

Predicting Extraction Performance using Context Language Models

Eugene Agichtein Silviu Cucerzan
Microsoft Research, Redmond, WA, USA
{eugeneag, silviu}@microsoft.com

ABSTRACT

Exploiting lexical and semantic relationships in text can dramatically improve information retrieval accuracy. Most notably, named entities and relations between entities are crucial for effective question answering and other information retrieval tasks. Unfortunately, the success in extracting these relationships can vary for different domains and document collections. Predicting extraction performance is an important step towards integration of information extraction technology for high accuracy information retrieval. In this paper, we present a general language modeling method for quantifying the difficulty of information extraction tasks. We demonstrate the viability of our approach by predicting extraction performance of two real world tasks, Named Entity Recognition and Relation Extraction.

Categories and Subject Descriptors

H.3.1 [INFORMATION STORAGE AND RETRIEVAL]: Information Search and Retrieval

General Terms Algorithms, Experimentation.

Keywords Language modeling, information extraction, named entity extraction, relation extraction, context language modeling.

1. OVERVIEW

The vast majority of text available online is still primarily accessible via keyword matching at the document level. Unfortunately, this approach largely ignores the underlying lexical and semantic relationships between terms, which can be exploited to answer questions, and, more generally, to better satisfy the informational needs of users. The retrieval relevance could be improved if we detect meaningful terms (e.g., named entities such as dates, persons, organizations, and locations), and related entities (e.g., pairs of entities such as “*person’s birth date*” and “*person who invented a device*”) and use them to answer questions directly.

However, real collections can exhibit properties that make them difficult for information extraction tasks. At the same time, tuning an information extraction system for a given collection, or porting an information extraction system to a

new language, can require significant human and computational effort. Hence, predicting if an extraction task will be successful (i.e., the required information can be extracted with high accuracy) is extremely important for adapting, deploying, and maintaining information extraction systems, and ultimately, for accurate information retrieval.

The goal of this paper is to develop a lightweight method for *predicting* the accuracy of information extraction for a given task and document collection. This could be an efficient way to estimate the expected success and cost of tuning a full-featured information extraction system *before* running expensive experiments. These predictions can be useful when adapting an IE system to a new task, to a new language, or to a new document collection.

We observe that document collection properties, such as typical text contexts surrounding the entities or relation tuples, can affect difficulty of an extraction task. In turn, we may be able to predict the extraction performance on this task. In this paper, we present a first general approach to use context language models for predicting whether an extraction task will succeed for a given document collection.

More specifically, we will consider two information extraction tasks that are of paramount importance to information retrieval: *Named Entity Recognition*, and *Relation Extraction*.

- Named Entity Recognition (NER) is a task of identifying entities such as “Person”, “Organization”, and “Location” in text. The ability to identify such entities has been established as an important pre-processing task in several areas including information extraction, machine translation, information retrieval, and question answering. NER often serves as an important step in the Relation Extraction task described next.
- Relation Extraction (RE), is a task of identifying semantic relationships between entities in the text, such as “*person’s birth date*”, which relates a person name in the text to a date that is the person’s birth date. Once the tuples for this relation (e.g., <“Albert Einstein”, “14 March 1879”>) are identified, they can be used to directly answer questions such as “When was Albert Einstein born?”

Most state of the art NER and RE systems rely on local context to identify entities or determine the relationship between target entities. In NER, contextual patterns such as “Mr.” or “mayor of” are crucial to hypothesizing occurrences of entities and classifying such identified entities, especially when they are polysemous or of a foreign origin. The local context is also extremely important for the RE task. Intuitively, if the context surrounding the entities of interest for a given relation looks very similar to the general text of the documents (i.e., there are no consistent and obvious “clues” that the entities or relationships of interest are present), then the RE task for that relation will be hard. While NER systems can resort to dictionary lookups in some cases (e.g., for the “Location” entities, dictionaries can be particularly helpful), for others (e.g., people’s names or organizations) high accuracy may not be possible. In contrast, if the text context around entities in the collection tends to contain telltale clues, such as “Mr.” preceding a person name, the extraction task is expected to be easier, and higher accuracy achievable.

Our approach formalizes and exploits this intuitive observation by building two *language models* for a collection—a task-specific *context language model* for the extraction task, and a *background* model for the collection. We can then compare the two models and compute the divergence of the context model from the background model. If the divergence is high (i.e., the context language model is different from the background model), the extraction task is expected to be easier than if the divergence was low (i.e., the context language model is similar to the background language model).

Interestingly, our task-specific language models may be helpful for other applications, including term weighting for information retrieval, and supporting active learning for interactive information extraction. For example, we could derive improved term weights for specific retrieval tasks such as birthday finding. We will discuss other promising future directions of this work in Section 5.

The rest of this paper is organized as follows. In the next section we review related work. In Section 3 we present our formal model and algorithms. In Section 4 we present our initial experimental results for NER and RE tasks over large document collections. In Section 5 we present our conclusions, and discuss the implications and future directions of this work.

2. RELATED WORK

Our work explores language modeling for information extraction and thus touches on areas of information retrieval, information extraction, and language modeling. In this section we briefly review related work in these areas.

Our approach is largely inspired by the work of Cronen-Townsend, Zhou, and Croft [7] on predicting query performance by measuring the similarity of a language model LM_Q derived from the retrieved documents for a query and a language model for the whole target collection of documents LM_{Coll} . Using simple unigram language models, they showed that the relative entropy between the query and collection language models correlates with the average precision in several TREC test collections. In this paper, we apply a similar language modeling technique to the task of predicting information extraction performance.

Language modeling, typically expressed as the problem of predicting the occurrence of a word in text or speech, has been an active area of research in speech recognition, optical character recognition, context-sensitive spelling, and machine translation. An in-depth analysis of this problem in natural language processing is presented in [12], Chapter 6. Language modeling has also been used to improve term weighting in information retrieval (e.g., [13] and others). However, in previous work LM was used as a tool for improving the specific system performance, whereas in our work we attempt to predict performance for general extraction tasks.

Using local context modeling has been previously used for IR tasks [18]. However, our work is different in that we only consider *task-specific* contexts. As our results indicate, using the locality in the overall document collection may not be sufficient, as local context models can become similar to the background model for overall document collection. Our model is similar in spirit to the use of entity language models described in [14] for classifying and retrieving named entities. A related language modeling approach was used for a different problem of predicting the reading difficulty of text for *human* readers [24]. A different approach in [25] uses the co-occurrence graph structure of the examples to predict accuracy of semi-supervised learning for semantic classification of phrases. Our work is complementary, as we present a general approach for modeling the performance of automatic systems on extraction tasks, including both named entity recognition and relation extraction.

For the named entity recognition task, numerous ways of exploiting local context were proposed, from relatively simple character-based models such as [8] and [11] to extremely complex models making use of various lexical, syntactic, morphological, orthographical information, such as [9] and [5]. In this work, we show that we can predict the difficulty of identifying several types of named entities by using relatively simple context language models. This study can be viewed as complementary to Collins’ work [6] on the difficulty of identifying named entity boundaries, regardless of entity type.

Relation extraction systems rely on variety of features (e.g., syntactic, semantic, lexical, co-occurrence), but all depend heavily on context. Once the entities are identified, it is the textual context that expresses the relationship between the entities. Partially supervised relation extraction systems (e.g., [1], [10], [16], [20], and others) rely on the text contexts of example facts to derive extraction patterns.

For relation extraction, the task difficulty was previously analyzed by considering the complexity of the target extraction templates ([21] and [22]). Another promising approach described in [23] modeled the task domain variability by considering the different paraphrases used to express the same information in the text. In contrast, our work quantifies the difference between the contexts around the entities and unrelated text contexts. If the contexts of the example facts are similar to the background text an extraction system is expected to have more difficulty deriving extraction patterns and recognizing the relevant entities.

3. PREDICTING EXTRACTION DIFFICULTY

In this section we describe the general approach we take for modeling the difficulty of an extraction task, and hence the expected performance of an extraction system on the task (Section 3.1). Then, in Section 3.2, we describe the algorithms for computing the language models to make our predictions.

3.1 Model

As we discussed, the textual context, i.e., the local properties of the text surrounding the entities and relations of interest are of crucial importance to extraction accuracy. Intuitively, if the contexts in which the entities occur are very similar to the text contexts where the target entities do not occur, then extraction is expected to be difficult. Otherwise, if there are strong clues in a context, the extraction should be easier and we should expect higher extraction accuracy.

To quantify the notion of context, we use a basic unigram language model, which is essentially a probability distribution over the words in the text’s vocabulary. In this study, we derive this probability distribution from the histogram of words occurring in the local context of target entities by using maximum likelihood estimation. Our purpose is to compare the language model associated with an entity type or relationship LM_C with a background language model for the whole target text, denoted by LM_{BG} . Therefore, no smoothing of these models is necessary. Intuitively, if the background language model for the collection is very similar to the language model constructed from the context of the valid entities then the task is

expected to be hard. Otherwise (if LM_C is very different from LM_{BG}), the task is expected to be easier.

A common way to measure the difference between two probability distributions is relative entropy, also known as the Kullback-Leibler divergence:

$$KL(LM_C \parallel LM_{BG}) = \sum_{w \in V} LM_C(w) \cdot \log \frac{LM_C(w)}{LM_{BG}(w)}$$

In Information Theory, KL-divergence represents the average number of bits wasted by encoding messages drawn from the distribution LM_C using as model the distribution LM_{BG} .

Alternatively, we can measure how different two models are by using cosine similarity, which represents the cosine of the angle between the two language models seen as vectors in a multidimensional space in which each dimension corresponds to one word in the vocabulary:

$$\text{Cosine}(LM_C, LM_{BG}) = \frac{\langle LM_C \cdot LM_{BG} \rangle}{\|LM_{BG}\|_2 \cdot \|LM_C\|_2}$$

The closer the cosine is to 1, the smaller the angle and thus, the more similar the two models. Hence, to measure the difference of the two models LM_C and LM_{BG} we define $CDist$ as:

$$CDist(LM_C \parallel LM_{BG}) = 1 - \text{Cosine}(LM_C, LM_{BG})$$

Which maintains the symmetry with the KL metric, with bigger values indicating larger difference between models.

3.2 Constructing the Language Models

We now describe how to construct a language model for a given extraction task. For clarity, we describe a unigram language model, but our methodology can be extended to higher-order features. For syntax-based extraction systems, we could parse the text and incorporate that information into the model as [4]. However, as we will show experimentally, our initial simple unigram model is sufficient to make useful predictions.

To construct the task-specific context language model LM_C we search the collection for occurrences of valid entities (or relation tuples). While for the NER and RE tasks LM_C is constructed slightly differently (as described below), the overall approach is to consider the text context to be the K words to the right and to the left of the entity in question.

More specifically, the language model for NER is constructed as outlined in Figure 3.1. We scan the document collection D , searching for occurrences of each known entity E_i . When an entity is detected, we add to LM_C up to K terms to the right and to the left of the entity.

```

ConstructNERLanguageModel (Entities  $E$ , Documents  $D$ ,  $K$ )
For each document  $d$  in  $D$ 
  For each entity  $E_i$  in  $E$ 
    if  $E_i$  is present in  $d$ 
      For each instance of  $E_i$  spanning from  $start$  to  $end$ 
        For each term  $w$  in  $d$  [ $start - K$ ], ...,  $d$  [ $start - 1$ ]
          Increment count of  $w$  in  $LM_C$ 
        For each term  $w$  in  $d$  [ $end + 1$ ], ...,  $d$  [ $end + K$ ]
          Increment count of  $w$  in  $LM_C$ 
  Normalize  $LM_C$ 
return  $LM_C$ 

```

Algorithm 3.1: NER Context language model construction.

The algorithm for constructing a task-specific language model for RE is outlined in Figure 3.2. The procedure is similar to the NER algorithm above. We scan the document collection D , searching for occurrences of each known example tuple T_i for the target relation. For this, we search for all attributes of T_i in the text. If all entities are present, we add up to K terms to the right of the leftmost entity, and up to K terms to the left of the rightmost entity to LM_C . If the entities in a relation tuple are close together (i.e., there are fewer than K words separating the entities in the text), we include all the terms separating the entities. Clearly, other variations of this algorithm are possible, and could be explored in future work.

```

ConstructRELanguageModel (Tuples  $T$ , Documents  $D$ ,  $K$ )
For each document  $d$  in  $D$ 
  For each tuple  $T_i=(t_i^1, t_i^2)$  in  $T$ 
    If  $t_i^1$  or  $t_i^2$  not present in  $d$  continue
    For each pair of adjacent instances of  $t_i^1, t_i^2$ 
      occurring at positions  $start$  and  $end$ 
        For each term  $w$  in  $d$  [ $start - K$ ], ...,  $d$  [ $start - 1$ ]
          Increment count of  $w$  in  $LM_C$ 
        For each term  $w$  in  $d$  [ $end + 1$ ], ...,  $d$  [ $end + K$ ]
          Increment count of  $w$  in  $LM_C$ 
  Normalize  $LM_C$ 
return  $LM_C$ 

```

Algorithm 3.2: RE Context language model construction.

Unfortunately, we don't have all the valid entities available (i.e., when predicting whether a task will succeed without going through the complete extraction process). Hence, our model is build based on *sampling* the collection using a small (20-40) sample of the known entities or tuples by providing only these example entities as input to the NER and RE language model construction algorithms above. The sample-based model is expected to be a reasonable approximation of the complete task specific language model.

The background language model, LM_{BG} is derived through maximum likelihood estimation using the word frequencies in each document collection. When we discard stopwords from LM_C we also discard them from LM_{BG} .

In order to interpret the divergence of a task specific language model LM_C from the background language model, we build a reference context language model LM_R (also denoted as RANDOM). We construct LM_R , by taking random samples of words in the vocabulary (excluding stopwords) of the same size as the entity samples. We then use these words input to Algorithm 3.1. Using LM_R we can then compute the "reference" divergence of a context language model from the background model for a given sample size. For large sample sizes, LM_R is expected to approximate the background model. Indeed, Figure 3.1 reports that for larger random word sample sizes, LM_R becomes more similar to the background model, and the divergence steadily decreases.

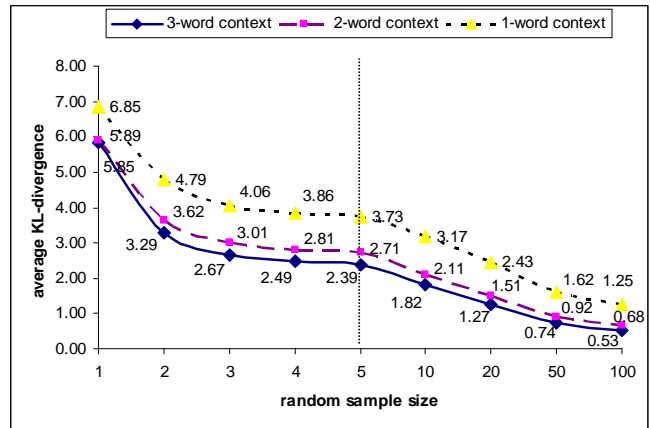


Figure 3.1: The average KL-divergence between the context language models for random samples of words and the background language model.

Constructing the context models LM_C and LM_R can be done efficiently by using any off-the-shelf search engine and considering only the documents retrieved by search for the example entities or tuples, and run Algorithms 3.1 and 3.2 only over these reduced document sets.

Having described constructing the language models for extraction tasks, we now turn to experimental evaluation.

4. EXPERIMENTS

We evaluated our prediction for two real-world tasks: Named Entity Recognition (NER) and Relation Extraction (RE). We first describe the experimental setup (Section 4.1), including the datasets, entity and relation types, and parameter settings we considered, as well as the methods for comparison. Then we describe our experiments for predicting NER difficulty (Section 4.2), followed by our experiments on predicting RE difficulty (Section 4.3).

4.1 Experimental Setup

In order to design a realistic evaluation we focused on two extraction tasks, NER and RE, over large document collections. The overall goal of the experiments is to determine if the language models, constructed from a realistically small sample of the extractions of interest, can make useful predictions about the observed accuracy of the extraction task for that collection.

The document collections used for these experiments are reported in Table 4.1. The Reuters RCV1 documents were drawn from the collection used in the CoNLL 2003 [17] NER shared task evaluation. For the RE experiments, we used a large online encyclopedia document collection.

Task	Collection	Size
NER	Reuters RCV1, 1/100	3,566,125 words
	Reuters RCV1, 1/10	35,639,471 words
RE	Encyclopedia documents	64,187,912 words

Table 4.1: Document collections used in experiments.

For all experiments, we start with a small sample (20-40) of entities or relation tuples, drawn at random from a list of known valid entities or tuples. In Table 4.2 we report the size and composition of the samples used for the experiments.

Task	Sample Extractions (Description)	Size
NER	Location names (LOC)	20
	Miscellaneous named entities (MISC)	20
	Organization names (ORG)	20
	Person names (PER)	20
RE	Person’s birth dates (BORN)	35
	Person’s death dates (DIED)	35
	Person’s inventions (INVENT)	35
	Person’s writings (WROTE)	35

Table 4.2: Entity and relations used in experiments.

To validate our extraction performance predictions for the NER task, we used as reference the top performing systems in the CoNLL shared task competition, which were evaluated over a manually annotated subset of news articles from the same RCV1 corpus as described above. Moreover, we built the samples of named entities by randomly sampling the set of named entities present in the training set provided by the CoNLL competition organizers [17].

To validate our performance predictions for the RE task, we used a simple bootstrapping-based extraction system similar to Snowball [1], which is heavily dependent on both the example entities and the text context in which they appear

to derive extraction patterns. The accuracy was computed by sampling the extracted relations. For comparison, we also report RANDOM, the divergence of the random keyword sample-based language model, LM_R .

In our experiments we explored the following parameters:

- Context size K : number of words to the left and to the right of entity to include as context.
- Divergence metric, $CDist$ or KL : The language model similarity metrics defined in Section 3.1.
- Example set size S : number of randomly drawn entities (or relation tuples). Fixed sample size for each task between 20 and 40.
- Random sample size R : number of randomly drawn terms to estimate the background model. Fixed to match the value of S above for each task.
- *Stopwords*: we analyze two cases, when stopwords (common English words such as prepositions, conjunctions, numerals, etc.) are included the vocabulary and when they are excluded. In both cases, we discard punctuation.

4.2 Predicting NER Difficulty

In order to evaluate the accuracy of our predictions for the difficulty of extracting different types of named entities, we use as reference the accuracy of the top five systems in the CoNLL 2003 shared task competition [17], which are summarized in Table 4.3. It is also worth noting the performance of a baseline system that only identifies and labels entities that occurred in the training set. This baseline system obtains F-measure scores of 80.5 for LOC, 83.5 for MISC, 66.4 for ORG, and 55.2 for PER.

	Florian et al. [9]	Chieu et al. [5]	Klein et al. [11]	Zhang et al. [19]	Carreras et al. [3]	Average
LOC	91.15	91.12	89.98	89.54	89.26	90.21
MISC	80.44	79.16	80.15	75.87	78.54	78.83
ORG	84.67	84.32	80.48	80.46	79.41	81.86
PER	93.85	93.44	90.72	90.44	88.93	91.47
Overall	88.76	88.31	86.31	85.50	85.00	86.77

Table 4.3: F-measures on the Reuters RCV1 collection reported by the top 5 systems participating in the CoNLL 2003 Shared Task competition.

We report results on predicting NER difficulty in Tables 4.4a, 4.4b, and 4.5. The first two tables present the results obtained on a smaller subset of the Reuters corpus (3.5 million words), while the latter shows the results obtained for a bigger subset of the corpus (35 million words). It is remarkable that language models estimated on the smaller corpus are as good predictors as those estimated on a corpus 10-times bigger.

RCV1 1/100, Left/Right Context Size 1, Counting Stopwords

	Sample 1	Sample 2	Sample 3	Average
LOC	1.47	1.54	1.49	1.50
MISC	1.31	2.09	2.29	1.89
ORG	4.36	2.25	4.12	3.57
PER	7.40	4.08	5.28	5.58
RANDOM	1.57	1.24	1.73	1.51

RCV1 1/100, Left/Right Context Size 2, Counting Stopwords

LOC	1.12	1.15	1.11	1.12
MISC	0.94	1.46	1.62	1.34
ORG	3.60	1.85	3.85	3.10
PER	5.71	3.22	4.30	4.41
RANDOM	1.04	0.73	1.00	0.92

RCV1 1/100, Left/Right Context Size 3, Counting Stopwords

LOC	0.92	0.94	0.93	0.93
MISC	0.78	1.18	1.33	1.09
ORG	2.95	1.65	3.45	2.68
PER	5.16	2.80	3.79	3.91
RANDOM	0.87	0.63	0.83	0.78

Table 4.4a: KL-divergence for the context models for random samples of 20 entities/random words and the background language model for a corpus of 3.5 million words, when the language models include stopwords.

RCV1 1/100, Left/Right Context Size 1, Ignoring Stopwords

	Sample 1	Sample 2	Sample 3	Average
LOC	2.44	2.66	2.48	2.52
MISC	2.40	3.49	3.77	3.22
ORG	4.58	4.49	6.74	5.27
PER	9.08	6.23	7.61	7.64
RANDOM	2.35	2.11	2.84	2.43

RCV1 1/100, Left/Right Context Size 2, Ignoring Stopwords

LOC	1.73	1.83	1.80	1.78
MISC	1.67	2.51	2.72	2.30
ORG	3.87	3.45	5.90	4.40
PER	8.03	4.87	5.91	6.27
RANDOM	1.68	1.22	1.62	1.50

RCV1 1/100, Left/Right Context Size 3, Ignoring Stopwords

LOC	1.44	1.50	1.50	1.48
MISC	1.35	1.98	2.16	1.83
ORG	3.28	2.85	5.31	3.81
PER	7.19	4.32	5.36	5.62
RANDOM	1.43	1.05	1.33	1.27

Table 4.4b: KL-divergence for the context models for random samples of 20 entities/random words and the background language model for a corpus of 3.5 million words, when the language models discard stopwords.

Our ranking identifies ORG and PER entities as “easy” to extract entity types and LOC and MISC as hard to extract. These correlate with the results reported by the participants in the CoNLL 2003 Shared Task competition (Table 4.2), with one exception: the LOC entities. We believe this happens for two reasons: first, the location entities in the test set overlap to a large degree with the locations in the training data, as indicated by the performance of the baseline system; second, all systems shown in Table 4.3, except [11] used extensive lists of gazetteers, which were likely to contain most locations that news articles may talk about and thus, covering most of the locations in the test. A drawback of our model is that it does not take into account how easy to identify entities of a certain type based on intrinsic information (e.g., morphology) or gazetteer lists.

RCV1 1/10, Left/Right Context Size 1, Ignoring Stopwords

	Sample 1	Sample 2	Sample 3	Average
LOC	1.65	1.82	1.81	1.76
MISC	1.79	2.75	2.99	2.51
ORG	4.36	2.93	5.48	4.25
PER	7.10	5.04	5.50	5.88
RANDOM	1.98	1.60	2.43	2.00

RCV1 1/10, Left/Right Context Size 2, Ignoring Stopwords

LOC	1.12	1.22	1.26	1.20
MISC	1.16	1.84	2.02	1.67
ORG	3.57	2.18	4.33	3.36
PER	5.95	3.81	4.29	4.68
RANDOM	1.36	0.83	1.24	1.14

RCV1 1/10, Left/Right Context Size 3, Ignoring Stopwords

LOC	0.92	0.99	1.04	0.98
MISC	0.91	1.41	1.55	1.29
ORG	2.94	1.79	3.76	2.83
PER	5.28	3.28	3.75	4.10
RANDOM	1.12	0.68	0.96	0.92

Table 4.5: KL-divergence for the context models for random samples of 20 entities/random words and the background language model for a corpus of 35 million words, when the language models discard stopwords.

Often, our predictions suggest a clear distinction between the four entity types considered. In most cases, the average KL-divergence value for one type of entities is greater than the maximum KL-divergence and smaller than the minimum KL-divergence obtained for any sample of another type of entities.

Tables 4.4a and 4.4b show the contrast between using stopwords in the language model and discarding the stopwords. As expected, the context language models are more similar to the background model when stopwords are included, but in both cases, the conclusions are essentially

Relation \ K	KL (Section 3.1)					$CDist$ (Section 3.1)				
	1	2	3	4	5	1	2	3	4	5
INVENT	4.33	3.76	3.55	3.33	3.3	0.24	0.2	0.16	0.12	0.13
BORN	8.68	7.62	6.86	6.4	6.16	0.86	0.74	0.58	0.54	0.54
DIED	7.72	7.72	6.87	6.49	6.75	0.62	0.66	0.49	0.42	0.41
WROTE	4.47	4.1	3.89	3.77	3.65	0.5	0.38	0.29	0.24	0.12
RANDOM	0.4	0.25	0.17	0.09	0.08	0.24	0.23	0.12	0.02	0.01

Table 4.6: Predicting RE performance for the INVENT, BORN, DIED, and WROTE relations when the language models include stopwords.

the same. This is encouraging, as it shows that this approach may work even for languages for which no lexical information (such as stopwords) is known *a priori*.

4.3 Predicting RE Performance

We now turn to predicting the performance for the relation extraction task (RE). The goal is to predict which relations are “hard” to extract, and which ones are “easy”. Table 4.7 reports the actual extraction accuracy on the RE task using a simple bootstrapping-based information extraction system similar to Snowball [1] and KnowItAll [20]. We report the precision of the facts extracted by the system estimated by sampling 100 facts from the extracted relation instances. As we can see, the BORN and DIED relations are “easy” for the extraction system (exhibiting precision of as high as 97%), whereas INVENT and WROTE are relatively “hard” (exhibiting precision as low as 50%).

Table 4.6 reports the KL and $CDist$ values computed from the models incorporating all words in the contexts. The KL divergence values of the BORN and DIED relations are significantly higher than the KL values for the INVENT and WROTE relations, predicting that the former should have higher accuracy than the latter. Hence, KL correctly identifies “easy” relations vs. “hard” relations to extract. On this task, the $CDist$ divergence values for BORN and DIED are also noticeably higher than the corresponding values for INVENT and WROTE.

$Relation$	$Accuracy$ (%)		$Task Difficulty$
	$strict$	$partial$	
INVENT	0.35	0.64	Hard
BORN	0.73	0.96	Easy
DIED	0.34	0.97	Easy
WROTE	0.12	0.50	Hard

Table 4.7: Precision for the RE task on the Encyclopedia collection for the INVENT, BORN, DIED, and WROTE relations.

Table 4.8 reports the divergence values using the language models built by discarding common English stopwords. As we can see, these models have higher divergence from the background than the models with stopwords included. This is not surprising, as stopwords tend to appear in both generic and task specific contexts. As before, the context models built without stopwords have higher KL divergence from the background model for the “easy” relations than the KL divergence of the context models for the “hard” relations. In contrast, the $CDist$ values for INVENT and DIED models now become more similar. In fact, the $CDist$ values for INVENT are actually higher than the corresponding values for DIED, incorrectly suggesting that the INVENT extraction task is “easy”. Hence, it appears that KL is a more robust predictor of extraction performance.

Relation \ K	KL (Section 3.1)					$CDist$ (Section 3.1)				
	1	2	3	4	5	1	2	3	4	5
INVENT	6.68	5.95	5.69	5.4	5.27	0.74	0.72	0.73	0.72	0.73
BORN	9.4	8.79	8.49	8.1	7.61	0.89	0.86	0.86	0.86	0.86
DIED	9.16	8.88	8.11	7.79	8.1	0.75	0.79	0.66	0.61	0.6
WROTE	6.72	5.95	5.82	5.71	5.48	0.62	0.61	0.61	0.62	0.61
RANDOM	0.36	0.22	0.18	0.13	0.12	0.4	0.24	0.2	0.14	0.13

Table 4.8: Predicting RE performance for INVENT, BORN, DIED, and WROTE relations when the language models discard stopwords.

5. CONCLUSIONS AND FUTURE WORK

We presented a context language modeling approach for predicting extraction performance. We have shown that our approach is effective for predicting extraction accuracy for tasks such as named entity recognition and relation extraction, both crucial for high accuracy and domain-specific information retrieval.

As our experiments indicate, starting with even a small sample of available entities can be sufficient for making a reasonable prediction about extraction accuracy. Our results are particularly encouraging as we consider a relatively simple model that does not require extra information to that typically available to modern NER and RE systems.

Extending our method to use more sophisticated language models (e.g., n -grams) can further improve our predictions. For languages where reliable NLP tools are available, one promising direction would be to incorporate syntactic features, and to apply techniques such as co-reference resolution to build richer and more accurate context language models. Additionally, incorporating gazetteer lists similar to those typically used by the NER systems can further improve prediction accuracy.

Furthermore, our techniques could be applied for building interactive information extraction systems that guide the user by requesting more examples for the extraction tasks predicted to be “hard”.

As our experiments show, context language models localized around entities and relation instances of interest diverge from the document-level language model. Thus, for tasks such as question answering and information extraction, modeling term proximity near entities of interest is a promising direction for improving information retrieval accuracy.

ACKNOWLEDGEMENTS

We thank Eric Brill for his encouragement and valuable suggestions, and the anonymous reviews for their insightful comments.

6. REFERENCES

- [1] E. Agichtein and L. Gravano, Snowball: Extracting Relations from Large Plain-Text Collection, in *ACM DL 2000*
- [2] E. Brill, S. Dumais, and M. Banko. An Analysis of the AskMSR Question-Answering System, in *EMNLP 2002*
- [3] X. Carreras, L. Marquez, and L. Padro, A Simple Named Entity Extractor using AdaBoost, in *CoNLL 2003*
- [4] C. Chelba and F. Jelinek, Exploiting Syntactic Structure for Language Modeling, in *COLING-ACL 1998*
- [5] H. L. Chieu and H. T. Ng, Named Entity Recognition with a Maximum Entropy Approach, in *CoNLL 2003*
- [6] M. Collins, Ranking Algorithms for Named Entity Extraction: Boosting and the Voted Perceptron, in *ACL 2002*
- [7] S. Cronen-Townsend, Y. Zhou, W. B. Croft, Predicting Query Performance, in *SIGIR 2002*
- [8] S. Cucerzan and D. Yarowsky, Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence, in *EMNLP-VLC 1999*
- [9] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang, Named Entity Recognition through Classifier Combination, in *CoNLL 2003*
- [10] R. Jones, A. McCallum, K. Nigam, and E. Riloff, Bootstrapping for Text Learning Tasks, in *IJCAI Workshop on Text Mining: Foundations, Techniques and Applications, 1999*
- [11] D. Klein, J. Smarr, H. Nguyen, and C. Manning, Named Entity Recognition with Character-Level Models, in *CoNLL 2003*
- [12] C. D. Manning and H. Sch tze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999
- [13] J. M. Ponte and W. B. Croft, A Language Modeling Approach to Information, in *SIGIR 1998*
- [14] H. Raghavan, J. Allan, and A. McCallum, An exploration of Entity Models, Collective Classification and Relation descriptions, in *LinkKDD 2004*
- [15] D. Ravichandran and E. Hovy, Learning Surface Text Patterns for a Question Answering System, in *ACL 2002*
- [16] E. Riloff and R. Jones, Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping, in *AAAI 1999*
- [17] E.F. Tjong Kim Sang and F. De Meulder, Introduction to the CoNLL-2003 Shared Task, in *CoNLL 2003*
- [18] J. Xu and W. B. Croft, Improving the effectiveness of information retrieval with local context analysis, in *ACM Transactions on Information Systems, 2000*
- [19] T. Zhang and D. Johnson, A Robust Risk Minimization based Named Entity Recognition System, in *CoNLL 2003*
- [20] O. Etzioni, M. Cafarella D. Downey, S. Kok, A. Popescul, T. Shaked, S. Soderland, D. Weld, and A. Yates, Web-scale information extraction in KnowItAll: preliminary results, in *WWW 2004*
- [21] A. Bagga, Analyzing the complexity of a domain with respect to an information extraction task, in *MUC-7, 1998*
- [22] S. Huttunen, R. Yangarber, and R. Grishman, Complexity of event structure in IE scenarios, in *COLING 2002*
- [23] I. Dagan and O. Glickman, Probabilistic textual entailment: generic applied modeling of language variability, in *Learning Methods for Text Understanding and Mining Workshop, 2004*
- [24] K. Collins-Thompson and J. P. Callan, A language modeling approach to predicting reading difficulty, in *HLT, 2004*
- [25] R. Jones, Semi-supervised Learning on Small Worlds, in *Link Discovery Workshop at KDD, 2004*