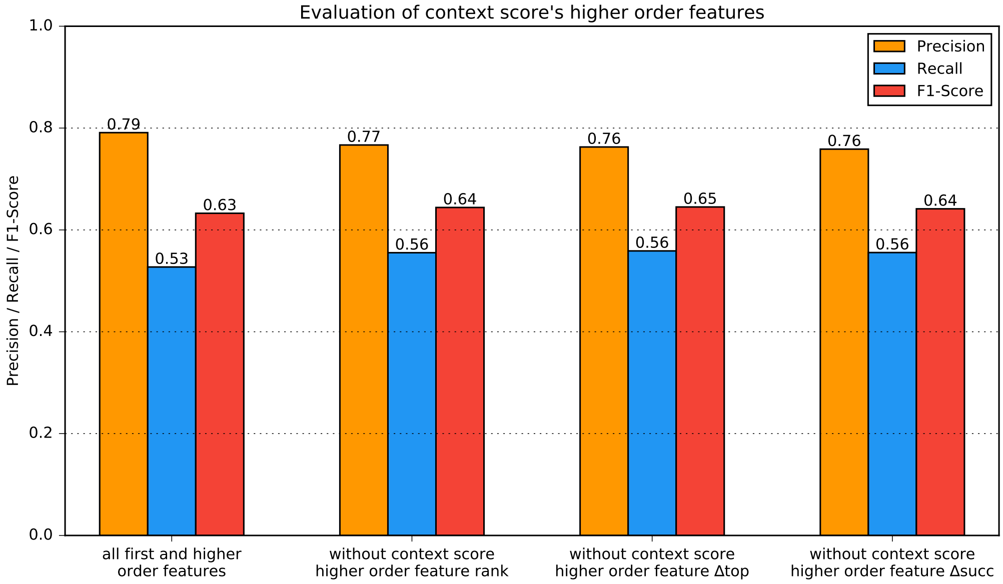

Figure 6: Leave-one-out evaluation of the context score's higher order features.

The removal of any higher order feature drops the precision by $2 - 3\%$ and increases the recall by 3%. While not visible, the largest impact has $\Delta succ$. Its removal drops the precision by 0.3% more than the removal of Δtop. This demonstrates that the higher order features work best when using all of them together, an observation Grütze et al. made as well [8].

6 NEL on Newspaper Articles

This section discusses the impact of different training data on and the performance of the presented NER and EL approach when annotating German newspaper articles. The documents were annotated on an eight node cluster using Apache Spark. Each node has a 6-Core CPU with 2.4 GHz and 64GB of RAM.

Impact of different Training Sets

When the classifier is trained using only feature entries generated by links the resulting annotations contain many false annotations. The inclusion of the trie aliases makes the classifier a lot more cautious by providing many occasions, where the alias is not linked at all. But this also adds many more feature entries and thereby increases the duration needed to train the model. It also adds so many negative entries that even Decision Tree models classify every feature entry as negative. The data set used in Section 4 and 5 contains a total of 1.8 million positive feature entries and 545.9 million negative feature entries. When removing all feature entries with a rank ≥ 10, the around 548 million feature entries are reduced to only 25.3 million. To preserve a realistic precision and recall only the feature entries used to train the model are filtered. Mainly negative feature entries are removed this way, which, therefore, fixes the problem of only negative predictions.

The selection of the Wikipedia pages used to generate the training data also influences the quality of the classifier. The precision and recall are not affected that strongly, but the annotation results vary greatly. Using all Wikipedia pages of businesses, which are around 1.5% of all pages, results in better annotation results than using a random 1% sample of pages. This is caused by the pages of businesses having, on average, more links than a random page and generally linking more to other businesses. More links per page means that there are more positive feature entries, which are rare, used to train the classifier.

Performance

The annotation of newspaper articles contains two steps. The first step is the generation of feature entries and the second the classification of those. The generation of the feature entries costs most of the time while the classification performs extremely well. The annotation of 3.6 million Wikipedia pages took 27 hours. As a comparison, the classification of 126 million feature entries, generated from 1% of Wikipedia pages, took only 2 minutes. The performance of the feature generation could be improved in the future by storing the documents in a more memory efficient way. An example would be using a global word dictionary for all words and storing the indices for each word in a document. The current performance of the document annotation is still good, and it is possible, as shown, to annotate millions of documents in an acceptable amount of time.

7 Conclusion

We presented an approach that combines both NER and EL and performs both in an efficient way. It is a practical solution as it is possible to annotate millions of documents in an acceptable amount of time. It annotated 3.6 million German Wikipedia articles in only 27 hours. With such a performance it is possible to annotate newly published newspaper articles in real-time. That way it is possible to extract relations from these annotated documents in real-time and keep the knowledge base up-to-date.

We used Apache Spark on an eight node cluster, as described in Section 6, to annotate the documents on a distributed system. Apache Spark scales horizontally, which means that to increase the performance only new nodes need to be added to the cluster. This is a very cost efficient way to scale since the nodes don't need top of the line hardware.

The combined NER and EL achieves a high precision of 90%. The recall can be increased to 80% if required at the cost of some precision. This is achieved by adjusting the class thresholds of the classification model as described in Section 4. These precision and recall values contain both NER and EL, which are both not easily solved.

There are a few ways to improve the performance and the quality of the presented approach. One way to increase the performance would be to decrease the memory footprint of the documents by using more memory efficient ways to encode them as described in Section 6. This would increase the scalability of the feature generation, which takes up most of the time. The quality of the classification could be increased by combining a high precision and a high recall model as proposed by Grütze et al. [8]. The current approach can be used to create both a high precision and high recall classifier by adjusting the class thresholds. These could then be combined by, e.g., using Random Walks to increase the recall of the high precision classifier without hurting its recall.

References

[1] Abgas-Skandal: VW-Zulieferer kämpfen mit Kurzarbeit. In: *Automobilwoche* (2016), October. `http://www.automobilwoche.de/article/20161018/AGENTURMELDUNGEN/310189944/abgas-skandal-vw-zulieferer-kaempfen-mit-kurzarbeit`

[2] Blessing, Andre ; Kuntz, Reinhard ; Schütze, Hinrich: Towards a Context Model Driven German Geo-tagging System. In: *Proceedings of the 4th ACM Workshop on Geographical Information Retrieval.* New York, NY, USA : ACM, 2007 (GIR '07). – ISBN 978–1–59593–828–2, 25–30

[3] Brauer, Falk ; Löser, Alexander ; Do, Hong-Hai: Mapping Enterprise Entities to Text Segments. In: *Proceedings of the 2Nd PhD Workshop on Information and Knowledge Management.* New York, NY, USA : ACM, 2008 (PIKM '08). – ISBN 978–1–60558–257–3, 85–88

[4] Bundesrepublik Deutschland, Statisches B.: *Unternehmensregister.* `https://www.destatis.de/DE/ZahlenFakten/GesamtwirtschaftUmwelt/UnternehmenHandwerk/Unternehmensregister/Tabellen/UnternehmenBeschaeftigteUmsatzWZ08.html`, Abruf: 2017-07-18

[5] Flaig, Imelda: VW-Abgasaffäre belastet Zulieferer im Land. In: *Stuttgarter Zeitung* (2016), October. `http://www.stuttgarter-zeitung.de/inhalt.baden-wuerttemberg-vw-abgasaffaere-belastet-zulieferer-im-land.06ea4637-d060-4109-ad3c-c341a04bf1bd.html`, Abruf: 2017-07-18

[6] Fredkin, Edward: Trie Memory. In: *Commun. ACM* 3 (1960), September, Nr. 9, 490–499. `http://dx.doi.org/10.1145/367390.367400`. – DOI 10.1145/367390.367400. – ISSN 0001–0782

[7] Gruner, Milan: *Analysis and Simplification of Business Graphs.* July 2017

[8] Grütze, Toni ; Kasneci, Gjergji ; Zuo, Zhe ; Naumann, Felix: CohEEL: Coherent and efficient named entity linking through random walks. In: *J. Web Sem.* 37-38 (2016), 75-89. `http://dblp.uni-trier.de/db/journals/ws/ws37.html#GrutzeKZN16`

[9] Janetzki, Jonathan: *Feature Extraction for Business Entity Linking in Newspaper Articles.* July 2017

[10] Loster, Michael ; Zuo, Zhe ; Naumann, Felix ; Maspfuhl, Oliver ; Thomas, Dirk: Improving Company Recognition from Unstructured Text by using Dictionaries, 2017

[11] Löper, Lando E. N. ; Radscheit, Matthias: *Evaluation of Duplicate Detection in the Domain of German Businesses.* July 2017

[12] Mikheev, Andrei ; Moens, Marc ; Grover, Claire: Named Entity Recognition Without Gazetteers. In: *Proceedings of the Ninth Conference on European Chapter of the Association for Computational Linguistics.* Stroudsburg, PA, USA : Association for Computational Linguistics, 1999 (EACL '99), 1–8

[13] Nadeau, David ; Sekine, Satoshi: A survey of named entity recognition and classification. In: *Linguisticae Investigationes* 30 (2007), January, Nr. 1, 3–26. `http://www.ingentaconnect.com/content/jbp/li/2007/00000030/00000001/art00002`. – Publisher: John Benjamins Publishing Company

[14] Pabst, Leonard: *Efficient Blocking Strategies on Business Data.* July 2017

[15] Rau, L. F.: Extracting Company Names from Text. In: *Proc. of the Seventh Conference on Artificial Intelligence Applications CAIA-91 (Volume II: Visuals).* Miami Beach, FL, 1991, S. 189–194

[16] Ritter, Alan ; Clark, Sam ; Mausam ; Etzioni, Oren: Named Entity Recognition in Tweets: An Experimental Study. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* Stroudsburg, PA, USA : Association for Computational Linguistics, 2011 (EMNLP '11). – ISBN 978–1–937284–11–4, 1524–1534

[17] Schneider, Alec: *Evaluation of Business Relation Extraction Methods from Text.* July 2017

[18] Sil, Avirup ; Yates, Alexander: Re-ranking for Joint Named-entity Recognition and Linking. In: *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management.* New York, NY, USA : ACM, 2013 (CIKM '13). – ISBN 978–1–4503–2263–8, 2369–2374

[19] Strelow, Nils: *Distributed Business Relationships in Apache Cassandra.* July 2017

Statutory Declaration

I declare that I have written this thesis independently, that I have not used any other than the declared resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Potsdam, July 21, 2017

. . . .
Jan Ehmüller