Computational Models for Incongruity Detection in Humour

Rada Mihalcea^{1,3}, Carlo Strapparava², and Stephen Pulman³

¹ Computer Science Department, University of North Texas
rada@cs.unt.edu

² FBK-IRST
strappa@fbk.eu

³ Computational Linguistics Group, Oxford University
sgp@clg.ox.ac.uk

Abstract. Incongruity resolution is one of the most widely accepted theories of humour, suggesting that humour is due to the mixing of two disparate interpretation frames in one statement. In this paper, we explore several computational models for incongruity resolution. We introduce a new data set, consisting of a series of ‘set-ups’ (preparations for a punch line), each of them followed by four possible coherent continuations out of which only one has a comic effect. Using this data set, we redefine the task as the automatic identification of the humorous punch line among all the plausible endings. We explore several measures of semantic relatedness, along with a number of joke-specific features, and try to understand their appropriateness as computational models for incongruity detection.

1 Introduction

Humour is one of the most interesting and puzzling aspects of human behaviour, and it is rightfully believed to play an important role in an individual’s development, as well as in interpersonal communication. Research on this topic has received a significant amount of attention from fields as diverse as linguistics, philosophy, psychology and sociology, and recent years have also seen attempts to build computational models for humour generation and recognition.

One of the most widely accepted theories of humour is the incongruity theory, which suggests that humour is due to the mixing of two disparate interpretation frames in one statement. One of the earliest references to an incongruity theory of humour is due to Aristotle [1] who found that the contrast between expectation and actual outcome is often a source of humour. The theory also found a supporter in Schopenhauer [20], who emphasized the element of surprise by suggesting that “the greater and more unexpected [...] the incongruity is, the more violent will be [the] laughter.”

In more recent work in the field of linguistics, the incongruity theory has been formalized as a necessary condition for humour and used as a basis for the Semantic Script-based Theory of Humour (SSTH) [16] and the General Theory of Verbal Humour (GTVH) [2].

The incongruity theory (also referred to as “incongruity resolution” theory) is a theory of comprehension. When a joke narration evolves, some “latent” terms are gradually introduced, which set the joke itself against a rigid and selective linear train of thought. In this way, a short circuit occurs: the available information does not become distorted in its content, but the starting point of the initial sequence suddenly changes. Because of these latent terms, the humorous input advances on two or more interpretation paths, consisting usually of a principal path of semantic integration that the listener is more aware of, and a secondary one, which is weak and latent but existent. This latter path gains more importance as elements are added to the current interpretation of the reader, and eventually ends up forming the punch line of the joke.

For instance, the following example (taken from [18]) illustrates this theory: “Why do birds fly south in winter? It’s too far to walk.” The first part of the joke (the set-up) has two possible interpretations, due to two possible foci of the question: “Why do birds go south?” (focus on “south”) versus “Why do birds fly, when traveling south?” (focus on “fly”). The first interpretation is more obvious, also due to the phrase “in winter” which emphasizes this interpretation), and thus initially preferred. However, the punch line “it’s too far to walk” changes the preference to the second interpretation, which is surprising and generates the humorous effect.

The goal of this paper is to develop and evaluate computational models for the identification of incongruity in humour. To this end, we build a data set consisting of short jokes (one-liners), each of them consisting of a set-up, followed by several possible coherent continuations out of which only one has a comic effect. The incongruity detection task is thus translated into the problem of automatically identifying the punch line among all the possible alternative interpretations. The task is challenging because all the continuations express some coherence with the set-up. We explore several measures of semantic relatedness, along with other joke-specific features, and try to understand their appropriateness as models of incongruity detection.

The paper is organized as follows: Section 2 introduces the data set we used in the experiments. In Section 3 we explore the identification of incongruity looking at two classes of models: models based on semantic relatedness (including knowledge-based and corpus-based metrics), and models based on joke-specific features. In Section 4 we report and discuss the results, and conclude the paper with final remarks.

2 Data

To evaluate the models of incongruity in humour, we construct a data set consisting of 150 set-ups, each of them followed by four possible continuations out of which only one had a comic effect. The task is therefore cast as an incongruity resolution task, and the accuracy of the models is defined as their ability to identify the humorous continuation among the four provided.

The data set was created in four steps. First, 150 one-liners were randomly selected from the humorous data set used in [13]. A one-liner is a short sentence with comic effects and an interesting linguistic structure: simple syntax, deliberate use of rhetoric devices (e.g. alliteration, rhyme), and frequent use of creative language constructions meant to attract the reader’s attention. While longer jokes can have a relatively complex

narrative structure, a one-liner must produce the humorous effect “in one shot,” with very few words. These characteristics make this type of humour particularly suitable for use in an automatic learning setting, as the humor-producing features are guaranteed to be present in the first (and only) sentence.

Each one-liner was then manually split into a set-up and a punch line. While there are several possible alternatives for doing this split, we tried to do it in a way that would result in a minimum-length punch line. The reason for this decision is the fact that we wanted to minimize the differences among the four alternative endings by keeping them short, thus making the task more difficult (and more realistic).

Next, we provided the set-up to 10 human annotators and asked them to complete the sentence. The annotators were required to write the continuations so that the sentences make sense and are complete. We also provided a range for the number of words to be used, which was determined as a function of the number of words in the punch line. Again, the reason for providing this range was to maximize the similarity between the punch line and the other continuations.

Table 1. Sample joke set-ups, with comic (a) and serious (b, c, d) continuations

Don't drink and drive. You might hit a bump and
a) spill your drink.
b) get a flat tire.
c) have an accident.
d) hit your head.
I took an IQ test and the results
a) were negative.
b) were average.
c) confused me.
d) said I'm dumb.
I couldn't repair your brakes, so I made
a) your horn louder.
b) a phone call.
c) a special stopping device.
d) some new ones.

Finally, the continuations were manually filtered, and three alternative continuations were kept for each one-liner. The filtering was done to make sure that the alternatives had no grammatical or spelling errors, were coherent, and did not have a comic effect. Table 1 shows three entries from our data set, each entry including one punch line (a) and three alternative continuations (b, c, d).

3 Models for Incongruity Detection

Humour recognition is a difficult task. In fact, the identification of incongruity in humour has to satisfy two apparently opposite requirements: jokes have to be coherent (and thus the requirement for coherence between the set-up and the punch line), but at

the same time they have to produce a surprising effect (and thus the requirement of an unexpected punch line interpretation based on the set-up).

In our experiments, since we assume that jokes already satisfy the first requirement (jokes are coherent since they are written by people), we emphasize the second requirement and try to find models able to identify the surprising effect generated by the punch line.

Specifically, we look at two classes of models: (1) models based on semantic relatedness, including knowledge-based metrics, corpus-based metrics and domain fitness, where we seek to minimize the relatedness between the set-up and the punch line; and (2) models based on joke-specific features, including polysemy and latent semantic analysis trained on joke data, where we seek to maximize the connection between the set-up and the punch line.

3.1 Knowledge-Based Semantic Relatedness

We use several knowledge-based metrics to measure the relatedness between the set-up and each candidate punch line. The intuition is that the correct punch line, which generates the surprise, will have a minimum relatedness with respect to the set-up.

Given a metric for word-to-word relatedness, similar to [12], we define the semantic relatedness of two text segments T_1 and T_2 using a metric that combines the semantic relatedness of each text segment in turn with respect to the other text segment. First, for each word w in the segment T_1 we try to identify the word in the segment T_2 that has the highest semantic relatedness, according to one of the word-to-word measures described below. Next, the same process is applied to determine the most similar word in T_1 starting with words in T_2 . The word similarities are then weighted, summed up, and normalized with the length of each text segment. Finally the resulting relatedness scores are combined using a simple average.

There are a number of measures that were developed to quantify the degree to which two words are semantically related using information drawn from semantic networks – see e.g. [4] for an overview. We present below several measures found to work well on the WordNet hierarchy. All these measures assume as input a pair of concepts, and return a value indicating their semantic relatedness. The six measures below were selected based on their observed performance in other language processing applications, and for their relatively high computational efficiency.¹

The **Leacock & Chodorow** [8] similarity is determined as:

$$Sim_{lch} = -\log \frac{length}{2 * D} \quad (1)$$

where *length* is the length of the shortest path between two concepts using node-counting, and D is the maximum depth of the taxonomy.

The **Lesk** similarity of two concepts is defined as a function of the overlap between the corresponding definitions, as provided by a dictionary. It is based on an algorithm proposed by Lesk [9] as a solution for word sense disambiguation. The application of

¹ We use the WordNet-based implementation of these metrics, as available in the WordNet::Similarity package [15].

the Lesk similarity measure is not limited to semantic networks, and it can be used in conjunction with any dictionary that provides word definitions.

The **Wu & Palmer** [23] similarity metric measures the depth of two given concepts in the WordNet taxonomy, and the depth of the least common subsumer (LCS), and combines these figures into a similarity score:

$$Sim_{wup} = \frac{2 * depth(LCS)}{depth(concept_1) + depth(concept_2)} \quad (2)$$

The measure introduced by **Resnik** [17] returns the information content (IC) of the LCS of two concepts:

$$Sim_{res} = IC(LCS) \quad (3)$$

where IC is defined as:

$$IC(c) = -\log P(c) \quad (4)$$

and $P(c)$ is the probability of encountering an instance of concept c in a large corpus.

The next measure we use in our experiments is the metric introduced by **Lin** [10], which builds on Resnik's measure of similarity, and adds a normalization factor consisting of the information content of the two input concepts:

$$Sim_{lin} = \frac{2 * IC(LCS)}{IC(concept_1) + IC(concept_2)} \quad (5)$$

Finally, the last similarity metric considered is **Jiang & Conrath** [6]:

$$Sim_{jnc} = \frac{1}{IC(concept_1) + IC(concept_2) - 2 * IC(LCS)} \quad (6)$$

Note that all the word similarity measures are normalized so that they fall within a 0–1 range. The normalization is done by dividing the similarity score provided by a given measure with the maximum possible score for that measure.

3.2 Corpus-Based Semantic Relatedness

Corpus-based measures of semantic similarity try to identify the degree of relatedness of words using information exclusively derived from large corpora. In the experiments reported here, we considered two metrics, namely: (1) pointwise mutual information [22], and (2) latent semantic analysis [7].

The simplest corpus-based measure of relatedness is based on the **vector space model** [19], which uses a *tf.idf* weighting scheme and a cosine similarity to measure the relatedness of two text segments.

The **pointwise mutual information** using data collected by information retrieval (**PMI**) was suggested by [22] as an unsupervised measure for the evaluation of the semantic similarity of words. It is based on word co-occurrence using counts collected over very large corpora (e.g. the Web). Given two words w_1 and w_2 , their PMI is measured as:

$$PMI(w_1, w_2) = \log_2 \frac{p(w_1 \& w_2)}{p(w_1) * p(w_2)} \quad (7)$$

which indicates the degree of statistical dependence between w_1 and w_2 , and can be used as a measure of the semantic similarity of w_1 and w_2 . From the four different types of queries suggested by Turney [22], we are using the *AND* query. Specifically, the following query is used to collect counts from the AltaVista search engine.

$$p_{AND}(w_1 \& w_2) \simeq \frac{hits(w_1 \text{ AND } w_2)}{WebSize} \quad (8)$$

With $p(w_i)$ approximated as $hits(w_i)/WebSize$, the following PMI measure is obtained:²

$$\log_2 \frac{hits(w_1 \text{ AND } w_2) * WebSize}{hits(w_1) * hits(w_2)} \quad (9)$$

Another corpus-based measure of semantic similarity is the **latent semantic analysis (LSA)** proposed by Landauer [7]. In LSA, term co-occurrences in a corpus are captured by means of a dimensionality reduction operated by a singular value decomposition (SVD) on the term-by-document matrix $\mathbf{T}$ representing the corpus.

For the experiments reported here, we run the SVD operation on two different corpora. One model (**LSA on BNC**) is trained on the British National Corpus (BNC) – a balanced corpus covering different styles, genres and domains. A second model (**LSA on jokes**) is trained on a corpus of 16,000 one-liner jokes, which was automatically mined from the Web [13].

SVD is a well-known operation in linear algebra, which can be applied to any rectangular matrix in order to find correlations among its rows and columns. In our case, SVD decomposes the term-by-document matrix $\mathbf{T}$ into three matrices $\mathbf{T} = \mathbf{U}\mathbf{\Sigma}_k\mathbf{V}^T$ where $\mathbf{\Sigma}_k$ is the diagonal $k \times k$ matrix containing the k singular values of $\mathbf{T}$, $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k$, and $\mathbf{U}$ and $\mathbf{V}$ are column-orthogonal matrices. When the three matrices are multiplied together the original term-by-document matrix is re-composed. Typically we can choose $k' \ll k$ obtaining the approximation $\mathbf{T} \simeq \mathbf{U}\mathbf{\Sigma}_{k'}\mathbf{V}^T$.

LSA can be viewed as a way to overcome some of the drawbacks of the standard vector space model (sparseness and high dimensionality). In fact, the LSA similarity is computed in a lower dimensional space, in which second-order relations among terms and texts are exploited. The similarity in the resulting vector space is then measured with the standard cosine similarity. Note also that LSA yields a vector space model that allows for a *homogeneous* representation (and hence comparison) of words, word sets, and texts.

The application of the LSA word similarity measure to text semantic relatedness is done using the *pseudo-document* text representation for LSA computation, as described by Berry [3]. In practice, each text segment is represented in the LSA space by summing up the normalized LSA vectors of all the constituent words, using also a *tf.idf* weighting scheme.

3.3 Domain Fitness

It is well-known that semantic domains (such as MEDICINE, ARCHITECTURE and SPORTS) provide an effective way to establish semantic relations among word senses. This domain relatedness (or lack thereof) was successfully used in the past for word

² We approximate the value of *WebSize* to 5×10^8 .

sense disambiguation [5,11] and also for the generation of jokes [21]. We thus conduct experiments to check whether domain similarity and/or opposition can constitute a feature to discriminate the humorous punch line.

As a resource, we exploit WORDNET DOMAINS, an extension developed at FBK-IRST starting with the English WORDNET. In WORDNET DOMAINS, synsets are annotated with subject field codes (or domain labels), e.g. MEDICINE, RELIGION, LITERATURE. WORDNET DOMAINS organizes about 250 domain labels in a hierarchy, exploiting Dewey Decimal Classification. Following [11], we consider an intermediate level of the domain hierarchy, consisting of 42 disjoint labels (i.e. we use SPORT instead of VOLLEY or BASKETBALL, which are subsumed by SPORT). This set allows for a good level of abstraction without losing relevant information.

In our experiments, we extract the domains from the set-up and the continuations in the following way. First, for each word we consider the domain of the most frequent sense. Then, considering the LSA space acquired from the BNC, we build the pseudo document representations of the domains from the set-up and the continuations respectively. Finally, we measure the domain (dis)similarity among the set-up and the candidate punch lines by using a cosine similarity applied on the pseudo document representations.

3.4 Other Features

Polysemy. The incongruity resolution theory suggests that humour exploits the interference of many different interpretation paths, for example by keeping alive multiple readings or double senses. Thus, we run a simple experiment where we check the mean polysemy among all the possible punch lines. In particular, given a set-up, from all the candidate continuations we choose the one that has the higher ambiguity.

Alliteration. Previous work in automatic humour recognition has shown that structural and phonetic properties of jokes constitute an important feature, especially in one-liners [14]. Moreover, linguistic theories of humour based on incongruity resolution, such as [2,16], account for the importance of meaning-to-sound theories of how sentences are being formed. Although alliteration is mainly a stylistic feature, it also has the effect of inducing expectation, and thus it can prepare and enforce incongruity effects.

To extract this feature, we identify and count the number of alliteration/rhyme chains in each example in our data set. The chains are automatically extracted using an index created on top of the CMU pronunciation dictionary.³ The underlying algorithm is basically a matching device that tries to find the largest and longest string matching chains using the transcriptions obtained from the pronunciation dictionary. The algorithm avoids matching non-interesting chains such as e.g. series of definite/indefinite articles, by using a stopword list of functional words that cannot be part of an alliteration chain.

We conduct experiments checking for the presence of alliteration in our data set. Specifically, we select as humorous the continuations that maximize the alliteration chains linking the punch line with the set-up.

³ Available at <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>

4 Results

Table 2 shows the results of the experiments. For each model, we measure the precision, recall and F-measure for the identification of the punch line, as well as the overall accuracy for the correct labeling of the four continuations (as punch line or neutral). The performance of the models is compared against a simple baseline that identifies a punch line through random selection.

When using the knowledge-based measures, even if the F-measure exceeds the random-choice baseline, the overall performance is rather low. This suggests that the typical relatedness measures based on the WordNet hierarchy, although effective for the detection of related words, are not very successful for the identification of incongruous concepts. A possible explanation of this low performance is the fact that knowledge-based semantic relatedness also captures coherence, which contradicts the requirement for a low semantic relatedness as needed by a surprising punch line effect. In other words, the measures are probably misled by the high coherence between the set-up and the punch line, and thereby fail to identify their low relatedness.

A similar behaviour is observed for the corpus-based measures: the F-measure is higher than the baseline (with the exception of the LSA model trained on BNC), but the overall accuracy is low. Somehow surprising is the fact that contrary to the observations made in previous work, where LSA was found to significantly improve over the vector

Table 2. Precision, recall, F-measure and accuracy for finding the correct punch line

Model	Precision	Recall	F-measure	Accuracy
SEMANTIC RELATEDNESS				
Knowledge-based measures				
Leacock & Chodorow	0.28	0.34	0.31	0.61
Lesk	0.24	0.36	0.29	0.56
Resnik	0.24	0.35	0.28	0.56
Wu & Palmer	0.28	0.34	0.31	0.62
Lin	0.25	0.34	0.29	0.58
Jiang & Conrath	0.25	0.31	0.27	0.59
Corpus-based measures				
PMI	0.27	0.29	0.28	0.63
Vector space	0.26	0.61	0.37	0.48
LSA on BNC	0.20	0.25	0.22	0.56
Domain fitness				
Domain fitness	0.28	0.37	0.32	0.60
JOKE-SPECIFIC FEATURES				
Polysemy	0.32	0.33	0.32	0.66
Alliteration	0.29	0.75	0.42	0.48
LSA on joke corpus	0.75	0.75	0.75	0.87
COMBINED MODEL				
SVM	0.84	0.50	0.63	0.85
BASELINE				
Random choice	0.25	0.25	0.25	0.62

space model, here the opposite holds, with a much higher F-measure obtained using a simple measure of vector space similarity.

The models that perform best are those that rely on joke-specific features. The best results are obtained with the LSA model trained on the corpus of jokes, which exceeds by a large margin the baseline as well as the other models. This is perhaps due to the fact that this LSA model captures the “surprise” word associations that are frequently encountered in jokes.

The other joke-specific features also perform well. The simple verification of the amount of polysemy in a candidate punch line leads to a noticeable improvement above the random baseline, which confirms the hypothesis that humour is often relying on a large number of possible interpretations, corresponding to an increased word polysemy. The alliteration feature leads to a high recall, even if at the cost of low precision.

Finally, in line with the results obtained using the semantic relatedness of the set-up and the punch line, the fitness of domains is also resulting in an F-measure higher than the baseline, but a low overall accuracy.

Overall, perhaps not surprisingly, the highest precision is due to a combined model consisting of an SVM learning system trained on a combination of knowledge-based, corpus-based, and joke-specific features. During a ten-fold cross-validation run, the combined system leads to a precision of 84%, which is higher than the precision of any individual system, thus demonstrating the synergistic effect of the feature combination.

5 Conclusions

In this paper, we proposed and evaluated several computational models for incongruity detection in humour.

The paper made two important contributions. First, we introduced a new data set consisting of joke set-ups followed by several possible coherent continuations out of which only one had a comic effect. The data set helped us map the incongruity detection problem into a computational framework, and define the task as the automatic identification of the punch line among all the possible alternative interpretations. Moreover, the data set also enabled a principled evaluation of various computational models for incongruity detection.

Second, we explored and evaluated several measures of semantic relatedness, including knowledge-based and corpus-based measures, as well as other joke-specific features. The experiments suggested that the best results are obtained with models that rely on joke-specific features, and in particular with an LSA model trained on a corpus of jokes. Additionally, although the individual semantic relatedness measures brought only small improvements over a random-choice baseline, when combined with the joke-specific features, they lead to a model that has the highest overall precision of 84%, several order of magnitude better than the random baseline of 25%.

Acknowledgments

Rada Mihalcea’s work was partially supported by the National Science Foundation under award #0917170. Carlo Strapparava was partially supported by the MUR

FIRB-project number RBIN045PXH. Stephen Pulman's work was partially supported by the Companions project (<http://www.companions-project.org>) sponsored by the European Commission as part of the Information Society Technologies programme under EC grant number IST-FP6-034434. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

References

1. Aristotle. *Rhetoric*. 350 BC
2. Attardo, S., Raskin, V.: Script theory revis(it)ed: Joke similarity and joke representation model. *Humor: International Journal of Humor Research* 4(3-4) (1991)
3. Berry, M.: Large-scale sparse singular value computations. *International Journal of Super-computer Applications* 6(1) (1992)
4. Budanitsky, A., Hirst, G.: Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In: *Proceedings of the NAACL Workshop on WordNet and Other Lexical Resources*, Pittsburgh (2001)
5. Buitelaar, P., Magnini, B., Strapparava, C., Vossen, P.: Domain specific sense disambiguation. In: Edmonds, P., Agirre, E. (eds.) *Word Sense Disambiguation: Algorithms, Applications, and Trends*. Text, Speech and Language Technology, vol. 33, pp. 277–301. Springer, Heidelberg (2006)
6. Jiang, J., Conrath, D.: Semantic similarity based on corpus statistics and lexical taxonomy. In: *Proceedings of the International Conference on Research in Computational Linguistics*, Taiwan (1997)
7. Landauer, T.K., Foltz, P., Laham, D.: Introduction to latent semantic analysis. *Discourse Processes* 25 (1998)
8. Leacock, C., Chodorow, M.: Combining local context and WordNet sense similarity for word sense identification. In: *WordNet, An Electronic Lexical Database*. The MIT Press, Cambridge (1998)
9. Lesk, M.: Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In: *Proceedings of the SIGDOC Conference 1986*, Toronto (June 1986)
10. Lin, D.: An information-theoretic definition of similarity. In: *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI (1998)
11. Magnini, B., Strapparava, C., Pezzulo, G., Gliozzo, A.: The role of domain information in word sense disambiguation. *Natural Language Engineering* 8(4), 359–373 (2002)
12. Mihalcea, R., Corley, C., Strapparava, C.: Corpus-based and knowledge-based approaches to text semantic similarity. In: *Proceedings of the American Association for Artificial Intelligence*, Boston, MA, pp. 775–780 (2006)
13. Mihalcea, R., Strapparava, C.: Making computers laugh: Investigations in automatic humor recognition. In: *Proceedings of the Human Language Technology / Empirical Methods in Natural Language Processing conference*, Vancouver (2005)
14. Mihalcea, R., Strapparava, C.: Learning to laugh (automatically): Computational models for humor recognition. *Computational Intelligence* 22(2), 126–142 (2006)
15. Patwardhan, S., Banerjee, S., Pedersen, T.: Using measures of semantic relatedness for word sense disambiguation. In: *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City (February 2003)

16. Raskin, V.: *Semantic Mechanisms of Humor*. Kluwer Academic Publications, Dordrecht (1985)
17. Resnik, P.: Using information content to evaluate semantic similarity. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada (1995)
18. Ritchie, G.: Developing the incongruity-resolution theory. In: *Proceedings of the AISB Symposium on Creative Language: Stories and Humour* (1999)
19. Salton, G., Lesk, M.: *Computer evaluation of indexing and text processing*, pp. 143–180. Prentice Hall, Inc., Englewood Cliffs (1971)
20. Schopenhauer, A.: *The World as Will and Idea*. Kessinger Publishing Company (1819)
21. Stock, O., Strapparava, C.: Getting serious about the development of computational humor. In: *Proceedings of the International Conference on Artificial Intelligence, IJCAI 2003* (2003)
22. Turney, P.: Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In: Flach, P.A., De Raedt, L. (eds.) *ECML 2001. LNCS (LNAI)*, vol. 2167, p. 491. Springer, Heidelberg (2001)
23. Wu, Z., Palmer, M.: Verb semantics and lexical selection. In: *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, New Mexico (1994)