Robust textual inference via graph matching

Aria D. Haghighi
Dept. of Computer Science
Stanford University
Stanford, CA
aria42@stanford.edu

Andrew Y. Ng
Dept. of Computer Science
Stanford University
Stanford, CA
ang@cs.stanford.edu

Christopher D. Manning
Dept. of Computer Science
Stanford University
Stanford, CA
manning@cs.stanford.edu

Abstract

We present an automated system for deciding whether a given sentence is entailed from a body of text. Each sentence is represented as a directed graph (extracted from a dependency parser) in which the nodes represent words or phrases, and the links represent syntactic and semantic relationships. A learned graph matching cost is used to measure how much of the semantic content of the sentence is contained in the text. We present results on the Recognizing Textual Entailment (RTE) dataset (Dagan et al., 2005), compare to other approaches, discuss common classes of errors, and discuss directions for improvement.

1 Introduction

A fundamental goal of NLP is the production of robust and accurate systems semantic inference. This A fundamental stumbling block for several NLP applications is the lack of a robust and accurate semantic inference. For instance, question answering systems must be able to recognize, or infer, an answer which may be expressed differently from the query. Information extraction systems must also be able to infer propositions and recognize the variability of equivalent linguistic expressions. Document Summarization systems must generate succinct sentences which express the same content as the original document. In Machine Translation evaluation, we must be able to recognize legitimate translations which structurally differ from our baseline.

One sub-task underlying these applications is the ability to *recognize* semantic entailment; whether one piece of text follows from another. In contrast to recent work (Moldovan et al., 2003), which have successfully utilized logic-based abductive approaches to inference, we utilize a graph-based representation of sentences, and use graph matching techniques to measure the semantic overlap of text. Graph matching techniques have proved to be a useful approach to tractable approximate matching in other domains including computer vision. Additionally in the domain of language, graphs provide a natural way to express the lexical dependencies between words and phrases in a sentence. Furthermore, graph matching also has the advantage of providing a framework for structural matching of phrases that would be difficult to resolve at the level of individual words.

2 Task Definition and Data

We describe our approach in the context of this years “Recognizing Textual Entailment” (RTE) challenge (Dagan et al., 2005), but note that our approach easily extends to other related inference tasks. The system presented here was one component of our research group’s RTE submission (Suppressed, 2005) which was the top-ranking system according to one of two evaluation metrics.

In this domain, we are given several pairs, each consisting of two parts: 1) the *text*, a small passage

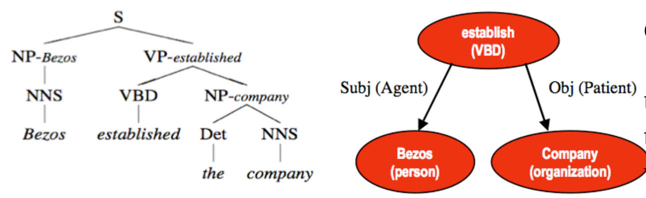

Figure 1: An example parse tree and the corresponding dependency graph. Each phrase of the parse tree is annotated with its head word, and the parenthetical edge labels in the dependency graph correspond to semantic roles.

[1], and the *hypothesis*, a single sentence. Our task is to decide if the hypothesis is entailed by the text. Here, entails does not mean strict *logical* implication, but roughly means that a competent speaker with minimal world-knowledge would infer the hypothesis from the the text. For a flavor of the nature (and difficulty) of the task, see Table 2.

For purposes of comparison, we give results on the data provided for the RTE task which consists of 567 development pairs and 800 test pairs. We will use the following toy example to illustrate our representation and matching technique:

Text: In 1994, Amazon.com was founded by Jeff Bezos.

Hypothesis: Bezos established a company.

3 Semantic Representation

Perhaps the most common representation of text for assessing content is "Bag-Of-Words" or "Bag-of-N-Grams" (Papineni et al., 2001). However, such representations lose syntactic information which can be essential to determining entailment. Consider a Question Answer system searching for an answer to *When was Israel established ?* A representation which did not utilize syntax would probably enthusiastically return an answer from: *The National Institute for Psychobiology in Israel was established in 1979*.

In this example it's important to try to match relationships as well as words. In particular, any answer to the question should preserve the dependency between *Israel* and *established* in the question. However, in the proposed answer, the expected dependency is missing.

Our approach is to view sentences as graphs between words and phrases, where dependency relationships are characterized by the path between the words. This approach has been successfully used to characterize semantic relationships such as hypernyms (Snow et al., 2004).

Given this representation, we judge entailment by measuring not only how many of the *hypothesis* vertices are matched to the *text* but also how well the relationships between vertices in the *hyposthesis* are preserved in their counter parts in the text. For the remainder of the section we outline how we produce graphs from text, and in the next section we introduce our graph matching model.

3.1 From Text To Graphs

Starting with raw English text, we use a version of the parser described in (Klein and Manning, 2003), to obtain a parse tree. Then, using a slightly modified versions of Collins' head propagation (Collins, 1999), we derive a dependency tree representation of the sentence. The labels of the edges in the dependency graph are given according to a hand-created set of `tgrep` expressions. These edges represent "surface" syntax relationships such as `subj` for subject and `amod` for adjective modifier, similar to the relations in *Minipar* (Lin and Pantel, 2001). The dependency graph is the basis for our graphical representation, but it is enhanced in the following ways:

1. Collapse Collocations and Named-Entities: We "collapse" dependency nodes which represent named entities (e.g., *Jeff Bezos* in Figure 2) and also collocations, including verbs and their particles (e.g. , *blow_off* in *He blew off his work*).

2. Dependency Folding : As in (Lin and Pantel, 2001), we found it useful to fold dependencies (such as modifying prepositions) so that modifiers became labels connecting the modifier's governor and dependent directly. For instance, in Figure 2, we have changed *in* from a word into a relation between its head verb and the head of its NP complement.

3. Semantic Role Labeling: We also augment the graph with the output of the Semantic Role La-

[1]Usually a single sentence, but occasionally longer

Task	Text	Hypothesis	Entailed
Question Answer (QA)	Prince Charles was previously married to Princess Diana, who died in a car crash in Paris in August 1997.	Prince Charles and Princess Diana got married in August 1997.	False
Machine Translation (MT)	Sultan Al-Shawi, a.k.a the Attorney, said during a funeral held for the victims, ”They were all children of Iraq killed during the savage bombing.”.	The Attorney, said at the funeral, ”They were all Iraqis killed during the brutal shelling.”.	True
Comparable Documents (CD)	Napster, which started as an unauthorized song-swapping Web site, has transformed into a legal service offering music downloads for a monthly fee.	Napster illegally offers music downloads.	False
Paraphrase Recognition (PP)	Kerry hit Bush hard on his conduct on the war in Iraq.	Kerry shot Bush.	False
Information Retrieval (IR)	The country's largest private employer, Wal-Mart Stores Inc., is being sued by a number of its female employees who claim they were kept out of jobs in management because they are women.	Wal-Mart sued for sexual discrimination.	True

Table 1: Some Textual Entailment examples. The first last three demonstrate some of the harder instances

beler of (Toutanova et al., 2005). Each predicate adds an arc labeled with the appropriate semantic role to the head of the argument phrase. This helps to create links between words which share a deep semantic relation not evident through the surface syntax. Additionally, modifying phrases are labeled with their semantic types (e.g., the *Temporal* edge in the Figure 2) which should be useful Question Answering type tasks.

4. Coreference Links: Using a co-rereference resolution tagger, `coref` links are added throughout the graph. These links allowed connecting the referent entity to the vertices of the referring vertex. Also in the case of multiple sentence texts, it is our only “link” in the graph entities in the two sentences.

For the remainder of the paper, we will refer to the text as T and hypothesis as H, and will speak of them in graph terminology. In addition we will use H_V and H_E to denote the vertices and edges, respectively, of H.

4 Entailment by Graph Matching

We take the view that a hypothesis is entailed from the text when the cost of matching the hypothesis graph to the text graph is low. For the remainder of this section, we outline a general model for assigning a match cost to graphs.

For hypothesis graph H, and text graph T, a *matching* M is a mapping from the vertices of H to those of T. For vertex v in H, we will use $M(v)$ to denote its “match” in T. As is common in machine translation, we allow nodes in H to map to a fictitious NIL vertex in T if necessary. Suppose the cost of matching M is Cost(M). If $\mathcal{M}$ is the set of such matchings, we define the cost of matching H to T:

$$\text{MatchCost}(H, T) = \min_{M \in \mathcal{M}} \text{Cost}(M) \qquad (1)$$

Suppose we have a model, VertexSub$(v, M(v))$, which gives us a cost in $[0, 1]$, for substituting vertex v in H for $M(v)$. One natural cost model is to use the normalized cost for each of the substitutions our matching makes:

$$\text{Cost}(M) = \frac{1}{Z}\sum_{v \in H_V} w(v)\text{VertexSub}(v, M(v)) \tag{2}$$

Here, $w(v)$ represents the weight or relative importance for vertex v, and $Z = \sum_{v \in H_V} w(v)$ is a normalization constant. In our implementation, the weight of each vertex was based on the part-of-speech tag of the word or the type of named entity, if applicable. However, there are several other possibilities including using tf-idf weights for words and phrases.

Notice that when $\text{Cost}(M)$ takes the form of (2), computing $\text{MatchCost}(H, T)$ is equivalent to finding the minimal cost bipartite graph-matching, which can be efficiently computed using linear programming.

As (Punyakanok et al., 2004) demonstrated, models which also match syntactic relationships between words can outperform bag-of-words models for TREC QA answer extraction. So it should be advantageous for our cost-model to incorporate some measure of how relationships in H are preserved in T under M. Ideally, a matching, should preserve all local relationships; i.e, if $v \to v' \in H$, then $M(v) \to M(v') \in T$. When this condition holds for all edges in H, H and T are isomorphic.

What we would like is an *approximate* notion of isomorphism, where we penalize the distortion of each edge relation in H. Consider an edge $e = (v, v') \in H_E$, and let $M(e)$ be the path from $M(v)$ to $M(v')$ in T.

Again, suppose we have a model, $\text{PathCost}(e, M(e))$ for assessing the "cost" of substituting a direct relation $e \in M$ for its counterpart, $M(e)$, under the matching. This leads to a formulation similar to (2), where we consider the normalized cost of substituting each edge relation in H with a path in T:

$$\text{RelationCost}(M) = \frac{1}{Z}\sum_{e \in H_E} w(e)\text{PathSub}(e, M(e)) \tag{3}$$

where $Z = \sum_{e \in H_E} w(e)$ is a normalization constant. As in the vertex case, we have weights for each hypothesis edge, $w(e)$, based upon the edge's label; typically subject and object relations

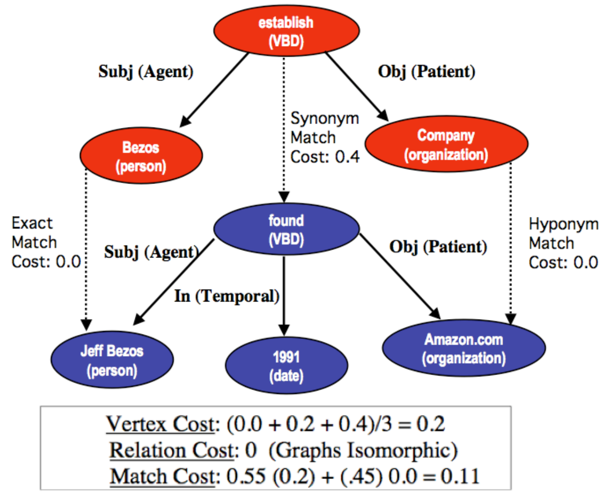

Figure 2: Example graph matching ($\alpha = 0.55$) for example pair. Dashed lines represent mapping.

are more important to match than others. Our final matching cost is given by a convex mixture of the vertex and relational match costs: $\text{Cost}(M) = \alpha\text{VertexCost}(M) + (1 - \alpha)\text{RelationCost}(M)$.

Notice that minimizing $\text{Cost}(M)$ is computationally hard since if our PathSub model assigns zero cost only for preserving edges, then $\text{RelationCost}(M) = 0$ if and only if H is isomorphic to a subgraph of T. As an approximation, we can efficiently find the matching M^* which minimizes $\text{VertexCost}(\cdot)$; we then perform local greedy hill-climbing search, beginning from M^*, to approximate the minimal matching. In practice, this approximation appears to perform rather well.

4.1 Relationship to Machine Translation

The model presented in this section bears many resemblances to a syntactic translation alignment model, where the source text is the target and the hypothesis the source. In particular, we can think of the matching as an alignment, and our equation (1) is a typical approximation of the translation model alignment probability.

5 Node and Edge Substitution Models

In the previous section we described our graph matching model in terms our VertexSub model, which gives a cost for substituting one graph vertex for another, and PathSub, which gives a cost for substituting the path relationship between two paths in one graph for that in another. We now outline these models.

Our VertexSub$(v, M(v))$ model is based upon a sliding scale, where progressively higher costs are given based upon the following conditions:

- **Exact Match**: v and $M(v)$ are identical words/phrases
- **Stem Match**: v and $M(v)$'s stems match or one is a derivational form of the other e.g., matching *coaches* to *coach*.
- **Synonym Match:** v and $M(v)$'s are synonyms according to *WordNet* (Fellbaum, 1998). In particular we use the top 3 senses of both words to determine synsets.
- **Hypernym Match:** v is a hypernym of $M(v)$ according to *WordNet*. Note that this feature is assymmetric. We do not match v to a hyponym $M(v)$.
- `WordNet Similarity:` v and $M(v)$ are similar according to `WordNet::Similarity` (Pedersen et al., 2004). In particular, we use the measure described in (Resnik, 1995). We found it useful to only use similarities above a fixed threshold to ensure precision.
- **LSA Match:** v and $M(v)$ are distributionally similar according to a freely available Latent Semantic Indexing package, or for verbs similar according to *VerbOcean* (Chklovski and Pantel, 2004). [2]
- **No Match:** $M(v)$ is NIL

Although the above conditions often produce reasonable matchings between text and hypothesis, we found that the recall of these lexical resources to be far from adequate. More robust lexical resources would almost certainly boost performance a significant amount.

Our PathSub model is also based upon a sliding scale cost based upon the following conditions:

- **Exact Match:** $M(v) \rightarrow M(v')$ is an en edge in T with the same label.
- **Partial Match:** $M(v) \rightarrow M(v')$ is an en edge in T, not necessarily with the same label.

[2] Available at `http://infomap.stanford.edu`

- **Ancestor Match:** $M(v)$ is an ancestor of $M(v')$. We use an exponential increasing cost for longer distance relationships.
- **Kinked Match:** $M(v)$ and $M(v')$ share a common parent or ancestor in T. We use an exponentially increasing cost based on the maximum of the node's distances to their least common ancestor in T.

These conditions capture many of the common way in which relationships between entities are distorted in semantically related sentences.

Give Justification for these conditions

5.1 Learning Weights

Is it possible to learn weights for the relative importance of the conditions in the VertexSub and PathSub models? Consider the case where match costs are given only by equation (2) and vertices are weighted uniformly ($w(v) = 1$). Suppose that $\Phi(v, M(v))$ is a vector of features [3] indicating the cost according to each of the conditions listed for matching v to $M(v)$. Also let w be weights for each element of $\Phi(v, M(v))$. Then we can model the substitution cost for a given matching as :

$$\text{VertexSub}(v, M(v)) = \frac{\exp\left(w^T \Phi(v, M(v))\right)}{1 + \exp\left(w^T \Phi(v, M(v))\right)}$$

Letting $s(\cdot)$ be the 1-sigmoid function used in the right hand side of the equation above, our final matching cost as a function of w is given by

$$c(H, T; w) = \min_{M \in \mathcal{M}} \frac{1}{|H_V|} \sum_{v \in H} s(w^T \Phi(v, M(v))) \tag{4}$$

Suppose we have a set of text/hypothesis pairs, $\{(T^{(1)}, H^{(1)}), \ldots, (T^{(n)}, H^{(n)})\}$, with labels $y^{(i)}$ which are 1 if $H^{(i)}$ is entailed by $T^{(i)}$ and 0 otherwise. Then we would like to choose w to minimize costs for entailed examples and maximize it for non-entailed pairs:

$$\ell(w) = \sum_{i:y^{(i)}=1} \log c(H^{(i)}, T^{(i)}; w) + \sum_{i:y^{(i)}=0} \log(1 - c(H^{(i)}, T^{(i)}; w))$$

[3] In the case of our "match" conditions, these features will be binary

Unfortunately, $\ell(w)$ is not a convex function. Notice that the cost of each matching, M, implicitly depends on the current setting of the weights, w. It can be shown that since each $c(H,T;w)$ involves minimizing $M \in \mathcal{M}$, which depends on w, it is not convex (**??**). Therefore, we can't hope to globally optimize our cost functions over w.

One approach is to use coordinate ascent over M and w. Suppose that we begin with arbitrary weights and given these weights choose $M^{(i)}$ to minimize each $c(H^{(i)}, T^{(i)}; w)$. Then we use a relaxed form of the cost function where we use the matchings found in the last step:

$$\hat{c}(H^{(i)}, T^{(i)}; w) = \frac{1}{|H_V|} \sum_{v \in H} s(w^T \Phi(v, M^{(i)}(v)))$$

Then we maximize w with respect to $\ell(w)$ witch each $c(\cdot)$ replaced with the cost-function $\hat{c}(\cdot)$. This step involves only logistic regression. We repeat this procedure until our weights converge.

Our preliminary experiments revealed that this procedure did not yield weights which improved performance very much our hand-set initializations. We believe this to be the case largely because of the presence of several local maxima and ridges in the parameter space. In the future, we hope to find better approximation techniques to this problem.

6 Checks

One systematic source of error coming from our basic approach is the implicit assumption of upwards monotonicity of entailment; i.e., if T entails H then adding *more* words to T should also give us a sentence which entails H. This assumption, also made by recent abductive approaches (Moldovan et al., 2003; Harabagiu et al., 2000), does not hold for several classes of examples. Below we outline the most common types of cases [4] that we check for after graph matching:

Negation Check

Text: Clinton's book is not a bestseller

Hypothesis: Clinton's book is a bestseller

To catch such examples, we check that each hypothesis verb is not matched to a text word which is negated (unless the verb pairs are antonyms) and vice versa. In this instance, the *is* in H, denoted by *is*$_H$, is matched to *is*$_T$ which has a negation modifier, *not*$_T$, absent for *is*$_H$. So the negation check is failed.

Factive Check

Text: Clonaid claims to have cloned 13 babies worldwide.

Hypothesis: Clonaid has cloned 13 babies.

Non-factive verbs (*claim*, *think*, *charged*, etc.) in contrast to factive verbs (*know*, *regret*, etc.) have sentential complements which do not represent true propositions. We detect such cases, by checking that each verb in H that is matched in T does not have a non-factive verb for a parent.

Superlative Check

Text: The Osaka World Trade Center is the tallest building in Western Japan.

Hypothesis: The Osaka World Trade Center is the tallest building in Japan.

In general superlative modifiers, (*most*, *biggest*, etc.), invert the typical monotinicity of entailment and must be handled as special cases. For any noun, n, with a superlative modifier (part-of-speech JJS) in H, we must ensure that all modifier relations of $M(n)$ are preserved in H. In this example, *building*$_H$ has a superlative modifier *tallest*$_H$, so we must ensure that each modifier relation of *Japan*$_T$, a noun dependent of *building*$_T$, has a *Westren*$_T$ modifier not in H. So its fails the superlative check.

During error analysis on the development set, we spotted the following cases where our VertexSub function erroneously labeled vertices as similair, and required special case consideration:

- **Antonym Check:** We consistently found that the `WordNet::Similarity` modules gave high-similarity to antonyms [5]. We explicitly check whether a matching involved antonyms and reject unless one of the vertices had a negation modifier.
- **Numeric Mismatch:** Since numeric expressions typically have the same part-of-speech tag (CD), they were typically matched when exact matches could not be found. However, mismatching numerical tokens usually indicated that H was not entailed, and so pairs with a numerical mismatch we rejected.

[4] All are actual, or slightly altered, RTE examples

[5] Which isn't necessarily incorrect, but simply not suitable for textual inference

Method	Accuracy	CWS
Random	50.0%	0.500
Bag-Of-Words	49.5%	0.548
TF-IDF	51.8%	0.560
GM-General	56.8%	0.614
GM-ByTask	56.8%	0.621

Table 2: Accuracy and confidence weighted score (CWS) for test set using various techniques.

7 Experiments and Results

For our experiments we used the devolpement and test sets from the Recognizing Textual Entailment challenge (Dagan et al., 2005). We give results for our system as well as for the following systems:

- Bag-Of-Words: We tokenize the text and hypothesis and strip the functional words, and stem the resulting words. The cost is given by the fraction of hypothesis not matched in the text.
- TF-IDF: Similar to Bag-Of-Words except that there is a tf-idf weight associated with each hypothesis word so that more "important" words are higher weight for matching.

We also present results for two graph matching (GM) systems. The GM-General system fits a single global threshold from the devolement set and the GM-ByTask system fits a different threshold for each of the "task".

Our results are summarized in Table 6 [6] As the result indicates, the task is particularly hard; all RTE participants scored between 50% and 60% (Dagan et al., 2005). Both GM systems perform better than either Bag-Of-Words or TF-IDF according to both raw accuracy and CWS.

We also present results on a per-task basis in Table 7. Interestingly, there is a large variation in performance depending on the task, suggesting the entailment task may be inherently more difficult than others.

[6] CWS (confidence weighted score) represents the average precision among our most confident predictions. If $\{c_1, \ldots, c_n\}$ are our confidence outputs then CWS $= \sum_{i=1}^{n} \frac{1}{n}$(number of correct predications in $c_1, \ldots, c_i$)

Task	`General`		`ByTask`	
	Accuracy	CWS	Accuracy	CWS
CD	72.0%	0.742	76.0%	0.7714
IE	55.9%	0.583	55.8%	0.595
IR	52.2%	0.5644	51.1%	0.572
MT	50.0%	0.497	43.33%	0.489
PP	58.0%	0.741	58.0%	0.746
QA	53.8%	0.537	55.4%	0.556
RC	52.1%	0.539	52.9%	0.523

Table 4: Accuracy and confidence weighted score (CWS) split by task on the RTE test set.

8 Conclusion

We have presented a graph matching based approach to determining semantic entailment.

References

Timothy Chklovski and Patrick Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *EMNLP*.

Michael Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis. Supervisor-Mitchell P. Marcus.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognizing textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop Recognizing Textual Entailment*.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

S. M. Harabagiu, Pasca M. A., and S.J. Mariorano. 2000. Experiments with open-domain textual question answering. In *COLING*, pages 292–298.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*, pages 423–430.

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules from text. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328, New York, NY, USA. ACM Press.

Dan I. Moldovan, Christine Clark, Sanda M. Harabagiu, and Steven J. Maiorano. 2003. Cogex: A logic prover for question answering. In *HLT-NAACL*.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2001. Bleu: a method for automatic evaluation of machine translation.

Text	Hypothesis	True Answer	Our answer	Conf	Comments
A Filipino hostage in Iraq was released.	A Filipino hostage was freed in Iraq.	True	True	0.84	Verb rewrite is handled. Phrasal ordering does not affect cost.
The government announced last week that it plans to raise oil prices.	Oil prices drop.	False	False	0.95	High cost given for substituting word for its antonym.
Shrek 2 rang up $92 million.	Shrek 2 earned $92 million.	True	False	0.59	Collocation "rang up" is not known to be similar to "earned".
Sonia Gandhi can be defeated in the next elections in India by BJP.	Sonia Gandhi is defeated by BJP.	False	True	0.77	"can be" does not indicate the complement event occurs.
Fighters loyal to Moqtada al-Sadr shot down a U.S. helicopter Thursday in the holy city of Najaf.	Fighters loyal to Moqtada al-Sadr shot down Najaf.	False	True	0.67	Should recognize non-Location cannot be substituted for Location.
C and D Technologies announced that it has closed the acquisition of Datel, Inc.	Datel Acquired C and D technologies.	False	True	0.64	Failed to penalize switch in semantic role structure enough

Table 3: Analysis of results on some RTE examples along with out guesses and confidence probabilities

Ted Pedersen, Siddharth Parwardhan, and Jason Michelizzi. 2004. Wordnet::similarity - measuring the relatedness of concepts. In *AAAI*.

V. Punyakanok, D. Roth, and W. Yih. 2004. Natural language inference via dependency tree mapping: An application to question answering. In *Computational Linguistics*.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453.

Rion Snow, Dan Jurafsky, and Andrew Y. Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*.

Author Suppressed. 2005. Title suppressed. In *Proceedings of the First PASCAL Challenges Workshop*. Southampton, UK.

Kristina Toutanova, Aria Haghighi, and Cristiopher Manning. 2005. Joint learning improves semantic role labeling. In *Association of Computational Linguistics (ACL)*.