Joint Bootstrapping Machines for High Confidence Relation Extraction

Pankaj Gupta\textsuperscript{1,2}, Benjamin Roth\textsuperscript{2}, Hinrich Schütze\textsuperscript{2}

\textsuperscript{1}Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
\textsuperscript{2}CIS, University of Munich (LMU) Munich, Germany

pankaj.gupta@siemens.com | pankaj.gupta@campus.lmu.de
\{beroth, inquiries\}@cis.lmu.de

Abstract

Semi-supervised bootstrapping techniques for relationship extraction from text iteratively expand a set of initial seed instances. Due to the lack of labeled data, a key challenge in bootstrapping is semantic drift: if a false positive instance is added during an iteration, then all following iterations are contaminated. We introduce BREX, a new bootstrapping method that protects against such contamination by highly effective confidence assessment. This is achieved by using entity and template seeds jointly (as opposed to just one as in previous work), by expanding entities and templates in parallel and in a mutually constraining fashion in each iteration and by introducing higher-quality similarity measures for templates. Experimental results show that BREX achieves an $F_1$ that is 0.13 ($0.87$ vs. $0.74$) better than the state of the art for four relationships.

1 Introduction

Traditional semi-supervised bootstrapping relation extractors (REs) such as BREDS (Batista et al., 2015), SnowBall (Agichtein and Gravano, 2000) and DIPRE (Brin, 1998) require an initial set of seed entity pairs for the target binary relation. They find occurrences of positive seed entity pairs in the corpus, which are converted into extraction patterns, i.e., extractors, where we define an extractor as a cluster of instances generated from the corpus. The initial seed entity pair set is expanded with the relationship entity pairs newly extracted by the extractors from the text iteratively. The augmented set is then used to extract new relationships until a stopping criterion is met.

Due to lack of sufficient labeled data, rule-based systems dominate commercial use (Chiticariu et al., 2013). Rules are typically defined by creating patterns around the entities (entity extraction) or entity pairs (relation extraction). Recently, supervised machine learning, especially deep learning techniques (Gupta et al., 2015; Nguyen and Grishman, 2015; Vu et al., 2016a,b; Gupta et al., 2016), have shown promising results in entity and relation extraction; however, they need sufficient hand-labeled data to train models, which can be costly and time consuming for web-scale extractions. Bootstrapping machine-learned rules can make extractions easier on large corpora. Thus, open information extraction systems (Carlson et al., 2010; Fader et al., 2011; Mausam et al., 2012; Mesquita et al., 2013; Angeli et al., 2015) have recently been popular for domain specific or independent pattern learning.

Hearst (1992) used hand written rules to generate more rules to extract hypernym-hyponym pairs, without distributional similarity. For entity extraction, Riloff (1996) used seed entities to generate extractors with heuristic rules and scored them by counting positive extractions. Prior work (Lin et al., 2003; Gupta et al., 2014) investigated different extractor scoring measures. Gupta and Manning (2014) improved scores by introducing expected number of negative entities.

Brin (1998) developed the bootstrapping relation extraction system DIPRE that generates extractors by clustering contexts based on string matching. SnowBall (Agichtein and Gravano, 2000) is inspired by DIPRE but computes a TF-IDF representation of each context. BREDS (Batista et al., 2015) uses word embeddings (Mikolov et al., 2013) to bootstrap relationships.

Related work investigated adapting extractor scoring measures in bootstrapping entity extraction with either entities or templates (Table 1) as seeds (Table 2). The state-of-the-art relation extractors bootstrap with only seed entity pairs and suffer due to a surplus of unknown extractions and the lack of labeled data, leading to low confidence extractors. This in turn leads to low confidence in the system output. Prior RE sys-
BREE Bootstrapping Relation Extractor with Entity pair
BRET Bootstrapping Relation Extractor with Template
BREJ Bootstrapping Relation Extractor in Joint learning
type a named entity type, e.g., person
typed entity a typed entity, e.g., <“Obama”,person>
entity pair a pair of two typed entities
template a triple of vectors ($\vec{v}_{-1}$, $\vec{v}_0$, $\vec{v}_1$) and an entity pair
instance entity pair and template (types must be the same)
$\gamma$ instance set extracted from corpus
$i$ a member of $\gamma$, i.e., an instance
$x(i)$ the entity pair of instance $i$
y(i) the template of instance $i$
$G_p$ a set of positive seed entity pairs
$G_n$ a set of negative seed entity pairs
$\Sigma_p$ a set of positive seed templates
$\Sigma_n$ a set of negative seed templates
$\mathcal{G}$ $<G_p, G_n, \Sigma_p, \Sigma_n>$
k_{it} number of iterations
$\lambda$ cluster of instances (extractor)
$c_{it}$ category of extractor $\lambda$
$\lambda_{NNHC}$ Non-Noisy-High-Confidence extractor (True Positive)
$\lambda_{NNLC}$ Non-Noisy-Low-Confidence extractor (True Negative)
$\lambda_{NHIC}$ Noisy-High-Confidence extractor (False Positive)
$\lambda_{NLC}$ Noisy-Low-Confidence extractor (False Negative)

Table 1: Notation and definition of key terms

We introduce cross-context similarities that compare all parts of the context (e.g., “to acquire” and “acquisition”) and show that these perform better (in terms of recall) than measures assuming consistent syntactic expression of relations.

(3) Experimental results demonstrate a 13% gain in $F_1$ score on average for four relationships and suggest eliminating four parameters, compared to the state-of-the-art method.

The motivation and benefits of the proposed JBM for relation extraction is discussed in depth in section 2.3. The method is applicable for both entity and relation extraction tasks. However, in context of relation extraction, we call it BREJ.

2 Method

2.1 Notation and definitions

We first introduce the notation and terms (Table 1).

Given a relationship like “$x$ acquires $y$”, the task is to extract pairs of entities from a corpus for which the relationship is true. We assume that the arguments of the relationship are typed, e.g., $x$ and $y$ are organizations. We run a named entity tagger in preprocessing, so that the types of all candidate entities are given. The objects the bootstrapping algorithm generally handles are therefore typed entities (an entity associated with a type).

For a particular sentence in a corpus that states that the relationship (e.g., “acquires”) holds between $x$ and $y$, a template consists of three vectors that represent the context of $x$ and $y$. $\vec{v}_{-1}$ represents the context before $x$, $\vec{v}_0$ the context between $x$ and $y$ and $\vec{v}_1$ the context after $y$. These vectors are simply sums of the embeddings of the corresponding words. A template is “typed”, i.e., in addition to the three vectors it specifies the types of the two entities. An instance joins an entity pair and a template. The types of entity pair and template must be the same.

The first step of bootstrapping is to extract a set of instances from the input corpus. We refer to this set as $\gamma$. We will use $i$ and $j$ to refer to instances. $x(i)$ is the entity pair of instance $i$ and $y(i)$ is the template of instance $i$.

A required input to our algorithm are sets of positive and negative seeds for either entity pairs ($G_p$ and $G_n$) or templates ($\Sigma_p$ and $\Sigma_n$) or both. We define $\mathcal{G}$ to be a tuple of all four seed sets.

We run our bootstrapping algorithm for $k_{it}$ iterations where $k_{it}$ is a parameter.
A key notion is the similarity between two instances. We will experiment with different similarity measures. The baseline is (Batista et al., 2015)'s measure given in Figure 4, first line: the similarity of two instances is given as a weighted sum of the dot products of their before contexts \((\vec{v}_1)\), their between contexts \((\vec{v}_0)\) and their after contexts \((\vec{v}_1)\) where the weights \(w_{ij}\) are parameters. We give this definition for instances, but it also applies to templates since only the context vectors of an instance are used, not the entities.

The similarity between an instance \(i\) and a cluster \(\lambda\) of instances is defined as the maximum similarity of \(i\) with any member of the cluster; see Figure 2, right, Eq. 5. Again, there is a straightforward extension to a cluster of templates: see Figure 2, right, Eq. 6.

The extractors \(\Lambda\) can be categorized as follows:

\[
\Lambda_{\text{NHC}} = \{\lambda \in \Lambda | \lambda \rightarrow \mathcal{G}^{\text{non-noisy}} \land \text{cnf}(\lambda, \mathcal{G}) \geq \tau_{\text{cnf}}\} \\
\Lambda_{\text{NLC}} = \{\lambda \in \Lambda | \lambda \rightarrow \mathcal{G} \land \text{cnf}(\lambda, \mathcal{G}) < \tau_{\text{cnf}}\} \\
\Lambda_{\text{NHC}} = \{\lambda \in \Lambda | \lambda \rightarrow \mathcal{G} \land \text{cnf}(\lambda, \mathcal{G}) \geq \tau_{\text{cnf}}\} \\
\Lambda_{\text{NLC}} = \{\lambda \in \Lambda | \lambda \rightarrow \mathcal{G} \land \text{cnf}(\lambda, \mathcal{G}) < \tau_{\text{cnf}}\}
\]

where \(\mathcal{G}\) is the relation to be bootstrapped. The \(\lambda_{\text{cat}}\) is a member of \(\Lambda_{\text{cat}}\). For instance, a \(\lambda_{\text{NLC}}\) is called as a \textit{non-noisy-low-confidence} extractor if it represents the target relation (i.e., \(\lambda \rightarrow \mathcal{G}\)), however with the confidence below a certain threshold \((\tau_{\text{cnf}})\). Extractors of types \(\Lambda_{\text{NHC}}\) and \(\Lambda_{\text{NLC}}\) are desirable, those of types \(\Lambda_{\text{NHC}}\) and \(\Lambda_{\text{NLC}}\) undesirable within bootstrapping.

### 2.2 The Bootstrapping Machines: BREX

To describe BREX (Figure 1) in its most general form, we use the term \textit{item} to refer to an entity pair, a template or both.

The input to BREX (Figure 2, left, line 01) is a set \(\gamma\) of instances extracted from a corpus and \(\mathcal{G}_{\text{seed}}\), a structure consisting of one set of positive and one set of negative seed items. \(\mathcal{G}_{\text{yield}}\) (line 02) collects the items that BREX extracts in several iterations. In each of \(k_\alpha\) iterations (line 03), BREX first initializes the cache \(\mathcal{G}_{\text{cache}}\) (line 04); this cache collects the items that are extracted in this iteration. The design of the algorithm balances elements that ensure high recall with elements that ensure high precision.

High recall is achieved by starting with the seeds and making three “hops” that consecutively consider order-1, order-2 and order-3 neighbors of the seeds. On line 05, we make the first hop: all instances that are similar to a seed are collected where “similarity” is defined differently for different BREX configurations (see below). The collected instances are then clustered, similar to work on bootstrapping by Agichtein and Gravano (2000) and Batista et al. (2015). On line 06, we make the second hop: all instances that are within \(\tau_{\text{sim}}\) of a hop-1 instance are added; each such instance is only added to one cluster, the closest one; see definition of \(\mu\): Figure 2, Eq. 8. On line 07, we make the third hop: we include all instances that are within \(\tau_{\text{sim}}\) of a hop-2 instance; see definition of \(\psi\): Figure 2, Eq. 7. In summary, every instance that can be reached by three hops from a seed is being considered at this point. A cluster of hop-2 instances is named as \textit{extractor}.

High precision is achieved by imposing, on line 08, a stringent check on each instance before its information is added to the cache. The core function of this check is given in Figure 2, Eq. 9. This definition is a soft version of the following hard max, which is easier to explain:

\[
\text{cnf}(i, \Lambda, \mathcal{G}) \approx \max_{\lambda \in \Lambda \land \mu(\lambda)} \text{cnf}(i, \lambda, \mathcal{G})
\]

We are looking for a cluster \(\lambda\) in \(\Lambda\) that licenses the extraction of \(i\) with high confidence. \(\text{cnf}(i, \lambda, \mathcal{G})\) (Figure 2, Eq. 10), the confidence of a single cluster (i.e., extractor) \(\lambda\) for an instance, is defined as the product of the overall reliability of \(\lambda\) (which is independent of \(i\)) and the similarity of \(i\) to \(\lambda\), the second factor in Eq. 10, i.e., \(\sim(i, \lambda)\). This factor \(\sim(i, \lambda)\) prevents an extraction by a cluster whose members are all distant from the instance – even if the cluster itself is highly reliable.
Algorithm: BREX

01 INPUT: \( \gamma, \mathcal{G}_{\text{seed}} \)
02 \( \mathcal{G}_{\text{yield}} := \mathcal{G}_{\text{seed}} \)
03 for \( k_i \) iterations:
04 \( \mathcal{G}_{\text{cache}} := \emptyset \)
05 \( \Theta := \bigcup \{ (i \in \gamma | \text{match}(i, \mathcal{G}_{\text{yield}})) \} \)
06 \( \Lambda := \{ \mu(\theta, \mathcal{G}) | \theta \in \Theta \} \)
07 for each \( i \in \bigcup_{\lambda \in \Lambda} \psi(\lambda) \):
08 if check \( (i, \Lambda, \mathcal{G}_{\text{yield}}) \):
09 \( \text{add}(i, \mathcal{G}_{\text{cache}}) \)
10 \( \mathcal{G}_{\text{yield}} \cup= \mathcal{G}_{\text{cache}} \)
11 OUTPUT: \( \mathcal{G}_{\text{yield}}, \Lambda \)

\[
\text{sim}(i, \lambda) = \max_{i \in \lambda} \forall i \in \lambda \text{sim}(i, i') \quad (5)
\]
\[
\psi(\lambda) = \{ i \in \gamma | \text{sim}(i, \lambda) \geq \tau_{\text{sim}} \} \quad (6)
\]
\[
\mu(\theta, \Theta) = \{ i \in \gamma | \text{sim}(i, \theta) = d \land d = \max_{\theta \in \Theta} \text{sim}(i, \theta) \geq \tau_{\text{sim}} \} \quad (7)
\]

Figure 2: BREX algorithm (left) and definition of key concepts (right)

| Seed Type | BREE | BRET | BREJ |
|-----------|------|------|------|
| (i) \( N_\gamma(\lambda, \mathcal{G}) \) | \( \{ i \in \lambda | x(i) \in \mathcal{G}_i \} \) | \( \{ i \in \lambda | \text{sim}(i, \mathcal{G}_i) \geq \tau_{\text{sim}} \} \) | \( \{ i \in \lambda | x(i) \in \mathcal{G}_i \} + \{ i \in \lambda | \text{sim}(i, \mathcal{G}_i) \geq \tau_{\text{sim}} \} \) |
| (ii) \( \{ w_n, w_u \} \) | \( (1.0, 0.0) \) | \( (1.0, 0.0) \) | \( (1.0, 0.0) \) |

Figure 3: BREX configurations

The first factor in Eq. 10, i.e., \( \text{cnf}(\lambda, \mathcal{G}) \), assesses the reliability of a cluster \( \lambda \): we compute the ratio \( \frac{\sum x(i) \in \mathcal{G}_i}{\sum x(i) \in \mathcal{G}_i} \), i.e., the ratio between the number of instances in \( \lambda \) that match a negative and positive gold seed, respectively; see Figure 3, line (i). If this ratio is close to zero, then likely false positive extractions are few compared to likely true positive extractions. For the simple version of the algorithm (for which we set \( w_n = 1, w_u = 0 \)), this results in \( \text{cnf}(\lambda, \mathcal{G}) \) being close to 1 and the reliability measure not discounted. On the other hand, if \( \frac{\sum x(i) \in \mathcal{G}_i}{\sum x(i) \in \mathcal{G}_i} \) is larger, meaning that the relative number of likely false positive extractions is high, then \( \text{cnf}(\lambda, \mathcal{G}) \) shrinks towards 0, resulting in progressive discounting of \( \text{cnf}(\lambda, \mathcal{G}) \) and leading to non-noisy-low-confidence extractor, particularly for a reliable \( \lambda \). Due to lack of labeled data, the scoring mechanism cannot distinguish between noisy and non-noisy extractors. Therefore, an extractor is judged by its ability to extract more positive and less negative extractions. Note that we carefully designed this precision component to give good assessments while at the same time making maximum use of the available seeds. The reliability statistics are computed on \( \lambda \), i.e., on hop-2 instances (not on hop-3 instances). The ratio \( \frac{\sum x(i) \in \mathcal{G}_i}{\sum x(i) \in \mathcal{G}_i} \) is computed on instances that directly match a gold seed – this is the most reliable information we have available.

After all instances have been checked (line 08) and (if they passed muster) added to the cache (line 09), the inner loop ends and the cache is merged into the yield (line 10). Then a new loop (lines 03–10) of hop-1, hop-2 and hop-3 extensions and cluster reliability tests starts.

Thus, the algorithm consists of \( k_{it} \) iterations. There is a tradeoff here between \( \tau_{\text{sim}} \) and \( k_{it} \). We will give two extreme examples, assuming that we want to extract a fixed number of \( m \) instances where \( m \) is given. We can achieve this goal either by setting \( k_{it} = 1 \) and choosing a small \( \tau_{\text{sim}} \), which will result in very large hops. Or we can achieve this goal by setting \( \tau_{\text{sim}} \) to a large value and running the algorithm for a larger number of \( k_{it} \). The flexibility that the two hyperparameters \( k_{it} \) and \( \tau_{\text{sim}} \) afford is important for good performance.
\begin{equation}
\text{sim}_{\text{match}}(i, j) = \sum_{p \in (-1, 0, 1)} w_p \varphi_p(i) \varphi_p(j) \quad ; \quad \text{sim}_{\text{sym}}^\text{I}(i, j) = \max_{p \in (-1, 0, 1)} \varphi_p(i) \varphi_0(j) \quad (13)
\end{equation}

\begin{equation}
\text{sim}_{\text{sym}}^\text{I}(i, j) = \max \left( \max_{p \in (-1, 0, 1)} \varphi_p(i) \varphi_0(j), \max_{p \in (-1, 0, 1)} \varphi_0(i) \varphi_p(j) \right) \quad (14)
\end{equation}

\begin{equation}
\text{sim}_{\text{sym}}^\text{2}(i, j) = \max \left( \left( \bar{v}_{-1}(i) + \bar{v}_1(i) \right) \bar{v}_0(j), \left( \bar{v}_{-1}(j) + \bar{v}_1(j) \right) \bar{v}_0(i), \bar{v}_0(i) \bar{v}_0(j) \right) \quad (15)
\end{equation}

Figure 4: Similarity measures. These definitions for instances equally apply to templates since the definitions only depend on the “template part” of an instance, i.e., its vectors. (value is 0 if types are different)

2.3 BREE, BRET and BREJ

The main contribution of this paper is that we propose, as an alternative to entity-pair-centered BREE (Batista et al., 2015), template-centered BREJ as well as BREJ (Figure 1), an instantiation of BREX that can take advantage of both entity pairs and templates. The differences and advantages of BREJ over BREE and BRET are:

(1) Disjunctive Matching of Instances: The first difference is realized in how the three algorithms match instances with seeds (line 05 in Figure 3). BREE checks whether the entity pair of an instance is one of the entity pair seeds. BRET checks whether the template of an instance is one of the template seeds and BREJ checks whether the disjunction of the two is true. The disjunction facilitates a higher hit rate in matching instances with seeds. The introduction of a few hand-crafted templates along with seed entity pairs allows BREJ to leverage discriminative patterns and learn similar ones via distributional semantics. In Figure 1, the joint approach results in hybrid extractors \( \Lambda \) that contain instances due to seed occurrences \( \Theta \) of both entity pairs and templates.

(2) Hybrid Augmentation of Seeds: On line 09 in Figure 3, we see that the bootstrapping step is defined in a straightforward fashion: the entity pair of an instance is added for BREE, the template for BRET and both for BREJ. Figure 1 demonstrates the hybrid augmentation of seeds via red and blue rings of output instances.

(3) Scaling Up Positives in Extractors: As discussed in section 2.2, a good measure of the quality of an extractor is crucial and \( N_+ \), the number of instances in an extractor \( \lambda \) that match a seed, is an important component of that. For BREE and BRET, the definition follows directly from the fact that these are entity-pair and template-centered instantiations of BREX, respectively. However, the disjunctive matching of instances for an extractor with entity pair and template seeds in BREJ (Figure 3 line “(i)” ) boosts the likelihood of finding positive instances. In Figure 5, we demonstrate computing the count of positive instances
\[ N_+(\lambda, \mathcal{G}) \text{ for an extractor } \lambda \text{ within the three systems. Observe that an instance } i \text{ in } \lambda \text{ can scale its } N_+(\lambda, \mathcal{G}) \text{ by a factor of maximum } 2 \text{ in BREJ if } i \text{ is matched in both entity pair and template seeds. The reliability } cnf(\lambda, \mathcal{G}) \text{ (Eq. 11) of an extractor } \lambda \text{ is based on the ratio } \frac{N_+(\lambda_G)}{N_+(\lambda_D)}, \text{ therefore suggesting that the scaling boosts its confidence.}

In Figure 6, we demonstrate with an example how the joint bootstrapping scales up the positive instances for a non-noisy extractor \( \lambda \) resulting in \( \lambda_{NNHC} \) for BREJ compared to \( \lambda_{NNLC} \) in BREE.

Due to unlabeled data, the instances not matching in seeds are considered either to be ignored/unknown \( N_0 \) or negatives in the confidence measure (Eq. 11). The former leads to high confidences for noisy extractors by assigning high scores, the latter to low confidences for non-noisy extractors by penalizing them. For a simple version of the algorithm in the illustration, we consider them as negatives and set \( w_n = 1 \). Figure 6 shows the three extractors (\( \lambda \)) generated and their confidence scores in BREJ, BRET and BREJ. Observe that the scaling up of positives in BREJ due to BREJ extractions (without \( w_n \)) discounts \( cnf(\lambda, \mathcal{G}) \) relatively lower than BREE. The discounting results in \( \lambda_{NNHC} \) in BREJ and \( \lambda_{NNLC} \) in BREE. The discounting in BREJ is adapted for non-noisy extractors facilitated by BRET in generating mostly non-noisy extractors due to stringent checks (Figure 3, line “(i)” and 05). Intuitively, the intermixing of non-noisy extractors (i.e., hybrid) promotes the scaling and boosts recall.

### 2.4 Similarity Measures

The before \( (\vec{v}_{i-1}) \) and after \( (\vec{v}_{i}) \) contexts around the entities are highly sparse due to large variation in the syntax of how relations are expressed. SnowBall, DIPRE and BREE assumed that the between \( (\vec{v}_0) \) context mostly defines the syntactic expression for a relation and used weighted mechanism on the three contextual similarities in

| Relationship | Seed Entity Pairs | Seed Templates |
|-------------|------------------|----------------|
| acquired    | [Adidas;Reebok], [Google;DoubleClick], [Widnes;Warrington], [Hewlett-Packard;Compaq] | \{[X] acquire [Y], [X] acquisition [Y], [X] buy [Y], [X] takeover [Y], [X] merger with [Y]\} |
| founder-of  | [CNN;Ted Turner], [Facebook;Mark Zuckerberg], [Microsoft;Paul Allen], [Amazon;Jeff Bezos] | \{[X] founded by [Y], [X] co-founder [Y], [X] started by [Y], [X] founder of [Y], [X] owner of [Y]\} |
| headquartered | [Nokia;Espos], [Pfizer;New York], [United Nations;New York], [NATO;Brussels] | \{[X] based in [Y], [X] headquarters in [Y], [X] head office in [Y], [X] main office building in [Y], [X] campus branch in [Y]\} |
| affiliation | [Google;Marissa Mayer], [Xerox;Ursula Burns], [Microsoft;Steve Ballmer], [Microsoft;Bill Gates] | \{[X] CEO [Y], [X] resign from [Y], [X] founded by [Y], [X] worked for [Y], [X] chairman director [Y]\} |

Table 2: Seed Entity Pairs and Templates for each relation. [X] and [Y] are slots for entity type tags.

| Parameter | Description/Search | Optimal |
|-----------|--------------------|---------|
| | maximum number of tokens in between context | 2 |
| | maximum number of tokens in between context | 6 |
| | maximum number of tokens in after context | 2 |
| \( \tau_{sim} \) | similarity threshold \([0.6, 0.7, 0.8]\) | 0.7 |
| \( \tau_{conf} \) | instance confidence thresholds \([0.6, 0.7, 0.8]\) | 0.7 |
| \( w_{neg} \) | weights to negative extractions \([0.0, 0.5, 1.0, 2.0]\) | 0.5 |
| \( w_{pos} \) | weights to unknown extractions \([0.0001, 0.00001]\) | 0.0001 |
| \( k_2 \) | number of bootstrapping epochs | 3 |
| \( dim_{emb} \) | dimension of embedding vector, \( V \) | 300 |
| \( PMI \) | PMI threshold in evaluation | 0.5 |
| Entity Pairs | Ordered Pairs (OP) or Bisetts (BS) | OP |

Table 3: Count of entity-type pairs in corpus

| Parameter | Description/Search | Optimal |
|-----------|--------------------|---------|
| | maximum number of tokens in between context | 2 |
| | maximum number of tokens in between context | 6 |
| | maximum number of tokens in after context | 2 |
| \( \tau_{sim} \) | similarity threshold \([0.6, 0.7, 0.8]\) | 0.7 |
| \( \tau_{conf} \) | instance confidence thresholds \([0.6, 0.7, 0.8]\) | 0.7 |
| \( w_{neg} \) | weights to negative extractions \([0.0, 0.5, 1.0, 2.0]\) | 0.5 |
| \( w_{pos} \) | weights to unknown extractions \([0.0001, 0.00001]\) | 0.0001 |
| \( k_2 \) | number of bootstrapping epochs | 3 |
| \( dim_{emb} \) | dimension of embedding vector, \( V \) | 300 |
| \( PMI \) | PMI threshold in evaluation | 0.5 |
| Entity Pairs | Ordered Pairs (OP) or Bisetts (BS) | OP |

Table 4: Hyperparameters in BREE, BRET and BREJ

Pairs, \( sim_{match} \) (Figure 4). They assigned higher weights to the similarity in between \((p = 0)\) contexts, that resulted in lower recall. We introduce attentive \( (max) \) similarity across all contexts (for example, \( v_{i-1}(i)v_{0}(j) \)) to automatically capture the large variation in the syntax of how relations are expressed, without using any weights. We investigate asymmetric \( (\text{Eq } 13) \) and symmetric \( (\text{Eq } 14 \text{ and } 15) \) similarity measures, and name them as cross-context attentive \( (\text{sim}_{ctx}) \) similarity.

### 3 Evaluation

#### 3.1 Dataset and Experimental Setup

We re-run BREE (Batista et al., 2015) for baseline with a set of 5.5 million news articles from AFP and APW (Parker et al., 2011). We use processed dataset of 1.2 million sentences (released by BREE) containing at least two entities linked to FreebaseEasy (Bast et al., 2014). We extract four relationships: acquired (ORG-ORG), founder-of (ORG-PER), headquartered (ORG-LOC) and affiliation (ORG-PER) for Organization (ORG), Person (PER) and Location (LOC) entity types. We bootstrap relations in BREE, BRET and BREJ, each with 4 similarity measures using seed entity
Table 5: Precision ($P$), Recall ($R$) and $F_1$ compared to the state-of-the-art (baseline). #out: count of output instances with $cnf(i, \Lambda, G) \geq 0.5$. avg: average. Bold and underline: Maximum due to BREJ and sim$_{cc}$, respectively.

| Relationships | #out | $P$ | $R$ | $F_1$ | #out | $P$ | $R$ | $F_1$ | #out | $P$ | $R$ | $F_1$ |
|--------------|-------|-----|-----|-------|-------|-----|-----|-------|-------|-----|-----|-------|
| BREJ         | baseline: BREE+sim$_{match}$ | config$_{9}$: BREE+sim$_{sim}^{cc}$ | config$_{10}$: BREE+sim$_{sim}^{sym}^{cc}$ | config$_{11}$: BREE+sim$_{sym}^{sim}^{cc}$ | config$_{12}$: BREE+sim$_{match}$ |
| acquired     | 2687  | 0.88 | 0.48 | 0.62  | 5771  | 0.88 | 0.66 | 0.76  | 3471  | 0.88 | 0.55 | 0.68  |
| founder-of   | 628   | 0.98 | 0.70 | 0.82  | 9553  | 0.86 | 0.95 | 0.89  | 1532  | 0.94 | 0.84 | 0.89  |
| headquartered| 16786 | 0.62 | 0.80 | 0.69  | 21299 | 0.66 | 0.85 | 0.74  | 17301 | 0.70 | 0.83 | 0.76  |
| affiliation  | 20948 | 0.99 | 0.73 | 0.84  | 27244 | 0.97 | 0.78 | 0.87  | 36795 | 0.95 | 0.82 | 0.88  |
| avg          | 10262 | 0.86 | 0.68 | 0.74  | 16011 | 0.84 | 0.81 | 0.82  | 14475 | 0.87 | 0.76 | 0.88  |

Table 6: Iterations ($k_i$) Vs Scores with thresholds ($\tau$) for relation acquired in BREJ. $\tau$ refers to $\tau_{sim}$ and $\tau_{cnf}$.

| $\tau$ | $k_i$ | #out | $P$ | $R$ | $F_1$ |
|-------|-------|------|-----|-----|-------|
| 0.6   | 1     | 691  | 0.99 | 0.21 | 0.35  |
|       | 2     | 11288| 0.85 | 0.79 | 0.81  |
| 0.7   | 1     | 610  | 0.9 | 0.19 | 0.32  |
|       | 2     | 7948 | 0.93 | 0.75 | 0.83  |
| 0.8   | 1     | 522  | 1.0 | 0.17 | 0.29  |
|       | 2     | 2969 | 0.90 | 0.51 | 0.65  |

Table 7: Comparative analysis using different thresholds $\tau$ to evaluate the extracted instances for acquired relations outperforms the baseline BREE and BRET in terms of $F_1$ score. On an average for the four relations, BREJ in configurations config$_9$ and config$_{10}$ results in $F_1$ that is 0.11 (0.85 vs 0.74) and 0.13 (0.87 vs 0.74) better than the baseline BREE.

We discover that sim$_{cc}$ improves #out and recall over sim$_{match}$ correspondingly in all three systems. Observe that sim$_{cc}$ performs better with BREJ than BREE due to non-noisy extractors in BREJ. The results suggest an alternative to the weighting scheme in sim$_{match}$ and therefore, the state-of-the-art (sim$_{cc}$) performance with the 3 parameters ($w_{-1}$, $w_0$ and $w_1$) ignored in bootstrap-pairs and templates (Table 2). See Tables 3, 4 and 5 for the count of candidates, hyperparameters and different configurations, respectively.

Our evaluation is based on Bronzi et al. (2012)’s framework to estimate precision and recall of large-scale RE systems using FreebaseEasy (Bast et al., 2014). Also following Bronzi et al. (2012), we use Pointwise Mutual Information (PMI) (Turney, 2001) to evaluate our system automatically, in addition to relying on an external knowledge base. We consider only extracted relationship instances with confidence scores $cnf(i, \Lambda, G)$ equal or above 0.5. We follow the same approach as BREE (Batista et al., 2015) to detect the correct order of entities in a relational triple, where we try to identify the presence of passive voice using part-of-speech (POS) tags and considering any form of the verb to be, followed by a verb in the past tense or past participle, and ending in the word ‘by’. We use GloVe (Pennington et al., 2014) embeddings.

3.2 Results and Comparison with baseline

Table 5 shows the experimental results in the three systems for the different relationships with ordered entity pairs and similarity measures (sim$_{match}$, sim$_{cc}$). Observe that BRET (config$_9$) is precision-oriented while BREJ (config$_{10}$) recall-oriented when compared to BREE (baseline). We see the number of output instances #out are also higher in BREJ, therefore the higher recall. The BREJ system in the different similarity configurations outperforms the baseline BREE and BRET in terms of $F_1$ score. On an average for the four relations, BREJ in configurations config$_9$ and config$_{10}$ results in $F_1$ that is 0.11 (0.85 vs 0.74) and 0.13 (0.87 vs 0.74) better than the baseline BREE.
Table 8: Disjunctive matching of Instances. \#hit: the count of instances matched to positive seeds in \(k_{it} = 1\).

| Attributes | \(\lambda\) | AIE | AES | ANE | ANNLC | AP | AN | \#BRET | \#BREE | \#BREJ |
|------------|-----------|-----|-----|-----|-------|----|----|--------|--------|--------|
| BRET       | 187       | 0.12| 0.51| 0.04| 0.16  | 0.14| 37.7| 93.1   | 2.46   |         |
| BRET       | 17        | 305.2| 1.00| 0.11| 0.09  | 0.00| 671.8| 0.12   | 0.00   |         |
| BRET       | 555       | 0.36| 0.74| 0.07| 0.29  | 0.03| 38.32| 44.8   | 0.14   |         |
| BRET       | 5         | 13.3 | 0.46| 0.75| 0.25  | 0.12| 44.9 | 60.05  | 13.57  |         |
| BRET       | 5         | 179.0| 1.00| 0.00| 1.00  | 0.00| 37.22| 0.00   | 0.00   |         |
| BRET       | 492       | 109.1| 0.90| 0.04| 0.06  | 0.00| 40.18| 79.5   | 0.18   |         |
| BRET       | 895       | 18.4 | 0.60| 0.07| 0.05  | 0.02| 46.3 | 82.7   | 1.78   |         |
| BRET       | 7         | 365.7| 1.00| 0.00| 1.00  | 0.00| 848.6| 0.05   | 0.00   |         |
| BRET       | 1311      | 45.5 | 0.80| 0.98| 0.02  | 0.00| 324.1| 77.5   | 0.24   |         |
| BREJ       | 198       | 99.7 | 0.55| 0.25| 0.75  | 0.34| 240.5| 152.2  | 0.65   |         |
| BREJ       | 19        | 846.9| 1.00| 0.00| 1.00  | 0.00| 217.6| 0.00   | 0.00   |         |
| BREJ       | 430       | 130.2| 0.72| 0.21| 0.79  | 0.06| 567.6| 122.7  | 0.22   |         |

Table 9: Analyzing the attributes of extractors \(\Lambda\) learned for each relationship. Attributes are: number of extractors \(|\Lambda|\), \(\text{avg}\) number of instances in \(\Lambda\) (AIE), \(\text{avg}\) \(\lambda\) score (AES), \(\text{avg}\) number of noisy \(\lambda\) (ANE), \(\text{avg}\) number of non-noisy \(\lambda\) (ANNLC), \(\text{avg}\) \(\Lambda\) confidence below confidence 0.5 (ANNLC), \(\text{avg}\) number of positives (AP) and negatives (AN), ratio of AN to AP (ANP). The bold indicates comparison of BREE and BREJ with \(\text{sim}_{\text{match}}\). \(\text{avg}: \text{average}\).

Table 10: \(\text{BREX}+\text{sim}_{\text{match}}\). Scores when \(w_n\) ignored.

Table 11: Some of the non-noisy extractors (simplified) learned in different configurations to illustrate boosting extractor confidence \(\text{cnf}(\lambda, G)\). Since, an extractor \(\lambda\) is a cluster of instances, therefore to simplify, we show one instance, therefore to simplify, we show one...
Table 11: Subset of the non-noisy extractors (simplified) with their confidence scores \( cnf \). Each cell \( \lambda \) and \( \mathcal{G} \) denotes that the extractor was never learned in config\(_1\) and config\(_2\), ♦ indicates that the extractor was never learned in config\(_1\), config\(_2\) and config\(_3\), [X] and [Y] indicate placeholders for entities.

Table 12: BREX+sim\(_{match}\) Scores with entity bises of both entity-pair-centered and template-centered approaches. We have demonstrated that the joint approach scales up positive instances that boosts the confidence of NNLC extractors and improves recall. The experiments showed that the cross-context similarity measures improved recall and suggest removing in total four parameters.

Acknowledgments

We thank our colleagues Bernt Andrassy, Mark Buckley, Stefan Langer, Ulli Waltinger and Usama Yaseen, and anonymous reviewers for their review comments. This research was supported by Bundeswirtschaftsministerium (bmvw.de), grant 01MD15010A (Smart Data Web) at Siemens AG-CT Machine Intelligence, Munich Germany.
References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In Proceedings of the 15th ACM conference on Digital libraries. Association for Computing Machinery, Washington, DC USA, pages 85–94.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. Association for Computational Linguistics, Beijing, China, volume 1, pages 344–354.

Hannah Bast, Florian Bäurle, Björn Buchhold, and Elmar Haußmann. 2014. Easy access to the freebase dataset. In Proceedings of the 23rd International Conference on World Wide Web. Association for Computing Machinery, Seoul, Republic of Korea, pages 95–98.

David S. Batista, Bruno Martins, and Mário J. Silva. 2015. Semi-supervised bootstrapping of relationship extractors with distributional semantics. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Lisbon, Portugal, pages 499–504.

Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In International Workshop on The World Wide Web and Databases. Springer, Valencia, Spain, pages 172–183.

Mirko Bronzi, Zhaochen Guo, Filipe Mesquita, Denilson Barbosa, and Paolo Merlaldo. 2012. Automatic evaluation of relation extraction systems on large-scale. In Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX). Association for Computational Linguistics, Montréal, Canada, pages 19–24.

Andrew Carlson, Justin Betteridge, Bryan Kiesiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In Proceedings of the 24th National Conference on Artificial Intelligence (AAAI). Atlanta, Georgia USA, volume 5, page 3.

Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. 2013. Rule-based information extraction is dead! long live rule-based information extraction systems! In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Seattle, Washington USA, pages 827–832.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Edinburgh, Scotland UK, pages 1535–1545.

Pankaj Gupta, Thomas Runkler, Heike Adel, Bernt Andrassy, Hans-Georg Zimmermann, and Hinrich Schütze. 2015. Deep learning methods for the extraction of relations in natural language text. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers. Osaka, Japan, pages 2537–2547.

Sonal Gupta, Diana L. MacLean, Jeffrey Heer, and Christopher D. Manning. 2014. Induced lexicosyntactic patterns improve information extraction from online medical forums. Journal of the American Medical Informatics Association 21(5):902–909.

Sonal Gupta and Christopher Manning. 2014. Improved pattern learning for bootstrapped entity extraction. In Proceedings of the 18th Conference on Computational Natural Language Learning (CoNLL). Association for Computational Linguistics, Baltimore, Maryland USA, pages 98–108.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In Proceedings of the 15th International Conference on Computational Linguistics. Nantes, France, pages 539–545.

Winston Lin, Roman Yangarber, and Ralph Grishman. 2003. Bootstrapped learning of semantic classes from positive and negative examples. In Proceedings of ICML 2003 Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining. Washington, DC USA, page 21.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. 2012. Open language learning for information extraction. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, Jeju Island, Korea, pages 523–534.

Filipe Mesquita, Jordan Schmidek, and Denilson Barbosa. 2013. Effectiveness and efficiency of open relation extraction. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Seattle, Washington USA, pages 447–457.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In Proceedings of the Workshop at the International Conference on Learning Representations. ICLR, Scottsdale, Arizona USA.
Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing. Association for Computational Linguistics, Denver, Colorado USA, pages 39–48.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword. Linguistic Data Consortium.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Doha, Qatar, pages 1532–1543.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In Proceedings of the 13th National Conference on Artificial Intelligence (AAAI). Portland, Oregon USA, pages 1044–1049.

Peter D. Turney. 2001. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In Proceedings of the 12th European Conference on Machine Learning. Springer, Freiburg, Germany, pages 491–502.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016a. Combining recurrent and convolutional neural networks for relation classification. In Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT). Association for Computational Linguistics, San Diego, California USA, pages 534–539.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, Shanghai, China, pages 6060–6064.