Improving Company Recognition from Unstructured Text by using Dictionaries

Michael Loster, Zhe Zuo, Felix Naumann
Hasso-Plattner-Institute
Potsdam, Germany
firstname.lastname@hpi.de

Oliver Maspfuhl, Dirk Thomas
Commerzbank
Frankfurt am Main, Germany
firstname.lastname@commerzbank.com

ABSTRACT

While named entity recognition is a much addressed research topic, recognizing companies in text is of particular difficulty. Company names are extremely heterogeneous in structure, a given company can be referenced in many different ways, their names include person names, locations, acronyms, numbers, and other unusual tokens. Further, instead of using the official company name, quite different colloquial names are frequently used by the general public.

We present a machine learning (CRF) system that reliably recognizes organizations in German texts. In particular, we construct and employ various dictionaries, regular expressions, text context, and other techniques to improve the results. In our experiments we achieved a precision of 91.11% and a recall of 78.82%, showing significant improvement over related work. Using our system we were able to extract 263,846 company mentions from a corpus of 141,970 newspaper articles.

1. FINDING COMPANIES IN TEXT

Named entity recognition (NER) defines the task of not only recognizing named entities in unstructured texts but also classifying them according to a predefined set of entity types. The NER task was first defined during the MUC-6 conference [8], where the objective was to discover general entity types, such as persons, locations, and organizations as well as time, currency, and percentage expressions in unstructured texts. Subsequent tasks, such as entity disambiguation, question answering, or relationship extraction (RE), rely heavily on the performance of NER systems, which perform as a preprocessing step.

This section highlights the particular difficulties of finding company entities in (German) texts and introduces our industrial use-case, namely risk management based on company-relationship graphs.

1.1 Recognizing company entities

Although there is a large body of work on recognizing entities starting from persons and organizations, to entities like gene mentions or chemical compounds, the current research often neglects the detection of more fine-grained subcategories, such as person roles or commercial companies. In many cases, the "standard" entity classes turn out to be too coarse-grained to be useful in subsequent tasks, such as automatic enterprise valuation, identifying the sentiment towards a particular company, or discovering political and company networks from textual data.

What makes recognizing company names particularly difficult is that in contrast to person names they are immensely heterogeneous in their structure. As such, they can be referenced in a multitude of ways and are often composed of many constituent parts, including person names, locations, and country names, industry sectors, acronyms, numbers, and other tokens, which makes them especially hard to recognize. This heterogeneity is expected to be true particularly for the range of medium-sized to small companies. Regarding examples like "Simon Kucher & Partner Strategy & Marketing Consultants GmbH", "Loni GmbH", or "Klaus Traeger", which all are official names of German companies, one can easily see that they vary not only in length and types of their constituent parts but also in the position where specific name components appear. In the example "Clean-Star GmbH & Co Autowaschanlage Leipzig KG" the legal form "GmbH & Co KG" is interleaved with information about the type of the company (carwash) and location information (Leipzig, a city in Germany). What is more, company names are not required to contain specific constituent parts: the example "Klaus Traeger" from above is simply the name of a person. It does not provide any additional information apart from the name itself, which leads to ambiguous names that are difficult to identify in practice.

Additionally, and in contrast to recognizing named entities from English texts, detecting them in German texts presents itself as an even greater challenge. As pointed out by Faruqui and Padó, this difficulty is due to the high morphological complexity of the German language, making tasks such as lemmatization much harder to solve [5]. Hence, features that are highly effective for English often lose their predictive power for German. Capitalization is a prime example of such a feature. Compared to English, where capitalization of common nouns serves as a useful indicator for named entities, in German *all* nouns are capitalized, which drastically lowers the predictive power of the feature.

We propose and evaluate a named entity recognizer for German company names by training a conditional random field (CRF) classifier [13]. Besides using different features,

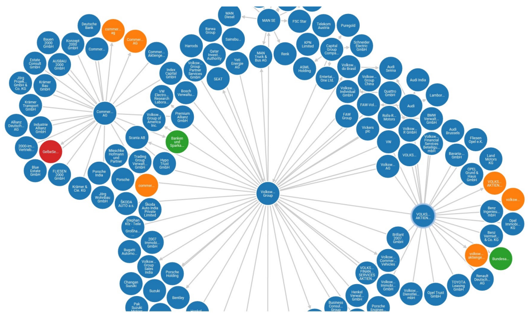

Figure 1: An example of a company graph.

the fundamental idea is to include domain knowledge into the training phase of the CRF by using different real-world company dictionaries. Transforming the dictionaries into token tries enables us to determine efficiently whether the analyzed text contains companies that are included in the dictionary. During a preprocessing step, we use the token trie to mark all companies in the analyzed text that occur in the used trie. In addition, we automatically extend the dictionaries with carefully crafted variants of company names, as we expect them to occur in written text.

1.2 Use case: Risk management using company graphs

Among the many possible applications for a company-focused NER system, we focus on modern risk management in financial institutions as one that would benefit from such a system. Named entity recognition and subsequent relationship extraction from text for the purpose of risk management in financial institutions is particularly important in the context of illiquid risk [1]. Illiquid financial risks basically represent contracts between two individuals, e.g., a bank granting a credit over 1 Mio USD (creditor) to a private company (obligor). Because the risk that the credit-taking company will not honor its repayment obligations cannot be easily transferred to other market participants, assessing the creditworthiness of an obligor is of major importance to the relatively small number of its creditors and other business partners. Also, insights gained by one bank on the obligor's ability to pay back are usually not shared. Hence, obtaining adequate and timely information about non-exchange-listed obligors becomes a difficult task for creditors.

To circumvent this difficulty, financial institutions rely on the so-called "insurance principle": pooling a huge number of independent gains or losses ultimately results in the diversification of risk, which in turn eliminates almost all of it. Unfortunately, risk mitigation based on the insurance principle relies on the independence assumption between individual gains or losses. At the latest with the financial crisis of 2008/2009, this low dependency assumption has turned out to be devastatingly wrong. Information on the economic dependency structure between contracting parties and assets can be seen as the holy grail of financial risk management.

Traditionally, the internal and external data sources used to assess credit risk focus on individual customers, not on the relationships between them. Dependency information is inferred from exposure to common risk factors and thus is inherently symmetric. Direct non-symmetric dependencies, such as supply chains, are not captured.

Fortunately, with the growing amount of openly available data sources, there is justified hope that dependency modeling becomes significantly easier by leveraging this vast amount of data. Sadly, most of those data sources are text-based and require considerable effort to extract the contained knowledge about relationships and dependencies between the entities of interest. The desired outcome of such an extraction effort can be organized in a graph as shown in Figure 1. The figure shows an example of an actual company graph. To be able to automatically extract such graphs from large amounts of unstructured data, a reliable NER system constitutes the first decisive prerequisite for a following relation extraction step.

As pointed out at the beginning, the described use case is merely one of many possible use cases, others might include semantic role labeling, machine translation, and question answering systems.

1.3 Contributions and structure

We address the problem of recognizing company names from textual data by incorporating dictionary matches into the training process using a feature that represents whether a token is part of a known company name. Our evaluation focusses on analyzing the impact of using a perfect dictionary and different real-world dictionaries, as well as the effects of different ways to integrate the knowledge contained in the dictionaries on the performance of the NER system. In particular, we make the following contributions:

- Creation of a NER system capable of successfully recognizing companies in German texts with a precision of 91.11% and a recall of 78.82%.

- Analysis of the impact of various dictionary-based feature strategies on the performance of the NER.

The remainder of this paper is organized as follows: Section 2 discusses related work, while Section 3 presents the baseline configuration for the CRF. In Section 4 we give an overview of the text corpus and the dictionaries we used. We describe the key data structures and technical aspects of the approach in Section 5. Finally, Section 6 presents our experimental results and Section 7 concludes the paper.

2. RELATED WORK

Since its first appearance on the MUC-6 conference [8], the problem of named entity recognition (NER) has become a well-established task leading to many systems and methods that have been developed over time [16]. Before discussing the differences of our approach to the most related approaches, we start by giving an overview of the related work.

Most existing NER systems can be classified into *rule-based* [3, 21], *machine learning-based* [15, 27], or *hybrid* systems [10, 22]. While rule-based systems make use of carefully hand-crafted rules, machine learning approaches tend to train statistical models, such as Hidden Markov Models (HMM) [27] or Conditional Random Fields (CRF) [13], to identify named entities. Hybrid systems combine different methods to compensate their individual shortcomings. They try to incorporate the best parts of the applied methods to reach a high system performance.

Currently, many approaches to the NER problem rely on CRFs [5, 12, 15]. One of the most popular and freely available NER choices for English texts is the Stanford NER system [6]. It recognizes named entities by employing a linear-chain CRF to predict the most likely sequence of named entity labels. While this system shows good performance on English texts, it's performance values decrease when applied to German texts. This effect has also been pointed out by Benikova et al. [2], who argue that German NER systems are not on the same level as their English counterparts despite the fact that German belongs to the group of well-studied languages. This difficulty arises from the fact, that the German language has a very rich morphology, making it especially challenging to identify named entities. Besides the already mentioned problem of capitalization, the German language is capable of creating complex noun compounds like "Vermögensverwaltungsgesellschaft" (asset management company) or "Industrieversicherungsmakler" (industry insurance broker), which make the application of traditional NLP methods even harder.

Nonetheless, German NER systems exist, and some were presented at the CoNLL-2003 Shared Task [23]. With the participating systems achieving F_1 scores between 48% and 73%, the winning system [7] obtained an overall F_1-measure of 72.41% on German texts and 64.62% on recognizing organizational entities. Since the creation of systems for the CoNLL-2003 Shared Task more than ten years ago, one of the most successful NER systems for the German language was introduced by Faruqui and Padó [5]. It reaches overall F_1 scores between 77.2% and 79.8% by using distributional similarity features and the Stanford NER system. Even more recently, additional German NER systems were presented at the GermEval-2014 Shared Task [2]. The GermEval Shared Task specifically focuses on the German language and represents an extension to the CoNLL-2003 Shared Task. The three best competing systems were ExB [9], UKP [19], and MoSTNER [20]. All of them apply machine learning methods, such as CRFs or Neural Nets, which leverage dependencies between the utilized features. Additionally, they use semantic generalization features, such as word embeddings or distributional similarity to alleviate the problem of limited lexical coverage, which, according to [26], is triggered by the often insufficient corpus size used in the training phase of statistical models. To summarize the performance of these systems, they operate in the range of 73% to 79% F_1-measure.

Considering the role of dictionaries in the process of building NER systems, Ratinov and Roth [18] argue that they are crucial for achieving a high system performance. The process of automatically or semi-automatically creating such dictionaries from various information sources has been addressed by [11, 18, 24]. Their research focuses on automatically creating large dictionaries, also known as gazetteers, from open and freely available data sources, such as Wikipedia. The general idea is to establish and assign category labels for each word sequence representing a viable entity by using the information contained in corresponding Wikipedia articles. According to [24], dictionaries can be separated into two different classes, so-called *trigger dictionaries*, which contain keywords that are indicative for a particular type of entity, and *entity dictionaries*, which are comprised of the entire entity labels. For example, a trigger dictionary for companies would most likely contain legal-form words for companies, such as "GmbH" (LLC) or "OHG" (general partnership), whereas an entity dictionary would contain the entire representation of the entity itself, e.g., "BMW Vertriebs GmbH". For our approach we decided to employ entity dictionaries, because there are many openly available data sources from which they can be constructed. Similar to semantic generalization features, features generated from dictionaries aim to mitigate the unseen word problem resulting from the low lexical coverage of statistically learned models.

Many systems make use of dictionaries to increase their performance. All systems mentioned above use dictionaries at some point in their process [9, 19, 20]. Most of the currently existing systems integrate the knowledge contained in dictionaries by constructing features that represent a dictionary lookup. Since each dictionary accounts for a particular type of entity, the constructed feature encodes to which dictionary the word currently under classification belongs and, therefore, implicitly provides evidence for its correct classification. These features are subsequently used in the training process of statistical models, such as CRFs or HMMs.

Another way of integrating dictionary knowledge into the training process of an NER system is described by Cohen and Sarawagi [4]. They present a semi-Markov extraction process capable of classifying entire word sequences instead of single words. By doing so, they effectively bridge the gap between NER methods that sequentially classify words and record linkage metrics that apply similarity measures to compare entire candidate names.

While the previously mentioned systems focus on detecting entities belonging to the entity class "organization", which, apart from companies, includes sports teams, universities, political groups, etc., our system, driven by our use case, specifically excludes such entities and solely focuses on detecting commercial companies. By using a preprocessing step that utilizes external knowledge from dictionaries, we annotate already known companies, which enables us to construct a feature that we use to train a CRF classifier. We concentrate on integrating the knowledge contained in the dictionary into the training process of the classifier. In this way, we use dictionaries from different sources and examine their impact on the overall system performance. Additionally, we report on strategies to integrate the domain knowledge provided by the dictionaries into the training process.

3. CONDITIONAL RANDOM FIELDS AS NER BASELINE

For the construction of our company-focused NER system, we use the CRFSuite Framework[1] to implement a conditional random field model (CRF), as one of the most popular models for building NER systems.

For the baseline configuration of the system, we used various features, such as n-grams, prefixes and suffixes, that are based on those used in the Stanford NER system [6]. Besides regarding different window sizes for each feature, we considered a variety of additional features, for example a token-type feature reducing the type of a token to categories like InitUpper, AllUpper etc., a feature that concatenates different prefix and suffix lengths for each token or features that try to capture some specific characteristic of German company names. However, these features did not result in additional improvements of our baseline configuration. In the end we arrived at a baseline configuration that consists of the following features:

[1]http://www.chokkan.org/software/crfsuite/

	The	auto	maker	VW	AG	is	now...
words :	w_{-3},	w_{-2},	w_{-1},	w_0,	w_1,	w_2,	w_3,
pos-tags :		p_{-2},	p_{-1},	p_0,	p_1,	p_2,	
shape :			s_{-1},	s_0,	s_1		
prefixes :			pr_{-1},	pr_0,			
suffixes :			su_{-1},	su_0,			
n-grams :				n_0,			

Here, the w symbol encodes the word token features of a text with its subscript marking the position of a token. Thus, w_0 refers to the current token whereas w_{-1} and w_1 refer to the previous and next tokens, respectively. The symbols p and s represent the part-of-speech and word shape features with analog subscript notation.

For the creation of POS tags we used the Stanford log-linear part-of-speech tagger [25]. As the name suggests, the shape feature condenses a given word to its shape by substituting each capitalized letter with an X and each lower case letter with an x. Thus the word "**Bosch**" would be transformed to "**Xxxxx**". We also added prefix and suffix features (pr, su) for the current and previous word. These features generate all possible prefixes and suffixes for the specific word. As the last feature we include the set n_0 of all n-grams of the term with n between 1 and the word length of the current word. This feature set yielded the best performance metrics for our baseline configuration without adding any external knowledge besides POS tags.

The baseline system achieves an F_1-measure of 80.65%. More detailed performance metrics of the baseline are presented later in Table 2, in the context of our overall experiments.

4. CORPUS & DICTIONARIES

Before describing our approach in Section 5, we introduce and examine the text corpus and the different information sources we used for building our dictionaries.

4.1 Text corpus

Our evaluation corpus consists of 141,970 documents containing approximately 3.17 million sentences and 54 million tokens. The documents were collected from five German newspaper websites, namely Handelsblatt, Märkische Allgemeine, Hannoversche Allgemeine, Express, and Ostsee-Zeitung. We intentionally selected not only large, national newspapers but also smaller, regional ones; we observe that larger newspapers have a tendency to report more about larger companies or corporations, while the regional press also mentions smaller companies due to their locality in the region. Thus, we expect to increase our chances of discovering smaller and middle tier companies (SMEs) in the long tail by using regional articles in our training process. We extract the main content from the articles by using jsoup[2] and hand-crafted selector patterns, which give us the raw text without HTML markup. Using our final NER system, we were able to extract a total of 263,846 company mentions from this corpus.

4.2 Dictionaries

To build our dictionaries we used two official information sources: the Bundesanzeiger (German Federal Gazette)[3] and the Global Legal Entity Identifier Foundation (GLEIF), which hosts a freely available company dataset[4]. Additionally, we used DBpedia[5] to account for large businesses and the German Yellow Pages[6] to cover middle-tier and local businesses. To simulate a best-case scenario, we composed a "perfect" dictionary containing all manually annotated companies from our testset. Finally, our last dictionary consists of the union of all dictionaries except the perfect one. Although the information sources discussed below contain many different attributes, we use only the company name for the creation of each dictionary.

Bundesanzeiger (BZ). The Bundesanzeiger is the official gazette for announcements made by German federal agencies. Among other things in contains official announcements from companies of various legal forms, such as corporations, limited liability companies, and others including those of foreign companies. Regarding this function, the role of the Bundesanzeiger, as well as the information it provides, is comparable to the U.S. Federal Register. By crawling the BZ company announcements we obtained 793,974 company names, their addresses, and their commercial register ID.

GLEIF (GL). The Global Legal Entity Identifier Foundation (GLEIF) was founded by the International Financial Stability Board[7] in 2014. It is a non-profit organization set up to aid the implementation of the Legal Entity Identifier (LEI). The LEI is designed to be a globally unambiguous, unique identifier for entities that partake in financial transactions. In this context, the dataset of legal entities assigned with a unique LEI is made available for public use by GLEIF. An entry in the provided dataset is, among other data, comprised of the LEI number, legal name, legal form, and address of a legal entity. At the time of writing, the dataset consists of 413,572 legal entities from all global countries that have been assigned a LEI. The subset for German legal entities (**GL.DE**) consists of 42,861 entries.

DBpedia (DBP). The DBpedia project is an effort to systematically extract information from Wikipedia and provide it to the public in a structured way [14]. Structuring the data contained in Wikipedia pages enables us to use query languages like SPARQL to answer complex queries based on data originated from Wikipedia. We queried for the names of all companies contained in the German DBpedia database, yielding a dictionary of 41,724 entries. The resulting dataset contains only companies that have a corresponding Wikipedia page. Thus, we expect that most of the collected company names in this dataset belong to larger, more important companies. Since the extracted names originating from Wikipedia pages, they are very often already in their colloquial form. Also, the dataset contains some additional aliases, such as "VW" for the "Volkswagen AG", which are difficult to generate automatically.

Yellow Pages (YP). As a marketing solutions provider, the German Yellow Pages maintains a large company register, which mainly contains information about small and middle-tier businesses. Using the web pages provided by the register, we were able to extract information, such as the company name, address, email address, phone number, and industrial sector for each company listed in the Yellow

[2]https://jsoup.org
[3]https://www.bundesanzeiger.de
[4]https://www.gleif.org
[5]http://wiki.dbpedia.org
[6]http://www.gelbeseiten.de
[7]http://www.fsb.org/

Pages. The dataset consists of 416,375 company entries.

Perfect Dictionary (PD). For evaluation purposes, we manually labeled company mentions in 1,000 documents (see Sec. 6.1 for details). The perfect dictionary contains exactly the 2,351 manually annotated companies from our training and testset. Because of their origin, the company names contained in this dictionary are already in their colloquial form. Hence, by using this dictionary in our approach, we were indeed able to correctly identify all companies occurring in our testset. Furthermore, this dictionary enables us to simulate the best case scenario in which the dictionary is comprised of all companies occurring in our testset.

All aforementioned dictionaries contain large sets of German company names, so we expect them to overlap. To gain a better understanding of our dictionary's coverages, we computed their mutual containment. We calculated the overlaps using exact match and a fuzzy match. The latter constitutes a more realistic matching scenario accounting for typos and other noise. For computing the matches we applied the method described in [17]. Summarizing their approach, the authors compute the similarity between two strings by splitting them up into n-grams and using similarity measures like Dice, Jaccard, or cosine similarity to determine their similarity using a threshold α. For our calculations we chose a trigram tokenization of the strings and cosine similarity as our metric. We calculated the fuzzy overlaps using different thresholds, and observed that a value of $\theta = 0.8$ performs best on our data.

The pairwise overlaps are shown in Table 1 on the left for exact matches and on the right for fuzzy matches. Surprisingly, even in the case of fuzzy overlaps, the highest overlap was only 11.24%, namely between the BZ and the GL dictionary. All other overlaps were below this value, except in cases where they were contained in each other (GL.DE$\subset$GL). The exact matching overlaps scored even lower with a maximum overlap of 1.37%.

We identified three possible reasons for these low overlaps. The first and most obvious reason is that our quite simplistic fuzzy matching is not sufficient to recognize many correct matches. Secondly, each of the dictionaries favors a different kind of company names and company sizes. For example, the DBpedia dictionary contains mostly colloquial names whereas the Bundesanzeiger refers to companies using their full legal name. Finally, the dictionaries where crawled at slightly different points in time, hence some of them may contain companies that no longer exist and are thus missing from the other dataset. As a consequence, we created an additional dictionary where we combined all of the mentioned dictionaries into one:

All Dictionaries (ALL). This dictionary is the union of all company names from all other dictionaries. In total it comprises 1,713,272 company names.

5. COMPANY RECOGNITION USING GAZETTEERS

Named entity recognition (NER) is a sequence labeling task that aims to sequentially classify each word in a given text as belonging to a specific class, e.g., person or company. As mentioned, we make use of the CRFSuite Framework to construct our NER system. First, we describe our alias generation process, which extends the given dictionaries, in Section 5.1. Then, Section 5.2 describes how we create the dictionaries and how we efficiently integrate the contained domain knowledge into the training process of the CRF.

5.1 Alias generation

Unfortunately, company names acquired from web sources contain noise, such as country names, legal forms, and other spurious terms. That is, they often differ significantly from their colloquial names. Here the "colloquial name" is to be understood as the name by which a company is commonly referred to in text. For example, while "Dr. Ing. h.c. F. Porsche AG" represents the official company name of the automobile manufacturer, we most often refer to the company by its colloquial name, which is simply "Porsche". Assuming that articles mention companies more frequently by their colloquial name then their official name, it becomes necessary to automatically derive such alternative names, in the following referred to as *aliases*, from a company's official name.

Regarding the alias generation, special attention should be paid to the fact that one company often possesses more then one alias. Considering again the example from above, the company Porsche has at least four valid and common aliases, namely "Dr. Ing. h.c. F. Porsche AG", "Ferdinand Porsche AG", "Porsche AG", or just plain "Porsche". Furthermore, there are a number of non-trivial aliases that are particularly difficult to anticipate by using an automated process. For example the automobile manufacturer "Volkswagen" is also referred to as "VW" or even "die Wolfsburger", referring to the the town of Wolfsburg, in which Volkswagen's headquarters is located.

Our alias generation process consists of the following five steps, using the example of TOYOTA MOTOR™USA INC..

	Step	Example
1	Removal of legal form designations	TOYOTA MOTOR™USA
2	Removal of special characters	TOYOTA MOTOR USA
3	Normalization	Toyota Motor USA
4	Country name removal	Toyota Motor
5	Stemming of company names	no change

Each of the Steps 1–4 yields one new alias for the currently processed company name resulting in four aliases per name. Note that some of the four aliases are identical and identical copies are removed. The fifth and final stemming step adds another five aliases by stemming the company name itself and all previously generated aliases. This means that a maximum of nine aliases could be generated by applying the five processing steps to a given company name.

1 & 2: Legal form & special character cleansing. We start to infer the aliases by using a rule-based approach based on regular expressions to strip away a company's legal form. The regular expressions we use are derived from the description of business entity types, found on Wikipedia[8]. The derivation process consists of looking at the business entity types for selected countries and manually creating regular expressions that are able to match the legal forms of the selected countries. We chose the countries based on the most frequent legal forms occurring in our datasets.

[8]http://en.wikipedia.org/wiki/Types_of_business_entity

	Exact match overlaps						Fuzzy match overlaps (cosine, $\theta = 0.8$)					
	BZ	**DBP**	**YP**	**GL**	**GL.DE**	**PD**	**BZ**	**DBP**	**YP**	**GL**	**GL.DE**	**PD**
BZ	796,389	-	-	-	-	-	796,389	4,746	114,958	122,308	119,514	4,900
DBP	333	41,724	-	-	-	-	2,436	41,724	2,049	3,472	1,775	857
YP	14,689	757	416,375	-	-	-	38,170	3,141	416,375	7,988	7,741	330
GL	16,420	792	2,166	413,572	-	-	25,419	4,569	6,546	413,572	43,838	504
GL.DE	16,370	452	2,130	42,861	42,861	-	23,372	1,907	6,128	42,861	42,861	249
PD	62	633	105	50	31	2,351	232	821	207	248	125	2,351

Table 1: Exact and fuzzy match dictionary overlaps. For instance, of 796,389 BZ entries, only 333 find and exact and 2,436 find a similar entry in DBP.

For example, the business entity types we used to derive the regular expressions for Germany include "Gesellschaft bürgerlichen Rechts (GbR)", "Kommanditgesellschaft (KG)", or "Offene Handelsgesellschaft (OHG)".

Step 2 further cleanses the names by removing various special characters, such as "®", "TM" and parentheses.

3: Normalization of company names. In Step 3, we tokenize the company name and "normalize" each token that has a length greater than four characters and is written in all capital letters. This normalization step consists of first lowercasing and then capitalizing each token that matches the aforementioned criterion. As an example, the normalization step would transform "VOLKSWAGEN AG" into "Volkswagen AG" and "BASF INDIA LIMITED" into "BASF India Limited".

4: Country name removal. During fourth step we remove all country names appearing in a company's name using a list of country names and their translations to other languages[9]. Although in general more intricate transformation rules can be created, we found that the ones presented here are sufficient for our purposes.

5: Stemming. Unfortunately, the technique described in the next section, which we employ to verify whether a token sequence is contained in a dictionary has some drawbacks. Using an exact matching strategy to match company names that deviate only slightly from the aliases stored in a dictionary can produce suboptimal results. For example, consider the name "Deutsche Presse Agentur", which can also occur as "Deutschen Presse Agentur", depending on the grammatical context. To mitigate these matching issues, we generate additional aliases by stemming each token in a company's name and all its generated aliases using a German Snowball Stemmer[10]. Using this strategy we generate the alias "Deutsch Press Agentur", which can in turn be used to match both representations of the aforementioned name. Adding the resulting aliases to a dictionary increases the chances to match a slightly varying company name to an entity contained in the dictionary while using an exact match strategy.

Our experiments shall show that using a stemmed dictionary has only a limited impact on the overall system performance. As the concepts of stemming and lemmatization are closely related, we also expect similar performance using a lemmatized dictionary and thus abstain from lemmatization.

5.2 Dictionary and feature construction

For the creation of the dictionary, we decided to use entity dictionaries, solely containing entire entity names, instead of using trigger dictionaries, which consist mostly of simple keywords. Using this approach simplifies the creation of dictionaries, because we need to add only a given company name to the list, instead of manually creating triggers.

[9]https://en.wikipedia.org/wiki/List_of_country_names_in_various_languages

[10]http://snowball.tartarus.org/algorithms/german/stemmer.html

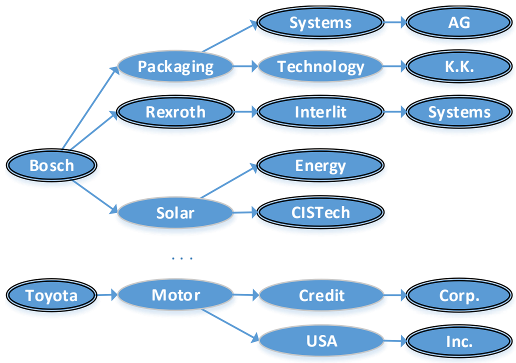

Figure 2: An example of a token trie. Double circles indicate final states.

To make use of the information contained in a dictionary during the CRF training process, we create a feature that encodes whether the currently classified token is part of a company name contained in one of the dictionaries. To efficiently match token sequences in a text against a particular dictionary we tokenize a company's official name and all its aliases and insert the generated tokens, according to their sequence, into a trie data structure. During the insertion, we mark the last inserted token of each token sequence with a flag, denoting the end of the inserted name. In this manner, we insert all company names into the token trie. Figure 2 shows an excerpt of such a token trie after inserting some company names. After its creation, the token trie functions as a finite state automaton (FSA) for efficiently parsing and annotating token sequences in texts as companies.

We perform the matches in a greedy fashion by always choosing the longest possible match. The outlined approach is crucial when using entity dictionaries. In contrast to trigger dictionaries which contain only single tokens, entity dictionaries mark the entire token sequence representing an entity (e.g., "Volkswagen Financial Services GmbH") and therefore need to keep track of their matching state to determine if a match occurred.

6. EXPERIMENTS

In this section we describe our experiments and present the results generated by our system. In Section 6.1 we discuss the setup of our experiments by introducing our test data, annotation policy, and the validation method used. Our overall goal is to evaluate the effect of using dictionaries for NER. Section 6.2 presents the evaluation results of our baseline system *without* the use of dictionaries, as well as a comparative evaluation against the Stanford NER system. The results of using *only* the generated dictionaries to discover companies in our test data are discussed in Section 6.3. Section 6.4 then shows and discusses the results of *integrating* the domain knowledge contained in the dictionaries into our baseline system. Finally, we discuss the case of a *perfect* dictionary in Section 6.5. The performance results in terms of precision, recall and F_1 measure for all analyzed system configurations can be found in Table 2.

6.1 Experimental setup

For the evaluation of our system we randomly selected 1,000 articles for which we could confirm that they contain at least one company mention. We manually annotated these articles by assigning the company-label to each token representing a company mention in the text. We used a very strict annotation policy for tagging the company names in each document; the goal of the policy is to distinguish between mentions referring to a company and mentions referring to related products, persons, or brands. To this end, we considered the context of a company mention to identify a "real" company like BMW, as opposed to a mention appearing as part of another phrase, such as BMW X6, which we did not annotate. In this case, the token X6 identifies the token BMW as part of a product mention. During the annotation process, we discovered and marked 2,351 company mentions in the chosen documents, each consisting of one or more tokens. Links to the news articles of this corpus together with titles and labeled entities are available at https://hpi.de/en/naumann/projects/repeatability/datasets/corpus-comp-ner.html.

To evaluate the performance of our system, we performed a ten-fold cross-validation by splitting the annotated documents into ten folds, each fold containing 900 articles for training and 100 articles for testing. For each fold, we measure precision, recall, and F_1-measure. As usual, the overall performance of the trained model is calculated by averaging the performance metrics over all folds.

We conduct a series of experiments to evaluate our system as well as the impact of different dictionary versions on the systems performance. The results of all experiments are given in Table 2. As our first experiment we compared the performance of our baseline system to the Stanford NER system as described in Section 6.2. Subsequently, we conducted multiple experiments to evaluate the impact of different dictionary versions on the performance of the generated CRF model. Therefore, we generated multiple dictionary versions, which correspond to the rows in Table 2. We created three different dictionary versions for the Bundesanzeiger, GLEIF, GLEIF(DE), Yellow Pages and DBpedia. The first dictionary version contains the original company names obtained from the crawled sources. The second version, marked with "+ Alias", additionally includes all aliases generated by the process described in Section 5.1. The last version, marked with "+ Alias + Stem", also incorporates a stemmed version of each company name and all its generated aliases. We excluded the perfect dictionary from the alias generation process, since it contains the manually tagged colloquial company names. Hence, the approximation of colloquial company names through alias generation is not necessary.

We evaluated each of the generated dictionary versions in two scenarios, illustrated by the two columns "Dict only" and "CRF" in Table 2. In the "Dict only" scenario, described in Section 6.3, we use each dictionary on its own to identify the companies contained in our testset. The "CRF" scenario is discussed in Section 6.4 where we focused on integrating the different dictionary versions into the training process of the CRF and use the generated model to discover company names.

6.2 No dictionaries

We started our experiments by evaluating the baseline configuration introduced in Section 2. Using the basic features mentioned there, we were able to achieve a performance of F_1=80.65% without adding any additional domain knowledge to the system (see Table 2 for details).

We additionally compare our baseline system to the Stanford NER system [6] as one of the most popular NER systems. We used the Stanford system to train a new model on the same training and test documents as for our system, using the configuration suggested on their web-page[11]. Using the resulting model, the Stanford system achieves a slightly better F_1 score of 81.76%. This result is 1.36 percentage points below the precision and 2.68 percentage points above the recall metrics, due to slight variations in the features used.

6.3 Dictionaries only

Next, we used the generated dictionaries on their own to discover the company mentions contained in our testset, as described in Section 5.2. The left, "Dict only" part of Table 2 represents the results of our experiments. The highest precision of 74.23% could be achieved by using the Bundesanzeiger dictionary in its original form. Using the DBpedia dictionary in its original form resulted in the highest F_1-measure value of 51.51%. It is worth noting that using this dictionary in combination with our baseline system and the generated aliases also yielded the best results as described in the following section. Not surprisingly, the highest recall of 72.16% was achieved by combining all dictionaries (except PD) that include the generated aliases and the stemmed name versions.

To understand the impact of alias generation, we compare the average recall of all basic dictionaries, which is 22.92%, with the average recall of all dictionary-extended dictionaries, which is 42.97% (data not shown). The difference of 20,06 percentage points is sufficiently high to justify the use of aliases in principle. Analogously, we analyzed stemming. The average improvement caused by using the dictionaries that include aliases as well as the stemmed names accounted for another increase of 0.21%. However, the improvements of recall are accompanied by an average decrease in precision of 13.46% from the no-aliases to the aliases version, and a further decrease by 14.44 percentage points to a total decrease of −18.28% when including the stemmed versions. In summary, we suggest the use of aliases but refrain from including company name stems in a dictionary.

In addition, we experimented with a dictionary that contained only the company names and their stemmed versions, but no aliases, to assess the impact of stemming on the dictionary-only approach. Here, the precision decreased by 18.94 percentage points while the recall increased only by

[11] http://nlp.stanford.edu/software/crf-faq.shtml

Dictionary	Dict only			CRF		
	P	**R**	**F$_1$**	**P**	**R**	**F$_1$**
Baseline (BL)	–	–	–	91.38%	72.25%	80.65%
Stanford NER	–	–	–	90.02%	74.93%	81.76%
BZ	**74.23**%	3.23%	6.15%	90.90%	75.79%	82.63%
BZ + Alias	16.20%	39.27%	22.91%	91.09%	75.74%	82.63%
BZ + Alias + Stem	6.38%	39.77%	10.98%	90.93%	76.03%	82.78%
GL	34.61%	2.92%	5.37%	90.91%	75.76%	82.62%
GL + Alias	41.71%	50.55%	45.67%	90.78%	77.43%	83.55%
GL + Alias + Stem	18.79%	50.77%	27.39%	90.83%	77.07%	83.36%
GL.DE	68.91%	1.17%	2.29%	90.92%	75.82%	82.66%
GL.DE + Alias	55.78%	21.58%	31.02%	90.97%	76.89%	83.30%
GL.DE + Alias + Stem	39.54%	21.58%	27.85%	90.83%	77.07%	83.36%
YP	16.11%	15.01%	15.53%	91.02%	75.88%	82.73%
YP + Alias	18.34%	21.26%	19.68%	90.92%	75.89%	82.67%
YP + Alias + Stem	7.05%	21.34%	10.58%	90.29%	75.92%	82.72%
DBP	63.13%	43.61%	**51.51**%	**91.25**%	78.54%	84.40%
DBP + Alias	44.18%	53.38%	48.29%	91.11%	**78.82**%	**84.50**%
DBP + Alias + Stem	29.79%	53.47%	38.24%	91.14%	78.76%	84.48%
ALL	20.07%	71.56%	31.33%	90.60%	77.36%	83.43%
ALL + Alias	20.11%	71.80%	31.39%	90.61%	77.33%	83.41%
ALL + Alias + Stem	8.15%	**72.16**%	14.64%	90.94%	76.93%	83.32%
PD (perfect dict.)	81.67%	100.00%	89.90%	94.68%	96.47%	95.56%
PD (perfect dict.) + Stem	81.67%	100.00%	89.90%	94.68%	96.47%	95.56%

Table 2: Results of including different dictionaries into the CRF training process

0.08 percentage points (not shown in Table 2). Hence, we conclude that the stemming of company names has a negative impact on the precision of the dictionary-only approach and does not lead to significant improvement of recall.

When averaging over all the different dictionary versions (without PD) we arrive at an overall performance of 32.39% precision and 36.36% recall. Considering these metrics, it becomes clear that a dictionary-only approach is not sufficient for discovering company names in textual data.

Regarding the perfect dictionary, it is interesting to see that while a recall of 100% could be achieved, the precision reached only a maximum of 81.67%, which is owed to false positives. These are mostly of the form mentioned earlier, where a company name is part of a product name or role description (the VW executive was ...). We expect such errors to be eliminated by the combination with the CRF approach, which makes use of a terms's context.

6.4 Combining dictionaries and CRF

We now discuss the results achieved by combining the domain knowledge contained in the dictionaries and the CRF training process. Overall, we were able to improve the overall performance over the no-dictionary and the dictionary-only approaches, regardless of which dictionary we used. Regarding the right columns of Table 2, we achieved the best results in recall and F_1-measure by using the dictionary generated from the DBpedia including the generated aliases (DBP + Alias) data. Using this dictionary, the system was able to reach an F_1 score of 84.50% with precision and recall values of 91.11% and 78.82%, respectively. By combining the colloquial names already contained in the DBpedia dictionary with the additionally generated alias names, we are able to match more companies than with any of the other dictionaries, explaining our high recall. Interestingly, the initial intuition that combining all dictionaries into one would result in the best performance of our system, turned out not to be true. A more concise dictionary, such as DBpedia, yields the slightly better results.

As we have done in the previous section, we calculated the average change in precision, recall, and F_1-measure. Table 3 shows the average change in performance for gradually evolving our baseline system by including the different dictionary versions. We calculated these values to determine which of the extension steps described in Section 5.1 had the largest impact on system performance. As can be seen, the average change in performance increases significantly moving from the baseline system to a system that uses additional domain knowledge by integrating the basic dictionary version without aliases or stemming. Using additional domain knowledge, the system's precision slightly decreased by 0.45 percentage points, whereas recall and F_1-measure improved on average by 4.28 and 2.43 percentage points, respectively.

Using the dictionary versions containing the generated aliases for each company name, the system gained on average another 0.26 percentage points in F_1-measure. With respect to average precision and recall, the recall increased by 0.49 percentage points while precision slightly decreased by 0.02 percentage points. Due to the alias generation process that condenses a given company name according to the rules described in Section 5.1, we were able to increase the recall while at the same time sustaining precision: we achieved a maximum increase of 6.57 percentage points for recall while the precision decreased only slightly by 0.28% using the DBpedia dictionary including generated alias names. The largest increase of 3.85 percentage points in F_1-measure was also recorded while using the same dictionary. The results suggest that by further improving the alias generation process it should be possible to increase the recall while sustaining high precision.

Regarding dictionaries containing the stemmed version of the original company names and their aliases, we conclude that stemming has only a limited impact; the results produced by including stemmed names are not significantly bet-

Transition	Avg. Precision	Avg. Recall	Avg. F_1
BL $\longrightarrow$ BL + Dict	-0.45%	$+4.28\%$	$+2.43\%$
BL + Dict $\longrightarrow$ BL + Dict + Stem	$+0.05\%$	-0.06%	-0.09%
BL + Dict $\longrightarrow$ BL + Dict + Alias	-0.02%	$+0.49\%$	$+0.26\%$
BL + Dict + Alias $\longrightarrow$ BL + Dict + Alias + Stem	-0.09%	-0.05%	-0.01%

Table 3: Performance change for different dictionary versions, averaged over all dictionaries except PD

ter. For a dictionary version that included only the company names and their stemmed version, the improvements were so low or even negative, that we report only on the average change of using this dictionary in Table 3. As it turned out, the reduction of company names to their stemmed form accounts only for a very limited number of cases. For instance, the airline Lufthansa can be referred to as "Deutsche Lufthansa" or "Deutschen Lufthansa", depending on the grammatical context. By using the common stemmed version ("Deutsch Lufthansa") of these two aliases, it is possible to match both company names. However, such circumstances occur much fewer times for company names than expected.

Because the dictionary feature might add a bias towards labeling known tokens as a company, we also examined how many novel named entities we find, i.e., ones that are not already included in the dictionary. For this experiment, we used each testset in our 10 folds, each consisting of 100 documents not used during the training of the corresponding model. Using the DBpedia including aliases model trained on the remaining 900 documents of each fold, we were able to discover on average 328 company mentions. Examining how many of these company mentions are already contained in the dictionary yielded, that on average 45.85% ($\approx$ 150 companies) of the discovered companies were already included in the dictionary, whereas the remaining 54.15% ($\approx$173) were newly discovered. This shows that although the dictionary feature adds a bias towards already known companies, it is still able to generalize to entities which are not part of the used dictionary.

6.5 Perfect dictionary

To simulate a scenario in which the dictionary can be used on its own to identify the company names in a given text, we use the perfect dictionary. As already mentioned in Section 4, the perfect dictionary consists of all manually annotated company mentions from our test and training sets.

Although using this dictionary yields the highest scores for precision, recall, and F_1-measure, the F_1-measure does not reach 100%. The reason for this behavior can be explained by our strict annotation policy. By using this annotation scheme it becomes hard for the algorithm to avoid producing false positives. Consider the case of recognizing the airline Boeing in the mentions "Boeing" and "Boeing 747". In both cases "Boeing" would be recognized as a company, producing one true positive and one false positive. Hence, a drawback of our system is that the dictionary feature introduces a bias towards companies contained within the dictionary, inducing some false positives if the dictionary feature turns out to be wrong. This problem translates to all other dictionaries that we use. Therefore, we argue that even under ideal circumstances where the dictionary contains all entities that we want to discover, it is not possible to sustain a high precision value by using the dictionary on its own.

Nonetheless, as can be seen by comparing the results in Table 2, using dictionaries to incorporate domain knowledge into the CRF method yields superior results over using them on their own to recognize company names. Considering the average precision, recall, and F_1-measure, the combination of dictionaries and CRF performs significantly better then the pure dictionary approach described in Section 6.3. Integrating the domain knowledge contained in the DBpedia dictionary we achieved a precision of 91.11% and a recall of 78.82%. Regarding the subsequent application or relationship extraction we consider this result as sufficient for recognizing companies in textual data.

7. CONCLUSION & FUTURE WORK

We described a named entity recognition system capable of recognizing companies in textual data with high lexical complexity, achieving a precision of up to 91.11% at a recall of 78.82%. Besides creating the NER system, the particular focus of this work was to analyze the impact of different dictionaries containing company names on the performance of the NER system. Our investigation showed that significant performance improvements can be made by carefully including domain knowledge in the form of dictionaries into the training process of an NER system. On average we were able to increase recall and F_1-measure by 6.57 and 3.85 percentage points, respectively, over our baseline that did not use any external knowledge. Additionally, we showed that applying an alias generation process leads to an increase in recall while sustaining a high precision.

While working with company names, it became increasingly clear that a more sophisticated alias generation process would be needed to handle some of the extremely complex company names. Thus, our future work shall address this issue by including a nested named entity recognition (NNER) step into the preprocessing phase of the dictionary entities. By doing so, we hope to gain semantic knowledge about the constituent parts that form a company name, enabling us to not only increase dictionary quality but to also better determine the *colloquial name* of a company, which in turn would increase the matches of company names in a given text. Another improvement would be to include entities of different entity types (e.g., brands or products) into the token trie, treating them as a blacklist that can then be used to determine whether a sequence of tokens should be marked as a company or not.

The observation that using the smallest dictionary yielded the best results on our newspaper corpus, could indicate that it is important to match the characteristic of the used dictionary with the characteristic of the text corpus. Thus it could be promising to investigate additional corpora, e.g., legal documents, and determine whether dictionaries that are closer to the characteristic of the new corpora also result in a higher system performance.

8. REFERENCES

[1] H. Amini, R. Cont, and A. Minca. Resilience to contagion in financial networks. *Mathematical Finance*, 26(2):329–365, 2016.

[2] D. Benikova, C. Biemann, M. Kisselew, and S. Padó. Germeval 2014 named entity recognition shared task: Companion paper. In *Proceedings of the KONVENS Workshop GermEval Shared Task on Named Entity Recognition*, 2014.

[3] L. Chiticariu, R. Krishnamurthy, Y. Li, F. Reiss, and S. Vaithyanathan. Domain adaptation of rule-based annotators for named-entity recognition tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2010.

[4] W. W. Cohen and S. Sarawagi. Exploiting dictionaries in named entity extraction: Combining semi-Markov extraction processes and data integration methods. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2004.

[5] M. Faruqui and S. Padó. Training and evaluating a german named entity recognizer with semantic generalization. In *Proceedings of the Conference on Natural Language Processing (KONVENS)*, 2010.

[6] J. R. Finkel, T. Grenager, and C. D. Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*, 2005.

[7] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. In *Proceedings of the Conference on Natural Language Learning at HLT-NAACL*, 2003.

[8] R. Grishman and B. Sundheim. Message Understanding Conference – 6: A brief history. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 1996.

[9] C. Hänig, S. Bordag, and S. Thomas. Modular classifier ensemble architecture for named entity recognition on low resource systems. In *Proceedings of the KONVENS Workshop GermEval Shared Task on Named Entity Recognition*, 2014.

[10] M. Hermann, M. Hochleitner, S. Kellner, S. Preissner, and D. Zhekova. Nessy: A hybrid approach to named entity recognition for German. In *Proceedings of the KONVENS Workshop GermEval Shared Task on Named Entity Recognition*, 2014.

[11] J. Kazama and K. Torisawa. Exploiting Wikipedia as external knowledge for named entity recognition. *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.

[12] V. Krishnan and C. D. Manning. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *Proceedings of the International Conference on Computational Linguistics (COLING) and Annual Meeting of the Association for Computational Linguistics (ACL)*, 2006.

[13] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the International Conference on Machine Learning (ICML)*, 2001.

[14] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia – a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*, 2014.

[15] A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Conference on Natural Language Learning at HLT-NAACL*, 2003.

[16] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30, 2007.

[17] N. Okazaki and J. Tsujii. Simple and efficient algorithm for approximate dictionary matching. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2010.

[18] L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Conference on Computational Natural Language Learning*, 2009.

[19] N. Reimers, J. Eckle-Kohler, C. Schnober, J. Kim, and I. Gurevych. GermEval-2014: Nested named entity recognition with neural networks. *Proceedings of the KONVENS Workshop GermEval Shared Task on Named Entity Recognition*, 2014.

[20] P. Schüller. MoSTNER: Morphology-aware split-tag german ner with factorie. *Proceedings of the KONVENS Workshop GermEval Shared Task on Named Entity Recognition*, 2014.

[21] S. Sekine and C. Nobata. Definition, dictionaries and tagger for extended named entity hierarchy. *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, 2004.

[22] R. K. Srihari. A hybrid approach for named entity and sub-type tagging. In *Proceedings of the Applied Natural Language Processing Conference*, 2000.

[23] E. F. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task. In *Proceedings of the Conference on Natural Language Learning at HLT-NAACL*, 2003.

[24] A. Toral and R. Muñoz. A proposal to automatically build and maintain gazetteers for named entity recognition by using Wikipedia. *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2006.

[25] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, 2003.

[26] P. Watrin, L. de Viron, D. Lebailly, M. Constant, and S. Weiser. Named entity recognition for german using conditional random fields and linguistic resources. *Workshop Proceedings of the KONVENS*, 2014.

[27] G. Zhou and J. Su. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*, 2002.