Machine Learning Techniques for Establishing the Provenance of Biological Interactions

Siwei Shen\textsuperscript{a}, Carlos Santos\textsuperscript{b}, David States\textsuperscript{b}, Güneş Erkan\textsuperscript{a}, Jing Gao\textsuperscript{b}, Dragomir R. Radev\textsuperscript{a,b,c}

\textsuperscript{a}Department of EECS, University of Michigan, Ann Arbor, MI 48109 USA, \textsuperscript{b}Bioinformatics Program, University of Michigan, Ann Arbor, MI 48109 USA, \textsuperscript{c}School of Information, University of Michigan, Ann Arbor, MI 48109 USA

ABSTRACT

Motivation: A substantial amount of knowledge about biomolecular interactions resides in the biomedical literature and has been increasingly made available as structured, curated database entries. Although recent research in biological assertion extraction has mostly focused on automatically detecting interacting entities, automated generation of provenance for interacting pairs remains of equal importance. We show that a number of machine learning algorithms can be used to directly establish sentence-level support for given entity-entity interactions in biological databases. We specifically focus on finding support for specific interaction entries in database assertions about protein-protein interactions.

Results: To find evidence backing up a biological interaction is to classify whether or not each sentence contains assertive support for the claimed interaction. We evaluated our method by comparing its classification against human judgement on a corpus of MEDLINE papers extracted from the BIND (Biomolecular Interaction Network Database). Each article in our experiment has at least one pair of biological entities known to interact with each other and whose interaction is supported by the article. The best two machine learning algorithms in our work achieve over 21% improvement on classification accuracy and over 6% improvement on F-measure, which is statistically significant.

Availability: Software code relevant to the work described in the paper are available upon request to the authors.

Contact: radev@umich.edu

1 INTRODUCTION

The sheer volume of biological literature and the complexity of information it contains far exceed the ability of single investigators to comprehensively study. The growing rate at which new articles are published has only aggravated the problem. Accordingly, accurately extracting, cataloging, and indexing this literature into structured databases is a critical requirement for investigators studying complex biological systems. For example, biological interaction extraction is extremely important for establishing the extent of signal pathway influence on target genes, or for understanding the behavior of cells under specific environmental conditions. Biological interaction information, however, is often a product of very specific experimental conditions and acquisition methods, ranging from high-throughput, proteome-scale interaction assays to detailed molecular biology experiments examining the interaction between individual biomolecules (e.g. protein-protein, protein-DNA, protein-small molecule, etc) in tightly constrained and well-understood experiments. The accurate cataloging and indexing of biological entity interaction data already reported from the biomedical literature has recently attracted a lot of attention in the research community ([14]). Cataloged and curated bases are not only essential in building models for complex networks like major cellular pathways or cell-wide simulations for systems biology; they also often serve as the main authoritative resource for experimenters wishing to gain in-depth knowledge about a cellular system that would otherwise remain deep in individual papers.

As an example of one of these curated pathways, Figure 1 [10] shows a model of a well-known signaling pathway, the Wnt Signaling Pathway. This pathway is only one of hundreds, and as currently understood, consists of over 50 interacting proteins. Maintaining this pathway current requires considerable human curation in order to remain accurate and current to the experimental literature. In the pathway, each interaction in turn is supported by a one to possibly hundreds of published experiments, each setting specific experimental parameters relating to the interaction, and each potentially having key information about that interaction that could be relevant to a researcher studying the behavior of this pathway. Only compounding the complexity of the problem are the even greater number of possible relations among them. At least two tasks need therefore be automated: 1) identifying interacting partners; and 2) verifying claimed interactions. We focus on the second problem in the current paper.

In recent years, there has been increasing availability of a variety of NLP (natural language processing) techniques,
ranging from syntactic tools (such as shallow/full parsers and grammar matching); to lexical analyses (for example keywords search and morphological analysis); and to statistical approaches (Bayesian classification, word statistics, and etc.). For an extensive review, readers should refer to [14]. We try to accomplish a task of classifying if a sentence is an affirmative assertion of specified interactions between biological entities. The primary aim of this project is to map existing database entries as accurately as possible to their source data, in order to establish a strong provenance link from each reported protein interaction to the exact sentences in the paper which is claimed to contain the interaction cited.

As a simplifying initial pre-processing step, sentences which do not mention the both interacting entity names are excluded from further processing. To classify the remaining sentences, we propose to use four machine learning algorithms – Naive Bayes, Support Vector Machine, Decision Tree, and TUMBL (Tripartite Update Method for Biased Learning). We adopt the implementation of the first three classifiers provided by WEKA (Waikato Environment for Knowledge Analysis), an open source package of machine learning algorithms written in Java [19]. The last algorithm has been recently developed by our research group as a general machine learning classification method. All algorithms use sentential features that encode properties of the individual sentence. We employ only lexical features that capture words of certain types, distance between indicative words, and many others surface-level characteristics.

Data in the current work consist of sentences extracted from a set of MEDLINE articles annotated as containing interacting partners by the BIND database as of September of 2003. Each paper corresponds to one or more pairs of entities whose interaction is manually reviewed by human curators for the BIND project and each interaction has been annotated as supported by the associated article.

We consider the base criteria for sentence inclusion to be the co-appearance of interacting entities in the paper and for sentences to be part of the abstract and body text of the paper (hence excluding references and titles, etc.). We recruit human annotators to classify the 3216 sentences thus extracted from the articles, and used their manual labeling as our gold standard. Classification accuracy and F-score are the performance metrics used. The latter is introduced by Keith van Rijsbergen in [17] as the harmonic mean (or conservative average) of precision and recall.

2 RELATED WORK

There are two tasks closely tied with each other in the knowledge acquisition of biological entity relations. In one approach, researchers try to automatically detect interacting pairs from scientific text. Conversely, however, people are interested in extracting supportive evidence of known interactions from relevant data sets. The current work falls into the second category.

There are extensive examples of named entity and relationship interaction reported in the literature, including automated annotation algorithms ([1, 3, 4, 6, 7, 12, 15, 18]) as well as methods to detect protein-protein interaction information ([2, 3, 9]). The methods employed share a mixture of text mining and indexing for terms which can be classified by Bayesian statistics ([18]), grammar matches ([16]), or string matches to known entities, as well as the use of partial and full parsers ([13]).

The inverse problem of extracting entities, however, is that of validating and resolving database interaction entries against their reported sources from the literature. These reverse links can serve both as validation and error-correction links, but most importantly, methods which can resolve them can give an automatic and extremely fine-grained provenance trail to human researchers interested in validating pathways or investigating them in depth. Similar in spirit (very different in goals), [8] studies the application of speculative sentences in bioscience literature and discusses the use of automated systems to identify them. [8] limits their scope to the MEDLINE abstracts only.

Establishing accurate provenance is essential in biology – the databases are only as valid as the direct assertions underlying each entry, and especially critical if such databases are to be used for derivative or cross-reference purposes. Interactions extracted from automated sources can be a powerful utility for researchers, but in the end they are only as reliable
as the underlying data. Spurious hits in result sets of thousands of automated interactions can substantially degrade the utility the entire dataset if the researcher must still validate each hit individually for accuracy. In particular, an automated system to validate independently these assertions would be just as essential for an accurate final database as the assertion generator in the first step.

Validation systems like ours can also assist in automatically establishing provenance information that can later serve as source material for further automated annotation. For instance, a validated provenance chain for a protein interaction could be resolved to the specific experimental section of the paper discussed, and the section then scanned for experimental details exactly describing how the interaction was originally established.

[3] extracts interaction information from the abstracts of scientific publications. Instead of syntactic analysis, it relies on lexical properties of the abstract sentences. One commonality between their work and ours is the use of “action verbs” indicative of object relations. While they use such verbs together with predetermined entity names to construct rules for recognizing interaction in text segments, we incorporate the verbs into many more features for our classifiers. [3] takes into account relative position of an action verb to its neighboring entities, but is limited to fairly simple pattern with the verb appearing between the entity names. As a consequence, their system is not capable of sifting out negative assertions, and nor is it able to identify interactions stated in other sentential forms (e.g. the verb preceding both entities), all of which are ubiquitous in bioscientific writings. Sentence 2 and 3 in Table 1 are two examples in which both entity names occur before the verb.

In terms of the classifier algorithms, we use several standard methods in the machine learning literature as well as a newly developed graph-based learning algorithm named TUMBL. The version of TUMBL we use in this work is improved from its initial introduction in [11], and will be further extended along directions of [20] and [5] that establish linkage between random walk and electric circuits.

3 PROBLEM DEFINITION

We first formalize the problem as follows:

Given an input instance (S, P), where S is a sentence from the molecular biology literature and P = (A, B) a pair of entities, we are to determine if S contains assertive evidence that backs up the alleged interaction between A and B. This is hence a binary classification problem. Here “interaction” represents a wide concept, including more fine-grained biological relations of the interacting partners, such as regulation, repression, translocation, direct binding, enzymatic interaction (phosphorylation), and etc.

As an example, Table 1 shows a sample MEDLINE article and a pair of biological entities known to have an interaction that is supported by this article. The top part of the table lists a number of sentences in this paper that provides evidence of the interaction between Cet1 and Ceg1 whereas those in the bottom part do not. It is obvious that the presence of the entity names do not necessarily imply a sentence being an assertion. Sentence 4 in the non-assertion set does not mention relations between the two entities whereas Sentence 5 phrases it as a hypothesis without asserting its existence. Besides the two negative examples shown here, negation has been known to affect classification accuracy [3]. Most of the time, negative assertions are signaled by certain negation-words such as “no”, “not”, “nor”, and etc. The simplest (though idealized) example would be “Protein A has been shown to not bind with B.” From Table 1, it is interesting to observe that the appearance of negation-words is also neither sufficient nor necessary condition of a sentence not being an assertion. For instance, Sentence 2 in the graph clearly affirms an interaction between Cet1 and Ceg1 even though it also contains a negating phrase “do not”. Therefore, the classification task we tackle is not a trivial matter of name entity searching, nor can it be accomplished by just matching indicative words.

4 APPROACH

This section includes description of our classification methods. In particular, we explain how they make use of the features designed specifically for this task.

4.1 Features

Features express properties of the sentences. Useful features would naturally encode sentential characteristics that correlate well with the class of the sentences. In the current work, we resort to only lexical features which have turned out to be both simple and efficient. Given that we are to determine if a sentence contains evidence of some known biological interaction, a wonderful resource for features is the entity pair associated with each sentence. Secondly, we have compiled a list of verbs that are often indicative of interactions in the biology domain. These words are what [3] refers to as “action-verbs”. As is reported by [3] and others, recognizing and hereafter excluding negative sentences remains a hard problem. Distinguishing between negative and affirmative assertions affects classification, information extraction, and natural language understanding in general, if not done appropriately. To address this challenge, we also collect a set of words often signaling negation or uncertainty. Table 4.1 displays the indicative verbs and the negation/uncertainty words used in our study. The inclusion of words like “if”, “test”, and “whether” might seem questionable given their high word frequency, yet experiments show that our methods achieve better performance with these words included on the devtest data. The majority of the features are built upon these special typed words (with stemming). All features are binary.

A subset of the features aim specifically at identifying negative assertions and/or uncertainties (like Sentence 5 in
Table 1. Sample MEDLINE article and the entity pairs of interest. The upper portion shows a number of assertive sentences supporting the relation of Cet1 and Ceg1. The lower portion includes negative examples in which sentences contain the interacting entities but fail to provide provenance of the interaction.

<table>
<thead>
<tr>
<th>Table 2.</th>
<th>List of action indicative verbs and negation words.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Action Indicative Verbs</strong></td>
<td>activate amplify antagonize associate bind block bound complex control coprecipitate degrade dimerize downregulate enhance heterodimerize homodimerize induce inhibit interact mediate phosphorylate precipitate recruit regulate repress stabilize stimulate transduce translocate trigger ubiquinate upregulate</td>
</tr>
<tr>
<td><strong>Words Signaling Negation/Uncertainty</strong></td>
<td>abolish absence cannot disable fail hypothesis if might may might may might not nor not prevent test unable uncertain whether</td>
</tr>
</tbody>
</table>

Table 1). Below is a list of several major categories of negative instances – sentences that contain interacting pairs but fail to assert a relation – prepared by human annotators participating in our study.

1. Figures and methods – e.g. “The figure shows immunoprecipitations of X, Y, ...”.
2. Statements of hypotheses – e.g. “If X phosphorylated Y, we would expect ...”.
3. Uncertain assertions – e.g. “Based on these results, it is not clear whether X binds Y”.
4. Ambiguous verbs – e.g. “X and Y co-activate gene Z.” Some readers think this implied interaction of X and Y, others do not.
5. Unresolved transitive interactions with unspecified or ambiguous interactions – e.g. “Z cooperates with X and Y to ...”.

Handling many of these cases is straightforward. For example, sentences corresponding to Case 1 might very well contain no interaction indicative verbs. In addition, sentences from the Methods and/or Figures section of experimental papers tend to be disproportionately long. In these cases, the number of clauses per sentence may vary as a discriminant feature. A clause of a sentence is defined as the contiguous text body between two consecutive punctuation marks, usually commas and periods. Although the length of a clause varies, we believe words in a same clause can generally be considered close to each other. Furthermore, words in a same clause are more likely to be syntactically and semantically related. Several of the features hence incorporate the notion of “same clause”, trying to predict if words of interested types are indeed interrelated by checking if they occur in a same clause. On the other hand, we handle Case 2 and 3 by treating “if”, “not”, and “whether” as negation/uncertainty lexicons. The last two cases are much more challenging to deal with. Syntactic features may have to be employed in these cases. We discuss further aspects of this in Section 6.1.

Table 4.1 lists all features used in the classification task. The particular order shown here is alphabetical with respect to feature name (these are not shown explicitly as they are not directly relevant to the outcome). The second column of the table has brief description of what each feature does.

Many of the features examine minimum/maximum action-entity, entity-entity, or average entity-action-entity distances. Another commonly evaluated property is whether words of certain types reside in a same clause. These features incorporate valid cases such as those assertions which contain action verbs and interacting entities within short context windows. In contrast, negative statements often contain negation words in the proximity of action verbs and/or entity names. The actual boundary numbers that delimit “small”, “medium”, and “large” are determined empirically for each individual feature by sampling the training data.

### 4.2 Algorithms

We discuss the TUMBL algorithm briefly in this section. A more detailed description is available in [11]. The remaining classifiers are standard learning methods in the literature, and were implemented as described by [19].

TUMBL (Tripartite Update Method for Biased Learning) is a graph learning algorithm recently developed at University of Michigan CLAIR (Computational Linguistics and Information Retrieval) research group. TUMBL uses a graph model to represent data in the classification problems, and bases its learning process on Markov random walk over the graph. The graph representation consists of all labeled training data, unlabeled test data, and nodes corresponding to binary feature objects. Edges exist only from some feature node to some data node. An edge exists if the feature on one end of it holds for the data instance on the other end. Edge weights reflect the importance of the incident feature. The graph can be considered tripartite because the component on the right hand side of the edges is split into...
two groups: labeled and unlabeled vertices. It is however a special case of a tripartite graph in that no links cross the two subcomponents.

Each node in TUMBL is associated with a class distribution, i.e. the probability of the node being assigned to each class. In binary classification problems like ours, the class distribution is hence a pair. The class distribution of feature nodes reflects a feature’s polarity, or how positive or negative prone the feature is. Class distributions of labeled data are definitive and stay constant over time. Those of feature and unlabeled nodes are initialized to neutral, (0.5, 0.5), and are updated subsequently. We assign equal weights to all binary features in our classification task, i.e. all edge weights are set to 1.

To update the class distribution of feature and unlabeled objects, TUMBL performs a sequence of 3 random walk steps: 1) from the labeled data to the feature nodes; 2) from the feature nodes to the unlabeled ones; and 3) from the unlabeled nodes back to the feature objects. The transition matrix of each random walk is defined by the weights of the edges connecting the two vertex sets involved in that step. The random walk steps propagate class distributions (multiplying the class distribution matrix to the transition matrix) from the labeled to the unlabeled through features. Once very 3 steps, TUMBL restarts from the labeled instances, and none of the random walk steps goes to the labeled objects. This way, labeled data serve solely as information source in TUMBL algorithm. Because the transitions follow links that connect a data point and a binary feature, labeled instances would effectively “push out” their class information to the unlabeled instances that share similar feature values, hence achieving the purpose of learning. After TUMBL terminates, we classify each unlabeled instance according to its final class distribution, choosing the class with higher probability over the other.

A comprehensive description of this algorithm is beyond the scope of this paper but may be found in [11]. We include the pseudocode of TUMBL in Figure 2 and a snapshot of the software visualizing the learning process of TUMBL in Figure 4.2.

<table>
<thead>
<tr>
<th>Feature Id</th>
<th>Feature Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>If some action verb and one of the interacting entities co-appear in a same clause.</td>
</tr>
<tr>
<td>2</td>
<td>If some action verb co-appears with both entities of the interaction in a same clause.</td>
</tr>
<tr>
<td>3</td>
<td>If the sentence has at least one action verb.</td>
</tr>
<tr>
<td>4</td>
<td>If the sentence has at least 3 action verbs.</td>
</tr>
<tr>
<td>5</td>
<td>If the sentence has more than 5 clauses.</td>
</tr>
<tr>
<td>6</td>
<td>If the sentence has at least 2 negation words.</td>
</tr>
<tr>
<td>7</td>
<td>If the sentence has at least 1 negation word.</td>
</tr>
<tr>
<td>8</td>
<td>If both interacting entities co-appear in a same clause.</td>
</tr>
<tr>
<td>9</td>
<td>If the maximum distance between an action verb and an interacting entity is small (d ≤ 13).</td>
</tr>
<tr>
<td>10</td>
<td>If the maximum average distance between an action verb and the two interacting entities is small (d ≤ 10).</td>
</tr>
<tr>
<td>11</td>
<td>If the maximum distance between the two interacting entities is small (d ≤ 11).</td>
</tr>
<tr>
<td>12</td>
<td>If the minimum distance between an action verb and an interacting entity is large (d &gt; 9).</td>
</tr>
<tr>
<td>13</td>
<td>If the minimum distance between an action verb and an interacting entity is medium (2 &lt; d ≤ 9).</td>
</tr>
<tr>
<td>14</td>
<td>If the minimum distance between an action verb and an interacting entity is small (d ≤ 2).</td>
</tr>
<tr>
<td>15</td>
<td>If the minimum average distance between an action verb and the two interacting entities is large (d &gt; 11).</td>
</tr>
<tr>
<td>16</td>
<td>If the minimum average distance between an action verb and the two interacting entities is medium (5 &lt; d ≤ 11).</td>
</tr>
<tr>
<td>17</td>
<td>If the minimum average distance between an action verb and the two interacting entities is small (d ≤ 5).</td>
</tr>
<tr>
<td>18</td>
<td>If the minimum distance between a negation word and an action verb is large (d &gt; 5).</td>
</tr>
<tr>
<td>19</td>
<td>If the minimum distance between a negation word and an action verb is medium (2 &lt; d ≤ 5).</td>
</tr>
<tr>
<td>20</td>
<td>If the minimum distance between a negation word and an action verb is small (d ≤ 2).</td>
</tr>
<tr>
<td>21</td>
<td>If the minimum distance between the two interacting entities is large (d &gt; 9).</td>
</tr>
<tr>
<td>22</td>
<td>If the minimum distance between the two interacting entities is medium (2 &lt; d ≤ 9).</td>
</tr>
<tr>
<td>23</td>
<td>If the minimum distance between the two interacting entities is small (d ≤ 2).</td>
</tr>
<tr>
<td>24</td>
<td>If some action verb and some negation word co-appear in a same clause.</td>
</tr>
</tbody>
</table>

Table 3. Binary features used in our classification problem.
TUMBL($n$, $d_0$)
// $n$: number of iterations; $d_0$: damping factor
Initialize class distribution matrices $X$, $Y_0$, and $Z_0$
for labeled nodes, feature nodes, and unlabeled nodes.

$d = 1$

for $t = 1 : n$

\[
Y_t = d \times T_t^T X + Y_{t-1}
\]
Row-normalize $Y_t$

\[
Z_t = d \times T_s Y_t + Z_{t-1}
\]
Row-normalize $Z_t$

\[
Y_t = d \times T_s^T Z_t + Y_t
\]
Row-normalize $Y_t$

$d = d \times d_0$

endfor

for $i = 1 : \|U\|$

if $Z_n(i, 1) > Z_n(i, 2)$ then
  Assign positive to instance
else if $Z_n(i, 1) < Z_n(i, 2)$ then
  Assign negative to instance
else
  Assign more probable class to instance
endif

endfor

Fig. 2. TUMBL Algorithm.

Fig. 3. Snapshot of the visualization software for TUMBL.

5.2 Human Annotation

For each (article, interaction) pair, we drew from the article sentences that contained both entity names in the specified interaction – resulting in 3216 sentences. There were duplicates in these 3216 because a sentence from some articles could be extracted multiple times for distinct interactions of interest.

Annotation was developed by a voting system deployed as a PHP web interface to a relational database (MS SQL Server 2000). Thirteen voters enrolled, of which 1 was a faculty member and 12 were graduate students or research scientists with training in molecular biology. Voters were randomly assigned and identified via unique login to roughly equivalent-sized batches of sentences and instructed to vote over a one week period as time permitted. Voters were instructed to vote in small batches and in short intervals to prevent fatigue. At login into the voting system, each voter was presented with sequential batches of 10 sentences at a time and instructed to assign vote of “yes” to each sentence where the two proteins listed for the sentence interacted directly (e.g. a physical interaction or a statement of two proteins cooperating as a complex). Voters were instructed to vote “no” for sentences where the protein names were not mentioned, for hypothesis statements (e.g. “we test interaction between A and B”), and for sentences where no interaction was mentioned. Voters were permitted to vote multiple times on the same sentences to accommodate changes in opinion, and each voter was shown the latest previous vote made by themselves or by another voter, for the any sentences in the batch where such votes existed. In the database, all votes were tallied in a cumulative manner, and we maintained a time stamped record of votes-by-voters throughout the project, recording every vote made, even in cases where more than one vote was made for each sentence. Figure 5.2 shows a snapshot of the GUI (graphical user interface) for the annotation task.

Table 4. Four interactions recorded by BIND to be supported by PubMed article 11283018.

<table>
<thead>
<tr>
<th>Interaction Id</th>
<th>Entity 1 (Symbol)</th>
<th>Entity 2 (Symbol)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6096</td>
<td>Bcl-XL</td>
<td>Bak1</td>
</tr>
<tr>
<td>6097</td>
<td>Bcl-XL</td>
<td>BaxB</td>
</tr>
<tr>
<td>6098</td>
<td>Bcl-XL</td>
<td>BCL2L12</td>
</tr>
<tr>
<td>6099</td>
<td>Bcl-XL</td>
<td>BimL</td>
</tr>
</tbody>
</table>

Fig. 4. Snapshot of the GUI for protein interaction annotation.

To measure inter-judge agreement, we examine all sentences that are annotated by more than one reviewer. In case where a sentence is scored by more than two judges, we...
consider all possible pairs of annotations when calculating inter-judge agreement. For example, if an instance receives 3 votes, 2 of which are 1 (yes) and the other 0 (no). The instance would then contribute 1 (pair) to the agreed set and 2 (pairs) to the disagreed set. Table 5.2 reports the counts of agreed and disagreed instances. Although only a subset of the sentences get multiple reviewers, the multiply scored instances are randomly picked. Inter-judge agreement is 226/220+220+226+226 = 0.89. While this number represents a relatively high level of consistency among our human reviewers, we excluded the 57 disagreed sentences from our experiments.

5.3 Data Split

Given the expense of human annotation, we left out some data for future extension and/or development of new classifiers for this problem. The rest of the data are divided into training, dev/test, and test set. Test data remained intact until we reported final performance of the algorithms.

- Training - 1000 sentences
- Dev/test - 500 sentences
- Test - 500 sentences
- Untouched - 1159 sentences (excluding the 57 disagreed)

6 RESULTS AND ANALYSIS

We measure accuracy and F-score of classification results. Whereas accuracy is well understood, F-score is computed as the follows,

\[ F_{\text{score}} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \]

Using F-score (instead of Precision or Recall) can avoid awarding systems that achieve perfect precision but very low recall or vice versa (e.g. the baseline algorithm that assigns all instances uniformly to the majority class). Improving the F-score of classification remains a big challenge in the literature.

We first present in Table 6 the performance achieved by all four classifiers on the test data when they are trained on the entire training set. As a reference, we also show the baseline numbers, which corresponds to classifying all test sentences to the majority class (1). The numbers represent different levels of effectiveness of these machine learning algorithms. All have achieved improvement over the baseline with respect to both metrics. In particular, TUMBL and SVM (support vector machines) outperform the baseline by over 6.2% on F-score and over 21.1% on accuracy. Both are statistically significant.

6.1 Error Analysis

We examine the sentences that our machine classifiers label wrong, and notice all algorithms get many more false-positives than false-negatives (a ratio of around 10:1). Table 6.1 shows the confusion matrix of the classification results of TUMBL on the test data. To interpret the numbers, there exist a large number (134) of negative instances that are classified as positive. In contrast, only 14 truly positive data points are labeled the other way. The most probable cause is that features designed to discover clues for negativeness or uncertainties are not yet robust enough. As remarked earlier in Table 1, several non-assertion cases are beyond the capabilities of the lexical-level features. We are showing a small list of real examples from the test set for which human reviewers mark negative, but the learning methods are tricked to classify otherwise.

- SLP-76 vs. FYN-T – “Immunofluorescence co-localization of FYB with FYN-T and SLP76."
- SLP-76 vs. SLAP-130 – “These observations place SLAP-130 downstream of SLP-76 in CRP-induced signaling.”
- Bim vs. Bcl-2 – “Bim induces apoptosis which can be inhibited by the general caspase inhibitor p35 and Bcl-2 but not by CrmA.
- Nkx-2.5 vs. GATA-4 – “MBP-GATA-4 pull-down assays were used to investigate the interaction of Nkx-2.5 and GATA-4 in solution.”

The first three examples trap machine classifiers by mentioning a third object having interaction with the one of the entities of interest. Lexical features are particularly vulnerable in such cases because the entities of interest are usually close to each other in the sentences, and sometimes, an action verb (e.g. “induce”) can also appear in the very proximity of the two entities. We expect the use of syntactic information is necessary to resolve problems of this sort. For example, a syntactic parsing of the second sentence would recognize that the verb “induced” is attached to CRP instead of the two we are about, hence concluding the sentence having no action verbs (relevant to the two entities of interest). The reason human

<table>
<thead>
<tr>
<th>Vote 1</th>
<th>Vote 2</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>27</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>226</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>220</td>
</tr>
</tbody>
</table>

Table 5. Inter-judge agreement on multiply scored sentences.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.5780</td>
<td>0.7326</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>0.6880</td>
<td>0.7459</td>
</tr>
<tr>
<td>Support Vector Machine</td>
<td>0.7000</td>
<td>0.7781</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>0.6780</td>
<td>0.7330</td>
</tr>
<tr>
<td>TUMBL</td>
<td>0.7040</td>
<td>0.7879</td>
</tr>
</tbody>
</table>

Table 6. Classification accuracy and F-score achieved by various algorithms on the test data.
Table 7. Confusion matrix of TUMBL classifying the test data.

<table>
<thead>
<tr>
<th></th>
<th>Truly 1</th>
<th>Truly 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classified as 1</td>
<td>275</td>
<td>14</td>
</tr>
<tr>
<td>Classified as 0</td>
<td>134</td>
<td>77</td>
</tr>
</tbody>
</table>

The system we presented may be a useful utility for automated database curation and provenance extension, both as a verification utility and as an update utility for when the database becomes outdated and needs to be re-established from an new corpus. The system could also serve as a useful adjunct to information extraction systems that rely on contextual information to automatically extract ancillary facts about specific entries in existing databases.

ACKNOWLEDGEMENTS

This work was supported in part by NIH Grant RO1-LM008106.

The authors also owe special thanks to Jeremy Phillips, Anura Hewagama, Kristen Veresh, Baxter Allen, Yili Chen, Rajasree Mennon, Damian Fermin, Ji Chen, Heather Grifka, Yin Xu, Jacob Balazer, Jahnna Otterbacher, Chenyue Xing, and Yang Ye for their kind help in annotation and various other parts of the paper.

REFERENCES


