Do Supervised Distributional Methods Really Learn Lexical Inference Relations?

Omer Levy†  Steffen Remus§  Chris Biemann§  Ido Dagan†

† Natural Language Processing Lab  Bar-Ilan University  Ramat-Gan, Israel  {omerlevy,dagan}@cs.biu.ac.il
§ Language Technology Lab  Technische Universität Darmstadt  Darmstadt, Germany  {remus,biem}@cs.tu-darmstadt.de

Abstract

Distributional representations of words have been recently used in supervised settings for recognizing lexical inference relations between word pairs, such as hypernymy and entailment. We investigate a collection of these state-of-the-art methods, and show that they do not actually learn a relation between two words. Instead, they learn an independent property of a single word in the pair: whether that word is a “prototypical hypernym”. We modify them accordingly by incorporating the similarity between $x$ and $y$. Unfortunately, the improvement in performance is incremental. We suspect that methods based solely on contextual features of single words are not learning lexical inference relations because contextual features might lack the necessary information to deduce how one word relates to another.

1 Introduction

Inference in language involves recognizing inference relations between two words ($x$ and $y$), such as causality ($flu \rightarrow fever$), hypernymy ($cat \rightarrow animal$), and other notions of lexical entailment. The distributional approach to automatically recognize these relations relies on representing each word $x$ as a vector $\vec{x}$ of contextual features: other words that tend to appear in its vicinity. Such features are typically used in word similarity tasks, where cosine similarity is a standard similarity measure between two word vectors: $\text{sim}(x, y) = \cos(\vec{x}, \vec{y})$.

Many unsupervised distributional methods of recognizing lexical inference replace cosine similarity with an asymmetric similarity function (Weeds and Weir, 2003; Clarke, 2009; Kotlerman et al., 2010; Santus et al., 2014). Supervised methods, reported to perform better, try to learn the asymmetric operator from a training set. The various supervised methods differ by the way they represent each candidate pair of words ($x, y$): Baroni et al. (2012) use concatenation $\vec{x} \oplus \vec{y}$, others (Roller et al., 2014; Weeds et al., 2014; Fu et al., 2014) take the vectors’ difference $\vec{y} - \vec{x}$, and more sophisticated representations, based on contextual features, have also been tested (Turney and Mohammad, 2014; Rimell, 2014).

In this paper, we argue that these supervised methods do not, in fact, learn to recognize lexical inference. Our experiments reveal that much of their previously perceived success stems from lexical memorizing. Further experiments show that these supervised methods learn whether $y$ is a “prototypical hypernym” (i.e. a category), regardless of $x$, rather than learning a concrete relation between $x$ and $y$.

Our mathematical analysis reveals that said methods ignore the interaction between $x$ and $y$, explaining our empirical findings. We modify them accordingly by incorporating the similarity between $x$ and $y$. Unfortunately, the improvement in performance is incremental. We suspect that methods based solely on contextual features of single words are not learning lexical inference relations because contextual features might lack the necessary information to deduce how one word relates to another.

2 Experiment Setup

Due to various differences (e.g. corpora, train/test splits), we do not list previously reported results, but apply a large space of state-of-the-art supervised methods and review them comparatively. We observe similar trends to previously published results, and make the dataset splits available for replication.\[1\]

---

[1]: http://u.cs.biu.ac.il/~nlp/resources/downloads/
### Table 1: Datasets evaluated in this work.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Instances</th>
<th>#Positive</th>
<th>#Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kotlerman 2010</td>
<td>2,940</td>
<td>880</td>
<td>2,060</td>
</tr>
<tr>
<td>Bless 2011</td>
<td>14,547</td>
<td>1,337</td>
<td>13,210</td>
</tr>
<tr>
<td>Baroni 2012</td>
<td>2,770</td>
<td>1,385</td>
<td>1,385</td>
</tr>
<tr>
<td>Turney 2014</td>
<td>1,692</td>
<td>920</td>
<td>772</td>
</tr>
<tr>
<td>Levy 2014</td>
<td>12,602</td>
<td>945</td>
<td>11,657</td>
</tr>
</tbody>
</table>

2.1 Word Representations

We built 9 word representations over Wikipedia (1.5 billion tokens) using the cross-product of 3 types of contexts and 3 representation models.

2.1.1 Context Types

**Bag-of-Words** Uses 5 tokens to each side of the target word (10 context words in total). It also employs subsampling (Mikolov et al., 2013a) to increase the impact of content words.

**Positional** Uses only 2 tokens to each side of the target word, and decorates them with their position (relative to the target word); e.g. the $-1$ is a common positional context of cat (Schütze, 1993).

**Dependency** Takes all words that share a syntactic connection with the target word (Lin, 1998; Padó and Lapata, 2007; Baroni and Lenci, 2010). We used the same parsing apparatus as in (Levy and Goldberg, 2014).

2.1.2 Representation Models

**PPMI** A word-context positive pointwise mutual information matrix $M$ (Niwa and Nitta, 1994).

**SVD** We reduced $M$’s dimensionality to $k = 500$ using Singular Value Decomposition (SVD).

**SGNS** Skip-grams with negative sampling (Mikolov et al., 2013b) with 500 dimensions and 5 negative samples. SGNS was trained using a modified version of word2vec that allows different context types (Levy and Goldberg, 2014).

2.2 Labeled Datasets

We used 5 labeled datasets for evaluation. Each dataset entry contains two words $(x, y)$ and a label whether $x$ entails $y$. Note that each dataset was created with a slightly different goal in mind, affecting word-pair generation and annotation. For example, both of Baroni’s datasets are designed to capture hypernyms, while other datasets try to capture broader notions of lexical inference (e.g. causality). Table 1 provides metadata on each dataset, and the description below explains how each one was created.

(Kotlerman et al., 2010) Manually annotated lexical entailment of distributionally similar nouns.

(Baroni and Lenci, 2011) a.k.a. BLESS. Created by selecting unambiguous word pairs and their semantic relations from WordNet. Following Roller et al. (2014), we labeled noun hypernyms as positive examples and used meronyms, noun ccohyponyms, and random noun pairs as negative.

(Baroni et al., 2012) Created in a similar fashion to BLESS. Hypernym pairs were selected as positive examples from WordNet, and then permuted to generate negative examples.

(Turney and Mohammad, 2014) Based on a crowdsourced dataset of 79 semantic relations (Jurgens et al., 2012). Each semantic relation was linguistically annotated as entailing or not.

(Levy et al., 2014) Based on manually annotated entailment graphs of subject-verb-object tuples (propositions). Noun entailments were extracted from entailing tuples that were identical except for one of the arguments, thus propagating the existence/absence of proposition-level entailment to the noun level. This dataset is the most realistic dataset, since the original entailment annotations were made in the context of a complete proposition.

2.3 Supervised Methods

We tested 4 compositions for representing $(x, y)$ as a feature vector: **concat** $(\vec{x} \oplus \vec{y})$ (Baroni et al., 2012), **diff** $(\vec{y} - \vec{x})$ (Roller et al., 2014; Weeds et al., 2014; Fu et al., 2014), **only x** $(\vec{x})$, and **only y** $(\vec{y})$. For each composition, we trained two types of classifiers, tuning hyperparameters with a validation set: logistic regression with $L_1$ or $L_2$ regularization, and SVM with a linear kernel or quadratic kernel.

3 Negative Results

Based on the above setup, we present three negative empirical results, which challenge the claim that the methods presented in §2.3 are learning a relation between $x$ and $y$. In addition to our setup, these results were also reproduced in preliminary exper-
Lexical + Contextual

Table 2: The performance ($F_1$) of lexical versus contextual feature classifiers on a random train/test split with lexical overlap.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Lexical</th>
<th>+Contextual</th>
<th>Δ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kotlerman 2010</td>
<td>.346</td>
<td>.437</td>
<td>.091</td>
</tr>
<tr>
<td>Bless 2011</td>
<td>.960</td>
<td>.960</td>
<td>.000</td>
</tr>
<tr>
<td>Baroni 2012</td>
<td>.638</td>
<td>.802</td>
<td>.164</td>
</tr>
<tr>
<td>Turney 2014</td>
<td>.644</td>
<td>.747</td>
<td>.103</td>
</tr>
<tr>
<td>Levy 2014</td>
<td>.302</td>
<td>.370</td>
<td>.068</td>
</tr>
</tbody>
</table>

Supervised vs Unsupervised

While supervised methods were reported to perform better than unsupervised ones, this is not always the case. As a baseline, we measured the “vanilla” cosine similarity of $\mathbf{x}$ and $\mathbf{y}$, tuning a threshold with the validation set. This unsupervised symmetric method outperforms all supervised methods in 2 out of 5 datasets (Table 3).

Ignoring $x$’s Information

We compared the performance of only $\mathbf{y}$ to that of the best configuration in each dataset (Table 3). In 4 out of 5 datasets, the difference in performance is less than 5 points. This means that the classifiers are ignoring most of the information in $\mathbf{x}$. Furthermore, they might be overlooking the compatibility (or incompatibility) of $\mathbf{x}$ to $\mathbf{y}$. Weeds et al. (2014) reported a similar result, but did not address the fundamental question it beckons: if the classifier cannot capture a relation between $\mathbf{x}$ and $\mathbf{y}$, then what is it learning?

4 Prototypical Hypernyms

We hypothesize that the supervised methods examined in this paper are learning whether $\mathbf{y}$ is a likely “category” word – a prototypical hypernym – and, to a lesser extent, whether $\mathbf{x}$ is a likely “instance” word. This hypothesis is consistent with our previous observations (§3).

Though the terms “instance” and “category” pertain to hypernymy, we use them here in the broader sense of entailment, i.e. as “tends to entail” and “tends to be entailed”, respectively. We later show (§4.2) that this phenomenon indeed extends to other inference relations, such as meronymy.

4.1 Testing the Hypothesis

To test our hypothesis, we measure the performance of a trained classifier on mismatched instance-
category pairs, e.g. (banana, animal). For each dataset, we generate a set of such synthetic examples $S$, by taking the positive examples from the test portion $T^+$, and extracting all of its instance words $T^+_x$ and category words $T^+_y$.

$$T^+_x = \{x | (x, y) \in T^+\} \quad T^+_y = \{y | (x, y) \in T^+\}$$

We then define $S$ as all the in-place combinations of instance-category word pairs that did not appear in $T^+$, and are therefore likely to be false.

$$S = (T^+_x \times T^+_y) \setminus T^+$$

Finally, we test the classifier on a sample of $S$ (due to its size). Since all examples are assumed to be false, we measure the false positive rate as match error – the error of classifying a mismatching instance-category pair as positive.

According to our hypothesis, the classifier cannot differentiate between matched and mismatched examples ($T^+$ and $S$, respectively). We therefore expect it to classify a similar proportion of $T^+$ and $S$ as positive. We validate this by comparing recall (proportion of $T^+$ classified as positive) to match error (proportion of $S$ classified as positive). Figure 1 plots these two measures across all configurations and datasets, and finds them to be extremely close (regression curve: $\text{match error} = 0.935 \cdot \text{recall}$), thus confirming our hypothesis.

### 4.2 Prototypical Hypernym Features

A qualitative way of analyzing our hypothesis is to look at which features the classifiers tend to consider. Since SVD and SGNS features are not easily interpretable, we used PPMI with positional contexts as our representation, and trained a logistic regression model with $L_1$ regularization using concat over the entire dataset (no splits). We then observed the features with the highest weights (Table 4).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Top Positional Contexts of $y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kotlerman 2010</td>
<td>grave$<em>{-1}$, substances$</em>{+2}$, lend-lease$<em>{-1}$, poor$</em>{-2}$, bureaucratic$<em>{-1}$, physical$</em>{-1}$, dry$<em>{-1}$, air$</em>{-1}$, civil$_{-1}$</td>
</tr>
<tr>
<td>Bless 2011</td>
<td>other$<em>{-1}$, resembling$</em>{+1}$, such$<em>{+1}$, assemblages$</em>{-1}$, magical$<em>{-1}$, species$</em>{+1}$, any$<em>{-2}$, invertebrate$</em>{-1}$</td>
</tr>
<tr>
<td>Baroni 2012</td>
<td>any$<em>{-1}$, any$</em>{-2}$, social$<em>{-1}$, every$</em>{-1}$, this$<em>{-1}$, kinds$</em>{-2}$, exotic$<em>{-1}$, magical$</em>{-1}$, institute$<em>{-2}$, important$</em>{-1}$</td>
</tr>
<tr>
<td>Turney 2014</td>
<td>of$<em>{-1}$, inner$</em>{-1}$, including$<em>{+1}$, such$</em>{+1}$, considerable$<em>{-1}$, their$</em>{-1}$, extra$<em>{-1}$, types$</em>{-2}$, different$<em>{-1}$, other$</em>{-1}$</td>
</tr>
<tr>
<td>Levy 2014</td>
<td>psychosomatic$<em>{-1}$, unidentified$</em>{-1}$, auto-immune$<em>{+2}$, specific$</em>{-1}$, unspecified$<em>{-1}$, treatable$</em>{-2}$, any$_{-1}$</td>
</tr>
</tbody>
</table>

Table 4: Top positional features learned with logistic regression over concat. Displaying positive features of $y$.

![Figure 1: The correlation of recall (positive rate on $T^+$) with match error (positive rate on $S$) compared to perfect correlation (green line).](image)

Many of these features describe dataset-specific category words. For example, in Levy’s medical-domain dataset, many words entail “symptom”, which is captured by the discriminative feature psychosomatic$_{-1}$. Other features are domain-independent indicators of category, e.g. any$_{-1}$, every$_{-1}$, and kinds$_{-2}$. The most striking features, though, are those that occur in Hearst (1992) patterns: other$_{-1}$, such$_{+1}$, including$_{+1}$, etc. These features appear in all datasets, and their analogues are often observed for $x$ (e.g. such$_{-2}$). Even qualitatively, many of the dominant features capture prototypical or dataset-specific hypernyms.

As mentioned, the datasets examined in this work also contain inference relations other than hypernymy. In Turney’s dataset, for example, 77% of positive pairs are non-hypernyms, and $y$ is often a quality (coat $\rightarrow$ warmth) or a component (chair $\rightarrow$ legs) of $x$. Qualities and components can often be detected via possessives, e.g. of$_{+1}$ and their$_{-1}$. Other prominent features, such as extra$_{-1}$.
and exotic, may also indicate qualities. These examples suggest that our hypothesis extends beyond hypernymy to other inference relations as well.

5 Analysis of Vector Composition

Our empirical findings show that concat and diff are clearly ignoring the relation between \( x \) and \( y \). To understand why, we analyze these compositions in the setting of a linear SVM. Given a test example, \((x, y)\) and a training example that is part of the SVM’s support \((x_s, y_s)\), the linear kernel function yields Equations (1) for concat and (2) for diff.

\[
K(x \oplus y, x_s \oplus y_s) = x \cdot x_s + y \cdot y_s \quad (1)
\]

\[
K(y - x, y_s - x_s) = y \cdot x_s + y \cdot y_s - x \cdot y_s - y \cdot y_s \quad (2)
\]

Assuming all vectors are normalized (as in our experiments), the kernel function of concat is actually the similarity of the \( x \)-words plus the similarity of the \( y \)-words. Two dis-similarity terms are added to diff’s kernel, preventing the \( x \) of one pair from being too similar to the other pair’s \( x \) (and vice versa).

Notice the absence of the term \( x \cdot y \). This means that the classifier has no way of knowing if \( x \) and \( y \) are even related, let alone entailing. This flaw makes the classifier believe that any instance-category pair \((x, y)\) is in an entailment relation, even if they are unrelated, as seen in §4. Polynomial kernels also lack \( x \cdot y \), and thus suffer from the same flaw.

6 Adding Intra-Pair Similarity

Using an RBF kernel with diff slightly mitigates this issue, as it factors in \( x \cdot y \), among other similarities:

\[
K_{RBF}(y - x, y_s - x_s) = e^{-\frac{1}{2\sigma^2}(y - x - (y_s - x_s))^2} = e^{-\frac{1}{\sigma^2}(\bar{y}\bar{y} + \bar{x}\bar{x} + \bar{y}\bar{y} - \bar{x}\bar{y} - \bar{x}\bar{y} - (\bar{y}\bar{y} - \bar{x}\bar{y} - \bar{x}\bar{y} - \bar{x}\bar{x}))} \quad (3)
\]

A more direct approach of incorporating \( x \cdot y \) is to create a new kernel, which balances intra-pair similarities with inter-pair ones:

\[
K_{SIM}((x, y), (x_s, y_s)) = (x \cdot y_s y_s + x_s y_s + y y_s - x y_s - y y_s - x y_s - 2) \quad (4)
\]

While these methods reduce match error – match error = 0.618 · recall versus the previous regression curve of match error = 0.935 · recall – their overall performance is only incrementally better than that of linear methods (Table 5). This improvement is also, partially, a result of the non-linearity introduced in these kernels.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>LIN(concat)</th>
<th>LIN(diff)</th>
<th>RBF(diff)</th>
<th>SIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kotlerman 2010</td>
<td>.367</td>
<td>.187</td>
<td>.407</td>
<td>.332</td>
</tr>
<tr>
<td>Bless 2011</td>
<td>.634</td>
<td>.665</td>
<td>.636</td>
<td>.687</td>
</tr>
<tr>
<td>Baroni 2012</td>
<td>.745</td>
<td>.769</td>
<td>.848</td>
<td>.859</td>
</tr>
<tr>
<td>Turney 2014</td>
<td>.696</td>
<td>.694</td>
<td>.691</td>
<td>.641</td>
</tr>
<tr>
<td>Levy 2014</td>
<td>.229</td>
<td>.219</td>
<td>.252</td>
<td>.244</td>
</tr>
</tbody>
</table>

Table 5: Performance (F1) of SVM across kernels. LIN refers to the linear kernel (equations (1) and (2)), RBF to the Gaussian kernel (equation (3)), and SIM to our new kernel (equation (4)). Uses lexical train/test splits.

7 The Limitations of Contextual Features

In this work, we showed that state-of-the-art supervised methods for recognizing lexical inference appear to be learning whether \( y \) is a prototypical hypernym, regardless of its relation with \( x \). We tried to factor in the similarity between \( x \) and \( y \), yet observed only marginal improvements. While more sophisticated methods might be able to extract the necessary relational information from contextual features alone, it is also possible that this information simply does not exist in those features.

A (de)motivating example can be seen in §4.2. A typical \( y \) often has such as a dominant feature, whereas \( x \) tends to appear with such. These features are relics of the Hearst (1992) pattern “\( y \) such as \( x \)”. However, contextual features of single words cannot capture the joint occurrence of \( x \) and \( y \) in that pattern; instead, they record only this observation as two independent features of different words. In that sense, contextual features are inherently handicapped in capturing relational information, requiring supervised methods to harness complementary information from more sophisticated features, such as textual patterns that connect \( x \) with \( y \) (Snow et al., 2005; Turney, 2006).

Acknowledgements

This work was supported by the Adolf Messer Foundation, the Google Research Award Program, and the German Research Foundation through the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1). We thank Stephen Roller for his valuable insights.
References


