Piggyback: Using Search Engines for Robust Cross-Domain Named Entity Recognition

Stefan Rüd
Institute for NLP
University of Stuttgart
Germany

Massimiliano Ciaramita
Google Research
Zürich
Switzerland

Jens Müller and **Hinrich Schütze**
Institute for NLP
University of Stuttgart
Germany

Abstract

We use search engine results to address a particularly difficult cross-domain language processing task, the adaptation of named entity recognition (NER) from news text to web queries. The key novelty of the method is that we submit a token with context to a search engine and use similar contexts in the search results as additional information for correctly classifying the token. We achieve strong gains in NER performance on news, in-domain and out-of-domain, and on web queries.

1 Introduction

As statistical Natural Language Processing (NLP) matures, NLP components are increasingly used in real-world applications. In many cases, this means that some form of cross-domain adaptation is necessary because there are distributional differences between the labeled training set that is available and the real-world data in the application. To address this problem, we propose a new type of features for NLP data, features extracted from *search engine results*. Our motivation is that search engine results can be viewed as a *substitute for the world knowledge* that is required in NLP tasks, but that can only be extracted from a standard training set or precompiled resources to a limited extent. For example, a named entity (NE) recognizer trained on news text may tag the NE *London* in an out-of-domain web query like *London Klondike gold rush* as a location. But if we train the recognizer on features derived from search results for the sentence to be tagged, correct classification as person is possible. This is because the search results for *London Klondike gold rush* contain snippets in which the first name *Jack* precedes *London*; this is a sure indicator of a last name and hence an NE of type person.

We call our approach *piggyback* and search result-derived features *piggyback features* because we piggyback on a search engine like Google for solving a difficult NLP task.

In this paper, we use piggyback features to address a particularly hard cross-domain problem, the application of an NER system trained on news to web queries. This problem is hard for two reasons. First, the most reliable cue for NEs in English, as in many languages, is *capitalization.* But queries are generally lowercase and even if uppercase characters are used, they are not consistent enough to be reliable features. Thus, applying NER systems trained on news to web queries requires a robust cross-domain approach.

News to queries adaptation is also hard because queries provide *limited context* for NEs. In news text, the first mention of a word like *Ford* is often a fully qualified, unambiguous name like *Ford Motor Corporation* or *Gerald Ford*. In a short query like *buy ford* or *ford pardon*, there is much less context than in news. The lack of context and capitalization, and the noisiness of real-world web queries (tokenization irregularities and misspellings) all make NER hard. The low annotator agreement we found for queries (Section 5) also confirms this point.

The correct identification of NEs in web queries can be crucial for providing relevant pages and ads to users. Other domains have characteristics similar to web queries, e.g., automatically transcribed speech, social communities like Twitter, and SMS. Thus, NER for short, noisy text fragments, in the absence of capitalization, is of general importance.

NER performance is to a large extent determined by the quality of the feature representation. Lexical, part-of-speech (PoS), shape and gazetteer features are standard. While the impact of different types of features is well understood for standard NER, fundamentally different types of features can be used when leveraging search engine results. Returning to the NE *London* in the query *London Klondike gold rush*, the feature "proportion of search engine results in which a first name precedes the token of interest" is likely to be useful in NER. Since using search engine results for cross-domain robustness is a new approach in NLP, the design of appropriate features is crucial to its success. A significant part of this paper is devoted to feature design and evaluation.

This paper is organized as follows. Section 2 discusses related work. We describe standard NER features in Section 3. One main contribution of this paper is the large array of piggyback features that we propose in Section 4. We describe the data sets we use and our experimental setup in Sections 5–6. The results in Section 7 show that piggyback features significantly increase NER performance. This is the second main contribution of the paper. We discuss challenges of using piggyback features – due to the cost of querying search engines – and present our conclusions and future work in Section 8.

2 Related work

Barr et al. (2008) found that capitalization of NEs in web queries is inconsistent and not a reliable cue for NER. Guo et al. (2009) exploit query logs for NER in queries. This is also promising, but the context in search results is richer and potentially more informative than that of other queries in logs.

The insight that search results provide useful additional context for natural language expressions is not new. Perhaps the oldest and best known application is pseudo-relevance feedback which uses words and phrases from search results for query expansion (Rocchio, 1971; Xu and Croft, 1996). Search counts or search results have also been used for sentiment analysis (Turney, 2002), for transliteration (Grefenstette et al., 2004), candidate selection in machine translation (Lapata and Keller, 2005), text similarity measurements (Sahami and Heilman, 2006), incorrect parse tree filtering (Yates et al., 2006), and paraphrase evaluation (Fujita and Sato, 2008). The specific NER application we address is most similar to the work of Farkas et al. (2007), but they mainly used frequency statistics as opposed to what we view as the main strength of search results: the ability to get additional contextually similar uses of the token that is to be classified.

Lawson et al. (2010), Finin et al. (2010), and Yetisgen-Yildiz et al. (2010) investigate how to best use Amazon Mechanical Turk (AMT) for NER. We use AMT as a tool, but it is not our focus.

NLP settings where training and test sets are from different domains have received considerable attention in recent years. These settings are difficult because many machine learning approaches assume that source and target are drawn from the same distribution; this is not the case if they are from different domains. Systems applied out of domain typically incur severe losses in accuracy; e.g., Poibeau and Kosseim (2000) showed that newswire-trained NER systems perform poorly when applied to email data (a drop of F_1 from .9 to .5). Recent work in machine learning has made substantial progress in understanding how cross-domain features can be used in effective ways (Ben-David et al., 2010). The development of such features however is to a large extent an empirical problem. From this perspective, one of the most successful approaches to adaptation for NER is based on generating shared feature representations between source and target domains, via unsupervised methods (Ando, 2004; Turian et al., 2010). Turian et al. (2010) show that adapting from CoNLL to MUC-7 (Chinchor, 1998) data (thus between different newswire sources), the best unsupervised feature (Brown clusters) improves F_1 from .68 to .79. Our approach fits within this line of work in that it empirically investigates features with good cross-domain generalization properties. The main contribution of this paper is the design and evaluation of a novel family of features extracted from the largest and most up-to-date repository of world knowledge, the web.

Another source of world knowledge for NER is Wikipedia: Kazama and Torisawa (2007) show that pseudocategories extracted from Wikipedia help for in-domain NER. Cucerzan (2007) uses Wikipedia and web search frequencies to improve NE disambiguation, including simple web search frequencies

	BASE: lexical and input-text part-of-speech features	
1	WORD(k,i)	binary: $w_k = w^i$
2	POS(k,t)	binary: w_k has part-of-speech t
3	SHAPE(k,i)	binary: w_k has (regular expression) shape regexp_i
4	PREFIX(j)	binary: w_0 has prefix j (analogously for suffixes)
	GAZ: gazetteer features	
5	GAZ-B$_l$(k,i)	binary: w_k is the initial word of a phrase, consisting of l words, whose gaz. category is i
6	GAZ-I$_l$(k,i)	binary: w_k is a non-initial word in a phrase, consisting of l words, whose gaz. category is i
	URL: URL features	
7	URL-SUBPART	N(w_0 is substring of a URL)/N(URL)
8	URL-MI(PER)	$1/N$(URL-parts)$\sum$[[$p \in$ URL-parts]] 3MI$_u(p,\mathrm{PER})$–MI$_u(p,\mathrm{O})$–MI$_u(p,\mathrm{ORG})$–MI$_u(p,\mathrm{LOC})$
	LEX: local lexical features	
9	NEIGHBOR(k)	$1/N(k$-neighbors) $\sum$[[$v \in k$-neighbors]] log[NE-BNC(v,k)/OTHER-BNC(v,k)]
10	LEX-MI(PER,d)	$1/N$(d-words) $\sum$[[$v \in$ d-words]] 3MI$_d(v,\mathrm{PER})$–MI$_d(v,\mathrm{O})$–MI$_d(v,\mathrm{ORG})$–MI$_d(v,\mathrm{LOC})$
	BOW: bag-of-word features	
11	BOW-MI(PER)	$1/N$(bow-words)$\sum$[[$v \in$ bow-words]] 3MI$_b(v,\mathrm{PER})$–MI$_b(v,\mathrm{O})$–MI$_b(v,\mathrm{ORG})$–MI$_b(v,\mathrm{LOC})$
	MISC: shape, search part-of-speech, and title features	
12	UPPERCASE	N(s_0 is uppercase)/N(s_0)
13	ALLCAPS	N(s_0 is all-caps)/N(s_0)
14	SPECIAL	binary: w_0 contains special character
15	SPECIAL-TITLE	N(s_{-1} or s_1 in title contains special character)/(N(s_{-1})+N(s_1))
16	TITLE-WORD	N(s_0 occurs in title)/N(title)
17	NOMINAL-POS	N(s_0 is tagged with nominal PoS)/N(s_0)
18	CONTEXT(k)	N(s_k is typical neighbor at position k of named entity)/N(s_0)
19	PHRASE-HIT(k)	N($w_k = s_k$, i.e., word at position k occurs in snippet)/N(s_0)
20	ACRONYM	N($w_{-1}w_0$ or w_0w_1 or $w_{-1}w_0w_1$ occur as acronym)/N(s_0)
21	EMPTY	binary: search result is empty

Table 1: NER features used in this paper. BASE and GAZ are standard features. URL, LEX, BOW and MISC are piggyback (search engine-based) features. See text for explanation of notation. The definitions of URL-MI, LEX-MI, and BOW-MI for LOC, ORG and O are analogous to those for PER. For better readability, we write $\sum[[x]]$ for $\sum_x$.

for compound entities.

3 Standard NER features

As is standard in supervised NER, we train an NE tagger on a dataset where each token is represented as a feature vector. In this and the following section we present the features used in our study divided in groups. We will refer to the *target token* – the token we define the feature vector for – as w_0. Its left neighbor is w_{-1} and its right neighbor w_1. Table 1 provides a summary of all features.

Feature group BASE. The first class of features, BASE, is standard in NER. The binary feature WORD(k,i) (line 1) is 1 iff w^i, the i^{th} word in the dictionary, occurs at position k with respect to w_0. The dictionary consists of all words in the training set. The analogous feature for part of speech, POS(k,t) (line 2), is 1 iff w_k has been tagged with PoS t, as determined by TnT tagger (Brants, 2000). We also encode surface properties of the word with simple regular expressions, e.g., *x-ray* is encoded as *x-x* and *9/11* as *d/dd* (SHAPE, line 3). For these features, $k \in \{-1, 0, 1\}$. Finally, we encode prefixes and suffixes, up to three characters long, for w_0 (line 4).

Feature group GAZ. Gazetteer features (lines 5 & 6) are an efficient and effective way of building world knowledge into an NER model. A gazetteer is simply a list of phrases that belong to a particular semantic category. We use gazetteers from (i) GATE (Cunningham et al., 2002): countries, first/last names, trigger words; (ii) WordNet: the 46 lexicographical labels (food, location, person etc.); and (iii) Fortune 500: company names. The two gazetteer features are the binary features GAZ-B$_l$(k,i) and GAZ-I$_l$(k,i). GAZ-B$_l$ (resp. GAZ-I$_l$) is 1

iff w_k occurs as the first (resp. non-initial or internal) word in a phrase of length l that the gazetteer lists as belonging to category i where $k \in \{-1, 0, 1\}$.

4 Piggyback features

Feature groups URL, LEX, BOW, and MISC are piggyback features. We produce these by segmenting the input text into overlapping trigrams $w_1w_2w_3$, $w_2w_3w_4$, $w_3w_4w_5$ etc. Each trigram $w_{i-1}w_iw_{i+1}$ is submitted as a query to the search engine. For all experiments we used the publicly accessible Google Web Search API.[1] The search engine returns a *search result* for the query consisting of, in most cases, 10 *snippets*,[2] each of which contains 0, 1 or more *hits* of the search term w_i. We then compute features for the vector representation of w_i based on the snippets. We again refer to the target token and its neighbors (i.e., the search string) as $w_{-1}w_0w_1$. w_0 is the token that is to be classified (PER, LOC, ORG, or O) and the previous word and the next word serve as context that the search engine can exploit to provide snippets in which w_0 is used in the same NE category as in the input text. O is the tag of a token that is neither LOC, ORG nor PER.

In the definition of the features, we refer to the word in the snippet that matches w_0 as s_0, where the match is determined based on edit distance. The word immediately to the left (resp. right) of s_0 in a snippet is called s_{-1} (resp. s_1).

For non-binary features, we first calculate real values and then binarize them into 10 quantile bins.

Feature group URL. This group exploits NE information in URLs. The feature URL-SUBPART (line 7) is the fraction of URLs in the search result containing w_0 as a substring. To avoid spurious matches, we set the feature to 0 if $\text{length}(w_0) \leq 2$.

For URL-MI (line 8), each URL in the search result is split on special characters into parts (e.g., domain and subdomains). We refer to the set of all parts in the search result as URL-parts. The value of $\text{MI}_u(p, \text{PER})$ is computed on the search results of the training set as the mutual information (MI) between (i) w_0 being PER and (ii) p occurring as part of a URL in the search result. MI is defined as follows:

$$\text{MI}(p, \text{PER}) = \sum_{i \in \{\bar{p}, p\}} \sum_{j \in \{\bar{\text{PER}}, \text{PER}\}} P(i, j) \log \frac{P(i, j)}{P(i)P(j)}$$

For example, for the URL-part p = "staff" (e.g., in `bigcorp.com/staff.htm`), $P(\text{staff})$ is the proportion of search results that contain a URL with the part "staff", $P(\text{PER})$ is the proportion of search results where the search token w_0 is PER and $P(\text{staff,PER})$ is the proportion of search results where w_0 is PER and one of the URLs returned by the search engine has part "staff".

The value of the feature URL-MI is the average difference between the MI of PER and the other named entities. The feature is calculated in the same way for LOC, ORG, and O.

Our initial experiments that used binary features for URL parts were not successful. We then designed URL-MI to integrate all URL information specific to an NE class into one measurement in a way that gives higher weight to strong features and lower weight to weak features. The inner sum on line 8 is the sum of the three differences $\text{MI}(\text{PER}) - \text{MI}(\text{O})$, $\text{MI}(\text{PER}) - \text{MI}(\text{ORG})$, and $\text{MI}(\text{PER}) - \text{MI}(\text{LOC})$. Each of the three summands indicates the relative advantage a URL part p gives to PER vs O (or ORG and LOC). By averaging over all URL parts, one then obtains an assessment of the overall strength of evidence (in terms of MI) for the NE class in question.

Feature group LEX. These features assess how appropriate the words occurring in w_0's local contexts in the search result are for an NE class.

For NEIGHBOR (line 9), we calculate for each word v in the British National Corpus (BNC) the count NE-BNC(v, k), the number of times it occurs at position k with respect to an NE; and OTHER-BNC(v, k), the number of times it occurs at position k with respect to a non-NE. We instantiate the feature for $k = -1$ (left neighbor) and $k = 1$ (right neighbor). The value of NEIGHBOR(k) is defined as the average log ratio of NE-BNC(v, k) and OTHER-BNC(v, k), averaged over the set k-neighbors, the set of words that occur at position k with respect to s_0 in the search result.

In the experiments reported in this paper, we use a PoS-tagged version of the BNC, a balanced corpus of 100M words of British English, as a model

[1] Now deprecated in favor of the new Custom Search API.

[2] Less than 0.5% of the queries return fewer than 10 snippets.

of word distribution in general contexts and in NE contexts that is not specific to either target or source domain. In the BNC, NEs are tagged with just one PoS-tag, but there is no differentiation into subcategories. Note that the search engine could be used again for this purpose; for practical reasons we preferred a static resource for this first study where many design variants were explored.

The feature LEX-MI interprets words occurring before or after s_0 as indicators of named entitihood. The parameter d indicates the "direction" of the feature: before or after. $\text{MI}_d(v, \text{PER})$ is computed on the search results of the training set as the MI between (i) w_0 being PER and (ii) v occurring close to s_0 in the search result either to the left ($d = -1$) or to the right ($d = 1$) of s_0. Close refers to a window of 2 words. The value of LEX-MI(PER,d) is then the average difference between the MI of PER and the other NEs. The definition for LEX-MI(PER,d) is given on line 10. The feature is calculated in the same way for LOC, ORG, and O.

Feature group BOW. The features LEX-MI consider a small window for cooccurrence information and distinguish left and right context. For BOW features, we use a larger window and ignore direction. Our aim is to build a bag-of-words representation of the contexts of w_0 in the result snippets.

$\text{MI}_b(v, \text{PER})$ is computed on the search results of the training set as the MI between (i) w_0 being PER and (ii) v occurring anywhere in the search result. The value of BOW-MI(PER) is the average difference between the MI of PER and the other NEs (line 11). The average is computed over all words $v \in$ bow-words that occur in a particular search result. The feature is calculated in the same way for LOC, ORG, and O.

Feature group MISC. We collect the remaining piggyback features in the group MISC.

The UPPERCASE and ALLCAPS features (lines 12&13) compute the fraction of occurrences of w_0 in the search result with capitalization of only the first letter and all letters, respectively. We exclude titles: capitalization in titles is not a consistent clue for NE status.

The SPECIAL feature (line 14) returns 1 iff any character of w_0 is a number or a special character.

NEs are often surrounded by special characters in web pages, e.g., *Janis Joplin - Summertime*. The SPECIAL-TITLE feature (line 15) captures this by counting the occurrences of numbers and special characters in s_{-1} and s_1 in titles of the search result.

The TITLE-WORD feature (line 16) computes the fraction of occurrences of w_0 in the titles of the search result.

The NOMINAL-POS feature (line 17) calculates the proportion of nominal PoS tags (NN, NNS, NP, NPS) of s_0 in the search result, as determined by a PoS tagging of the snippets using TreeTagger (Schmid, 1994).

The basic idea behind the CONTEXT(k) feature (line 18) is that the occurrence of words of certain shapes and with certain parts of speech makes it either more or less likely that w_0 is an NE. For $k = -1$ (the word preceding s_0 in the search result), we test for words that are adjectives, indefinites, possessive pronouns or numerals (partly based on tagging, partly based on a manually compiled list of words). For $k = 1$ (the word following s_0), we test for words that contain numbers and special characters. This feature is complementary to the feature group LEX in that it is based on shape and PoS and does not estimate different parameters for each word.

The feature PHRASE-HIT(-1) (line 19) calculates the proportion of occurrences of w_0 in the search result where the left neighbor in the snippet is equal to the word preceding w_0 in the search string, i.e., $k = -1$: $s_{-1} = w_{-1}$. PHRASE-HIT(1) is the equivalent for the right neighbor. This feature helps identify phrases – search strings containing NEs are more likely to occur as a phrase in search results.

The ACRONYM feature (line 20) computes the proportion of the initials of $w_{-1}w_0$ or w_0w_1 or $w_{-1}w_0w_1$ occurring in the search result. For example, the abbreviation *GM* is likely to occur when searching for *general motors dealers*.

The binary feature EMPTY (line 21) returns 1 iff the search result is empty. This feature enables the classifier to distinguish true zero values (e.g., for the feature ALLCAPS) from values that are zero because the search engine found no hits.

5 Experimental data

In our experiments, we train an NER classifier on an in-domain data set and test it on two different out-of-domain data sets. We describe these data sets in

	CoNLL trn	CoNLL tst	IEER	KDD-D	KDD-T
LOC	4.1	4.1	1.9	11.9	10.6
ORG	4.9	3.7	3.2	8.2	8.3
PER	5.4	6.4	3.8	5.3	5.4
O	85.6	85.8	91.1	74.6	75.7

Table 2: Percentages of NEs in CoNLL, IEER, and KDD.

this section and the NER classifier and the details of the training regime in the next section, Section 6.

As training data for all models evaluated we used the CoNLL 2003 English NER dataset, a corpus of approximately 300,000 tokens of Reuters news from 1992 annotated with person, location, organization and miscellaneous NE labels (Sang and Meulder, 2003). As out-of-domain newswire evaluation data[3] we use the development test data from the NIST 1999 IEER named entity corpus, a dataset of 50,000 tokens of New York Times (NYT) and Associated Press Weekly news.[4] This corpus is annotated with person, location, organization, cardinal, duration, measure, and date labels. CoNLL and IEER are professionally edited and, in particular, properly capitalized news corpora. As capitalization is absent from queries we lowercased both CoNLL and IEER. We also reannotated the lowercased datasets with PoS categories using the retrained TnT PoS tagger (Brants, 2000) to avoid using non-plausible PoS information. Notice that this step is necessary as otherwise virtually no NNP/NNPS categories would be predicted on the query data because the lowercase NEs of web queries never occur in properly capitalized news; this causes an NER tagger trained on standard PoS to underpredict NEs (1–3% positive rate).

The 2005 KDD Cup is a query topic categorization task based on 800,000 queries (Li et al., 2005).[5] We use a random subset of 2000 queries as a source of web queries. By means of simple regular expressions we excluded from sampling queries that looked like urls or emails ($\approx$ 15%) as they are easy to identify and do not provide a significant challenge. We also excluded queries shorter than 10 characters (4%) and longer than 50 characters (2%) to provide annotators with enough context, but not an overly complex task. The annotation procedure was carried out using Amazon Mechanical Turk. We instructed workers to follow the CoNLL 2003 NER guidelines (augmented with several examples from queries that we annotated) and identify up to three NEs in a short text and copy and paste them into a box with associated multiple choice menu with the 4 CoNLL NE labels: LOC, MISC, ORG, and PER. Five workers annotated each query. In a first round we produced 1000 queries later used for development. We call this set KDD-D. We then expanded the guidelines with a few uncertain cases. In a second round, we generated another 1000 queries. This set will be referred to as KDD-T. Because annotator agreement is low on a per-token basis ($\kappa = .30$ for KDD-D, $\kappa = .34$ for KDD-T (Cohen, 1960)), we remove queries with less than 50% agreement, averaged over the tokens in the query. After this filtering, KDD-D and KDD-T contain 777 and 819 queries, respectively. Most of the rater disagreement involves the MISC NE class. This is not surprising as MISC is a sort of place-holder category that is difficult to define and identify in queries, especially by untrained AMT workers. We thus replaced MISC with the null label O. With these two changes, κ was .54 on KDD-D and .64 on KDD-T. This is sufficient for repeatable experiments.[6]

Table 2 shows the distribution of NE types in the 5 datasets. IEER has fewer NEs than CoNLL, KDD has more. PER is about as prevalent in KDD as in CoNLL, but LOC and ORG have higher percentages, reflecting the fact that people search frequently for locations and commercial organizations. These differences between source domain (CoNLL) and target domains (IEER, KDD) add to the difficulty of cross-domain generalization in this case.

6 Experimental setup

Recall that the input features for a token w_0 consist of standard NER features (BASE and GAZ) and features derived from the search result we obtain by

[3] A reviewer points out that we use the terms in-domain and out-of-domain somewhat liberally. We simply use "different domain" as a short-hand for "different distribution" without making any claim about the exact nature of the difference.

[4] `nltk.googlecode.com/svn/trunk/nltk_data`

[5] `www.sigkdd.org/kdd2005/kddcup.html`

[6] The two KDD sets, along with additional statistics on annotator agreement requested by a reviewer, are available at `ifnlp.org/~schuetze/piggyback11`.

running a search for $w_{-1}w_0w_1$ (URL, LEX, BOW, and MISC). Since the MISC NE class is not annotated in IEER and has low agreement on KDD in the experimental evaluation we focus on the four-class (PER, LOC, ORG, O) NER problem on all datasets. We use BIO encoding as in the original CoNLL task (Sang and Meulder, 2003).

								ALL	LOC	ORG	PER
CoNLL											
c1	l	BASE	GAZ					88.8*	91.9	77.9	93.0
c2	l		GAZ	URL		BOW	MISC	86.4*	90.7	74.0	90.9
c3	l	BASE		URL		BOW	MISC	92.3*	93.7	84.8	96.0
c4	l	BASE	GAZ			BOW	MISC	91.1*	93.3	82.2	94.9
c5	l	BASE	GAZ	URL			MISC	92.7*	**94.9**	84.5	95.9
c6	l	BASE	GAZ	URL		BOW		92.3*	94.2	84.4	95.8
c7	l	BASE	GAZ	URL		BOW	MISC	**93.0**	**94.9**	**85.1**	96.4
c8	l	BASE	GAZ	URL	LEX	BOW	MISC	92.9	94.7	84.9	**96.5**
c9	c	BASE	GAZ					92.9	95.3	87.7	94.6
IEER											
i1	l	BASE	GAZ					57.9*	71.0	37.7	59.9
i2	l		GAZ	URL	LEX	BOW	MISC	63.8*	76.2	26.0	75.9
i3	l	BASE		URL	LEX	BOW	MISC	64.9*	71.8	38.3	73.8
i4	l	BASE	GAZ		LEX	BOW	MISC	67.3	76.7	**41.2**	74.6
i5	l	BASE	GAZ	URL		BOW	MISC	67.8	76.7	40.4	75.8
i6	l	BASE	GAZ	URL	LEX		MISC	**68.1**	77.2	36.9	77.8
i7	l	BASE	GAZ	URL	LEX	BOW		66.6*	**77.4**	38.3	73.9
i8	l	BASE	GAZ	URL	LEX	BOW	MISC	**68.1**	**77.4**	36.2	**78.0**
i9	c	BASE	GAZ					68.6*	77.3	52.3	73.1
KDD-T											
k1	l	BASE	GAZ					34.6*	48.9	19.2	34.7
k2	l		GAZ	URL	LEX		MISC	40.4*	52.1	15.4	50.4
k3	l	BASE		URL	LEX		MISC	40.9*	50.0	20.1	48.0
k4	l	BASE	GAZ		LEX		MISC	41.6*	55.0	**25.2**	45.2
k5	l	BASE	GAZ	URL			MISC	43.0	**57.0**	15.8	50.9
k6	l	BASE	GAZ	URL	LEX			40.7*	55.5	15.8	42.9
k7	l	BASE	GAZ	URL	LEX		MISC	**43.8**	56.4	17.0	52.0
k8	l	BASE	GAZ	URL	LEX	BOW	MISC	**43.8**	56.5	17.4	**52.3**

Table 3: Evaluation results. l = text lowercased, c = original capitalization preserved. ALL scores significantly different from the best results for the three datasets (lines c7, i8, k7) are marked $*$ (see text).

We use SuperSenseTagger (Ciaramita and Altun, 2006)[7] as our NER tagger. It is a first-order conditional HMM trained with the perceptron algorithm (Collins, 2002), a discriminative model with excellent efficiency-performance trade-off (Sha and Pereira, 2003). The model is regularized by averaging (Freund and Schapire, 1999). For all models we used an appropriate development set for choosing the only hyperparameter, T, the number of training iterations on the source data. T must be tuned separately for each evaluation because different target domains have different overfitting patterns.

We train our NER system on an 80% sample of the CoNLL data. For our *in-domain* evaluation, we tune T on a 10% development sample of the CoNLL data and test on the remaining 10%. For our *out-of-domain* evaluation, we use the IEER and KDD test sets. Here T is tuned on the corresponding development sets. Since we do not train on IEER and KDD, these two data sets do not have training set portions. For each data set, we perform 63 runs, corresponding to the $2^6 - 1 = 63$ different non-empty combinations of the 6 feature groups. We report average F_1, generated by five-trial training and evaluation, with random permutations of the training data. We compute the scores using the original CoNLL phrase-based metric (Sang and Meulder, 2003). As a benchmark we use the baseline model with gazetteer features (BASE and GAZ). The robustness of this simple approach is well documented; e.g., Turian et al. (2010) show that the baseline model (gazetteer features without unsupervised features) produces an F_1 of .778 against .788 of the best unsupervised word representation feature.

7 Results and discussion

Table 3 summarizes the experimental results. In each column, the best numbers within a dataset for the "lowercased" runs are bolded (see below for discussion of the "capitalization" runs on lines c9 and i9). For all experiments, we selected a subset of the combinations of the feature groups. This subset always includes the best results and a number of other combinations where feature groups are added to or removed from the optimal combination.

Results for the CoNLL test set show that the 5 feature groups without LEX achieve optimal performance (line c7). Adding LEX improves performance on PER, but decreases overall performance (line c8). Removing GAZ, URL, BOW and MISC

[7] `sourceforge.net/projects/supersensetag`

from line c7, causes small comparable decreases in performance (lines c3–c6). These feature groups seem to have about the same importance in this experimental setting, but leaving out BASE decreases F_1 by a larger 6.6% (lines c7 vs c2).

The main result for CoNLL is that using piggyback features (line c7) improves F_1 of a standard NER system that uses only BASE and GAZ (line c1) by 4.2%. Even though the emphasis of this paper is on cross-domain robustness, we can see that our approach also has clear in-domain benefits.

The baseline in line c1 is the "lowercase" baseline as indicated by "l". We also ran a "capitalized" baseline ("c") on text with the original capitalization preserved and PoS-tagged in this unchanged form. Comparing lines c7 and c9, we see that piggyback features are able to recover all the performance that is lost when proper capitalization is unavailable. Lin and Wu (2009) report an F_1 score of 90.90 on the original split of the CoNLL data. Our F_1 scores $> 92\%$ can be explained by a combination of randomly partitioning the data and the fact that the four-class problem is easier than the five-class problem LOC-ORG-PER-MISC-O.

We use the t-test to compute significance on the two sets of five F_1 scores from the two experiments that are being compared (two-tailed, $p < .01$ for $t > 3.36$).[8] CoNLL scores that are significantly different from line c7 are marked with $*$.

For IEER, the system performs best for all six feature groups (line i8). Runs significantly different from i8 are marked $*$. When URL, LEX and BOW are removed from the set, performance does not decrease, or only slightly (lines i4, i5, i6), indicating that these three feature groups are least important. In contrast, there is significant evidence for the importance of BASE, GAZ, and MISC: removing them decreases performance by at least 1% (lines i2, i3, i7). The large increase of ORG F_1 when URL is not used is surprising (41.2% on line i4, best performance). The reason seems to be that URL features (and LEX to a lesser extent) do not generalize for ORG. Locations like *Madrid* in CoNLL are frequently tagged ORG when they refer to sports clubs like *Real Madrid*. This is rare in IEER and KDD.

[8] We make the assumption that the distribution of F_1 scores is approximately normal. See Cohen (1995), Noreen (1989) for a discussion of how this affects the validity of the t-test.

Compared to standard NER (using feature groups BASE and GAZ), our combined feature set achieves a performance that is by more than 10% higher (lines i8 vs i1). This demonstrates that piggyback features have robust cross-domain generalization properties. The comparison of lines i8 and i9 confirms that the features effectively compensate for the lack of capitalization, and perform almost as well as (although statistically worse than) a model trained on capitalized data.

The best run on KDD-D was the run with feature groups BASE, GAZ, URL, LEX and MISC. On line k7, we show results for this run for KDD-T and for runs that differ by one feature group (lines k2–k6, k8).[9] The overall best result (43.8%) is achieved when using all feature groups (line k8). Omitting BOW results in the same score for ALL (line k7). Apparently, the local LEX features already capture most useful cooccurrence information and looking at a wider window (as implemented by BOW) is of limited utility. On lines k2–k6, performance generally decreases on ALL and the three NE classes when dropping one of the five feature groups on line k7. One notable exception is an increase for ORG when feature group URL is dropped (line k4, 25.2%, the best performance for ORG of all runs). This is in line with our discussion of the same effect on IEER.

The key take-away from our results on KDD-T is that piggyback features are again (as for IEER) significantly better than standard feature groups BASE and GAZ. Search engine based adaptation has an advantage of 9.2% compared to standard NER (lines k7 vs k1). An F_1 below 45% may not yet be good enough for practical purposes. But even if additional work is necessary to boost the scores further, our model is an important step in this direction.

The low scores for KDD-T are also partially due to our processing of the AMT data. Our selection procedure is biased towards short entities whereas CoNLL guidelines favor long NEs. We can address this by forcing AMT raters to be more consistent with the CoNLL guidelines in the future.

We summarize the experimental results as follows. Piggyback features consistently improve NER for non-well-edited text when used together with standard NER features. While relative improve-

[9] KDD-D F_1 values were about 1% higher than for KDD-T.

ment due to piggyback features increases as out-of-domain data become more different from the in-domain training set, performance declines in absolute terms from .930 (CoNLL) to .681 (IEER) and .438 (KDD-T).

8 Conclusion

Robust cross-domain generalization is key in many NLP applications. In addition to surface and linguistic differences, differences in world knowledge pose a key challenge, e.g., the fact that *Java* refers to a location in one domain and to coffee in another. We have proposed a new way of addressing this challenge. Because search engines attempt to make optimal use of the context a word occurs in, hits shown to the user usually include other uses of the word in semantically similar snippets. These snippets can be used as a more robust and domain-independent representation of the context of the word/phrase than what is available in the input text.

Our first contribution is that we have shown that this basic idea of using search engines for robust domain-independent feature representations yields solid results for one specific NLP problem, NER. Piggyback features achieved an improvement of F_1 of about 10% compared to a baseline that uses BASE and GAZ features. Even in-domain, we were able to get a smaller, but still noticeable improvement of 4.2% due to piggyback features. These results are also important because there are many application domains with noisy text without reliable capitalization, e.g., automatically transcribed speech, tweets, SMS, social communities and blogs.

Our second contribution is that we address a type of NER that is of particular importance: NER for web queries. The query is the main source of information about the user's information need. Query analysis is important on the web because understanding the query, including the subtask of NER, is key for identifying the most relevant documents and the most relevant ads. NER for domains like Twitter and SMS has properties similar to web queries.

A third contribution of this paper is the release of an annotated dataset for web query NER. We hope that this dataset will foster more research on cross-domain generalization and domain adaptation – in particular for NER – and the difficult problem of web query understanding.

This paper is about cross-domain generalization. However, the general idea of using search to provide rich context information to NLP systems is applicable to a broad array of tasks. One of the main hurdles that NLP faces is that the single context a token occurs in is often not sufficient for reliable decisions, be they about attachment, disambiguation or higher-order semantic interpretation. Search makes dozens of additional relevant contexts available and can thus overcome this bottleneck. In the future, we hope to be able to show that other NLP tasks can also benefit from such an enriched context representation.

Future work. We used a web search engine in the experiments presented in this paper. Latencies when using one of the three main commercial search engines Bing, Google and Yahoo! in our scenario range from 0.2 to 0.5 seconds per token. These execution times are prohibitive for many applications. Search engines also tend to limit the number of queries per user and IP address. To gain widespread acceptance of the piggyback idea of using search results for robust NLP, we therefore must explore alternatives to search engines.

In future work, we plan to develop more efficient methods of using search results for cross-domain generalization to avoid the cost of issuing a large number of queries to search engines. Caching will be of obvious importance in this regard. Another avenue we are pursuing is to build a specialized search system for our application in a way similar to Cafarella and Etzioni (2005). While we need good coverage of a large variety of domains for our approach to work, it is not clear how big the index of the search engine must be for good performance. Conceivably, collections much smaller than those indexed by major search engines (e.g., the Google 1T 5-gram corpus or ClueWeb09) might give rise to features with similar robustness properties. It is important to keep in mind, however, that one of the key factors a search engine allows us to leverage is the notion of relevance which might not be always possible to model as accurately with other data.

Acknowledgments. This research was funded by a Google Research Award. We would like to thank Amir Najmi, John Blitzer, Richárd Farkas, Florian Laws, Slav Petrov and the anonymous reviewers for their comments.

References

Rie Kubota Ando. 2004. Exploiting unannotated corpora for tagging and chunking. In *ACL, Companion Volume*, pages 142–145.

Cory Barr, Rosie Jones, and Moira Regelson. 2008. The linguistic structure of English web-search queries. In *EMNLP*, pages 1021–1030.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning*, 79:151–175.

Thorsten Brants. 2000. TnT – A statistical part-of-speech tagger. In *ANLP*, pages 224–231.

Michael J. Cafarella and Oren Etzioni. 2005. A search engine for natural language applications. In *WWW*, pages 442–452.

Nancy A. Chinchor, editor. 1998. *Proceedings of the Seventh Message Understanding Conference*. NIST.

Massimiliano Ciaramita and Yasemin Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 594–602.

Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Paul R. Cohen. 1995. *Empirical methods for artificial intelligence*. MIT Press, Cambridge, MA, USA.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, pages 1–8.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *EMNLP-CoNLL*, pages 708–716.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *ACL*, pages 168–175.

Richárd Farkas, György Szarvas, and Róbert Ormándi. 2007. Improving a state-of-the-art named entity recognition system using the world wide web. In *Industrial Conference on Data Mining*, pages 163–172.

Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in twitter data with crowdsourcing. In *NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88.

Yoav Freund and Robert E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296.

Atsushi Fujita and Satoshi Sato. 2008. A probabilistic model for measuring grammaticality and similarity of automatically generated paraphrases of predicate phrases. In *COLING*, pages 225–232.

Gregory Grefenstette, Yan Qu, and David A. Evans. 2004. Mining the web to create a language model for mapping between English names and phrases and Japanese. In *Web Intelligence*, pages 110–116.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *SIGIR*, pages 267–274.

Jun'ichi Kazama and Kentaro Torisawa. 2007. Exploiting Wikipedia as external knowledge for named entity recognition. In *EMNLP-CoNLL*, pages 698–707.

Mirella Lapata and Frank Keller. 2005. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1):1–31.

Nolan Lawson, Kevin Eustice, Mike Perkowitz, and Meliha Yetisgen-Yildiz. 2010. Annotating large email datasets for named entity recognition with mechanical turk. In *NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 71–79.

Ying Li, Zijian Zheng, and Honghua (Kathy) Dai. 2005. KDD CUP 2005 report: Facing a great challenge. *SIGKDD Explorations Newsletter*, 7:91–99.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1030–1038.

Eric W. Noreen. 1989. *Computer-Intensive Methods for Testing Hypotheses : An Introduction*. Wiley-Interscience.

Thierry Poibeau and Leila Kosseim. 2000. Proper name extraction from non-journalistic texts. In *CLIN*, pages 144–157.

J. J. Rocchio. 1971. Relevance feedback in information retrieval. In Gerard Salton, editor, *The Smart Retrieval System – Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall.

Mehran Sahami and Timothy D. Heilman. 2006. A web-based kernel function for measuring the similarity of short text snippets. In *WWW*, pages 377–386.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL 2003 Shared Task*, pages 142–147.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 134–141.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *ACL*, pages 384–394.

Peter D. Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *ACL*, pages 417–424.

Jinxi Xu and W. Bruce Croft. 1996. Query expansion using local and global document analysis. In *SIGIR*, pages 4–11.

Alexander Yates, Stefan Schoenmackers, and Oren Etzioni. 2006. Detecting parser errors using web-based semantic filters. In *EMNLP*, pages 27–34.

Meliha Yetisgen-Yildiz, Imre Solti, Fei Xia, and Scott Russell Halgrim. 2010. Preliminary experience with Amazon's Mechanical Turk for annotating medical named entities. In *NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 180–183.