Supervised Learning of Semantic Relatedness

Ran El-Yaniv and David Yanay

Department of Computer Science, Technion - Israel Institute of Technology
`rani@cs.technion.ac.il, dudu.yanay@gmail.com`

Abstract. We propose and study a novel supervised approach to learning statistical semantic relatedness models from subjectively annotated training examples. The proposed semantic model consists of parameterized co-occurrence statistics associated with textual units of a large background knowledge corpus. We present an efficient algorithm for learning such semantic models from a training sample of relatedness preferences. Our method is corpus independent and can essentially rely on any sufficiently large (unstructured) collection of coherent texts. Moreover, the approach facilitates the fitting of semantic models for specific users or groups of users. We present the results of extensive range of experiments from small to large scale, indicating that the proposed method is effective and competitive with the state-of-the-art.

1 Introduction

Semantic relatedness (SR) has been steadily gaining attention among statistical NLP and AI researchers. The interest in SR research has been reinforced by the emergence of applications that can greatly benefit from SR capabilities. Among these applications we mention targeted advertising [1], information retrieval and web search [2], automatic tagging and linking [3], and text categorization [4]. The importance of SR is particularly evident when attempting to categorize short texts (e.g., ads, tweets, search queries) where standard bag-of-words representation is not sufficiently effective [5].

The goal in SR is to quantify the intensity of how much two target terms are related to each other, and all relation types between these terms might be considered. These relations can be among the linguistically formal ones, which typically have a name (e.g., synonyms, hypernyms, etc.), but in general, such relations can be informal in the sense that they have not been coined, and they express some (perhaps complex) association between the two terms. For example, consider the following term pairs: (`Michael Jordan,Basketball`), (`Madonna,Pop`), and (`Marilyn Monroe,Movie`). All three pairs, (`X,Y`), are strongly related via a common relation.[1] Thus, the SR task involves all possible relations whose number is in principle unbounded. We note that in a restricted version of SR, called *semantic similarity*, one considers only synonymy relations. As argued by Budanitsky and Hirst [6] and others, SR is considered more general than semantic similarity, and in this sense it is more difficult.

In this work we consider the SR task and we aim to correctly qualify the relatedness of two given terms where the underlying relation can be linguistic or informal. However,

[1] The relation we had in mind is “X is an all times Y star”; other variations are sensible.

we do not aim at identifying or characterizing the underlying relation. Note also that in the standard SR setting we consider here (see definitions in Section 3), the terms to be evaluated for relatedness are provided without a context, unlike typical disambiguation tasks. Hence, as most existing works on SR focusing on this or equivalent setup, we do not aim at directly solving the disambiguation problem along the way.

SR is an elusive concept. While a rigorous mathematical definition of SR is currently beyond grasp, the concept is intuitively clear to everyone. Indeed, the popular conception of SR is reflected in its nebulous Wikipedia entry (at the moment of writing this) stating that: *In essence, semantic similarity, semantic distance, and semantic relatedness all mean, "How much does term A have to do with term B"?* Clearly, SR "has to do" with the understanding of meaning – a grand challenge in AI research. Can a computer program quantify the extent to which two terms share the same meaning?

The statistical NLP and AI communities have adopted a pragmatic modus operandi to these questions: even if we don't know how to define SR, we can still create computer programs that generate useful SR assesments. Indeed, a number of effective heuristic approaches to SR have been proposed, and this line of work has proven to be rewarding, see e.g., [7, 8]. In particular, it has been shown that useful SR scores can be systematically extracted from large lexical databases or electronic repositories of common-sense and domain-specific background knowledge.

With the exception of a few papers, most of the algorithms proposed for SR valuation have been following an *unsupervised* learning or *knowledge engineering* procedures. Such SR valuation functions have been generated, for the most part, using some hand-crafted formula applied to semantic information that is extracted from a (structured) background knowledge corpus. The proposed methods have employed a number of interesting techniques, some of which are discussed in Section 2.

One motivation for the present work is the realization that SR assessments are *subjective* and often *relative*, rather than objective and absolute. While we can expect some kind of consensus among people on the (relative) relatedness valuations of basic terms, the relatedness assessments of most terms depend on many subjective factors such as literacy, intelligence, context, time and location. For example, the name `Michael Jordan` is generally strongly related to `Basketball`, but some people in the machine learning community may consider it more related to `Machine Learning`. As another example, consider WordSim353 [9], a standard benchmark dataset for evaluating and comparing SR measures. This benchmark contains some controversial relative preferences between word pairs such as (`Arafat,Peace`) vs. (`Arafat,Terror`) and (`Jerusalem,Israel`) vs. (`Jerusalem, Palestinian`). Can you tell which pair is more related in each instance? Obviously, the answer must be personal/subjective.

This sensitivity of SR to subjective factors should make it very hard, if not impossible, to satisfy all SR needs using a single "universal" method. Indeed, some published SR measures outperform others in certain benchmarks tests and underperform in others. For example, Strube and Ponzetto [10] mentioned that the WordNet-based measures perform better than the Wikipedia-based measures on the Rubenstein and Goodenough benchmark, but the WordNet methods are inferior over WordSim353.

In this work we propose a novel *supervised* approach to learning SR from examples. Following Agirre at al. [11] we model SR learning as a binary classification problem where each instance encodes the relative relatedness of two term pairs. Given a labeled training set our goal is to learn an SR function capable of determining the labels of unobserved instances. We present an empirical risk minimization (ERM) algorithm that learns by inducing a weighted measure of terms co-occurrence defined over a background knowledge corpus of free-text documents. The labeled examples are used to fit this model to the training data. The resulting algorithm is relatively simple, has only few hyper-parameters, and is corpus independent. Our experiments show that the algorithm achieves notable generalization performance. This is observed over a wide range of experiments on a number of benchmarks. We examine and demonstrate the effectiveness of our algorithm using two radically different corpora: an old version of Wikipedia and the books in the Project Gutenberg.

2 Related Work

SR techniques typically rely on some kind of world or expert knowledge, which we term here *background knowledge (BK)* corpus. The BK corpus is a key element in many methods and we categorize existing SR techniques into three main families according to type and structure of their BK corpus. *Lexical* methods rely on lexical databases such as WordNet or Rodget's Thesaurus. *Wiki* methods rely on structured BK corpora like Wikipedia or the Open Directory Project (DMOZ). Finally, SR techniques that rely on unstructured text collections are referred to as *structure-free* methods. Currently, the largest publicly available SR benchmark dataset is WordSim353 [9]. Encompassing a variety of semantic relations, WordSim353 has been providing a focal point to empirical SR research in the past years. In the semantic relatedness literature it is common to evaluate relatedness ranking using the Spearman correlation. Hence, in the sequel we mention WordSim353 Spearman correlation scores in cases where they were reported.

Unsupervised methods: lexical, Wiki and structure-free. Numerous *lexical methods* utilize the WordNet database, which organizes words in *synsets* (sets of synonyms). The lexical relations among synsets are categorized into types such as synonyms, antonyms and hypernyms. Another lexical database is the well-known Roget's Thesaurus. Similarly to WordNet, Rodget's Thesaurus contains groups of terms, called *semicolon groups*. These groups are inter-linked, but the links are not lexically annotated as in WordNet. Lexical SR methods typically view the lexical database as a graph whose nodes are terms and edges are lexical relations.

Several researchers have defined SR measures combining lexical links with structure free corpora [12–14]. Others only utilize the lexical links [15–17]. Yet others [18–20] use WordNet by taking advantage of *glosses* (terms' definitions). Refer to [6, 21, 22] for various other lexical methods.

Structure in Wiki BK corpora can be manifested in various ways, and the most important ones are semantic coherency of documents and titles [8, 23], meaningful interlinks [24, 25], and hierarchical categorization [26]. Many such *Wiki methods* associate an article in Wikipedia to each term and utilize known or newly proposed measures between Wikipedia's articles. Examples include Strube and Ponzetto [10], as well as Gabrilovich and Markovitch's celebrated Explicit Semantic Analysis (*ESA*) method [8], whereby each term is represented as a sparse vector containing a non-zero component for each significant Wikipedia article in which it appears, and the SR score of two terms is defined as the cosine of their vectors. ESA achieved a correlation of 0.75 with WordSim353, and is used as a subroutine in applications [25, 23, 27]. The Temporal Semantic Analysis (*TSA*) measure proposed by Radinsky et al. [23] recently achieved 0.82 correlation for WordSim353, which is the best known *unsupervised* result on WordSim353.

Structure-free methods posit that SR is a function of the co-occurrence statistics of a pair of terms in a BK corpus. Lin [28] proposed information-based methods to define and quantify term similarity. Dagan et al. [29] and Terra and Clarke [30] experimented with various statistical co-occurrence measures for estimating SR from structure-free corpora. Deerwester et al. [31] used algebraic representation of the BK. Applying this measure, Finkelstein et al. [9] achieved 0.56 correlation with WordSim353. Reisinger and Mooney [32] obtained a correlation of 0.77 with WordSim353 by generating for a term t a feature vector for each context in which t appears.

Supervised methods. There have been a few successful attempts to utilize *supervised* learning techniques for constructing SR functions. For the most part, these works follow a similar methodology whereby the features of a learning instance are assembled from scores obtained from various unsupervised methods (such as those discussed above). Using this feature generation approach, the existing works then resorted to known inductive learning schemes, such as support vector machines (SVMs). For a partial list of results, refer to [33, 11, 27, 10]. Haralambous and Klyuev reported on 0.8654 correlation score with WirdSim353 [27], which is the best that was ever reported. The second best result is by Agirre et al. [11], achieving 0.78 correlation with WordSim353. These two works are also the closest to ours, mainly in their formulation of the learning problem. However, our solution methodology is fundamentally different.

3 Problem Setup

We consider a fixed corpus, $\mathcal{C} \triangleq \{c_1, c_2, \ldots, c_N\}$, defined to be a set of contexts. Each *context* c_i, $i = 1, \ldots, N$, is a textual unit conveying some information in free text. In this work we consider contexts that are sentences, paragraphs or whole documents. Let $D \triangleq \{t_1, t_2, \ldots, t_d\}$ be a *dictionary* consisting of all the terms appearing in the corpus. A term may be any frequent phrase (unigram, bigram, trigram, etc.) in the corpus, e.g., “book”, “New York”, “The Holly Land.” Ultimately, our goal is to automatically construct a function $f(t_1, t_2)$ that correctly ranks the relatedness of the terms $t_1, t_2 \in D$ in accordance with the subjective semantics of a given user. We emphasize that we do not require f to provide absolute scores but rather a relative value inducing a complete order over the relatedness of all terms. In reality this total order assumption doesn't hold, since the comparison between two term pairs not sharing any term might be meaningless. Furthermore, human preferences may contain cycles, perhaps due to comparisons made using different features (like in rock-paper-scissors game).

Our goal is to construct the function f using supervised learning. Specifically, the user will be presented with a training set $\{X_1, \ldots, X_m\}$ to be labeled, where each $X_i \triangleq (\{t_1^i, t_2^i\}, \{t_3^i, t_4^i\})$ is a quadruple of terms. The binary label, $y_i \in \{\pm 1\}$, of the instance X_i should be $+1$ if the terms in the first pair $\{t_1^i, t_2^i\}$ are more related to each other than the terms in the second pair $\{t_3^i, t_4^i\}$, and -1 otherwise. Each quadruple along with its label, (X_i, y_i) is also called a *preference*.

Denote by $S_m \triangleq \{(X_1, y_1), \ldots, (X_m, y_m)\}$, a set of labeled training examples received from the user. We assume that if $(X, y) \in S_m$ then $(X, -y) \notin S_m$. A binary classifier in our context is a function $h : D^4 \to \{\pm 1\}$ satisfying, for all $(\{t_1, t_2\}, \{t_3, t_4\}) \in D^4$, the "anti-symmetry" condition $h(\{t_1, t_2\}, \{t_3, t_4\}) = -h(\{t_3, t_4\}, \{t_1, t_2\})$. The 0/1 *training error* of h is, $R_m(h) \triangleq \frac{1}{m} \sum_i \mathbb{I}\{h(X_i) \neq y_i\}$. The standard underlying assumption in supervised learning is that (labeled) instances are drawn i.i.d. from some unknown distribution $P(X, Y)$ defined over $D^4 \times \{\pm 1\}$. The classifier h is chosen from some hypothesis class $\mathcal{H}$. In this work we focus on the *realizable setting* whereby labels are defined by some unknown *target hypothesis* $h^* \in \mathcal{H}$. Thus, the underlying distribution reduces to $P(X)$. The performance of a classifier h is quantified by its true or (0/1) *test error*, $R(h) \triangleq \mathbf{E}_P\{h(X) \neq Y\}$.

Why do we choose to ask the user about pairwise preferences rather than requesting an absolute relatedness score of a single pair of terms? Our choice is strongly motivated by recent work showing that answers to such questions are more accurate than answers to questions about absolute quality. In order to extract an absolute score, a user must rely on some implicit global scale, which may or may not exist. For a sample of literature justifying this general approach (both theoretically and empirically) see [34, 35].

4 Adaptive Measure

Recognizing the widely accepted idea that the SR of two terms is a function of their co-occurrence pattern in documents, we would like to somehow measure co-occurrence using a corpus of BK where such patterns are manifested. Therefore, a major component of the proposed algorithm is an appropriate co-occurrence measure. However, we also require adaptivity to specific user's subjective relatedness preferences. Our observation is that such adaptivity can be achieved by learning from examples user specific weights to be assigned to contexts. Overall, our approach is to construct a reasonable initial model, derived only from the BK corpus (without supervision), which fits a rough general consensus on relatedness of basic terms. This initial model is the starting point of a learning process that will refine the model to fit specific user preferences.

We examined various co-occurrence indices, such as Jaccard measure, pointwise mutual information, KL- and Jensen-Shannon divergences, and latent semantic analysis. Based on this study and some other results [29, 30, 36], we selected the normalized semantic distance measure of Cilibrasi and Vitanyi [37].[2] Indeed, NSD by itself can achieve a high 0.745 Spearman correlation with WordSim353 (via our implementation using Wikipedia as the BK corpus) thus providing a very effective starting point. We note that information measures are also effective, but not quite as good.[3] We also find it appealing that this measure was derived from solid algorithmic complexity principles.

[2] Note that Cilibrasi and Vitanyi termed this function "Google similarity distance" and applied it by relying on Google to retrieve proxies for co-occurrence statistics. In our discussion co-occurrence statistics can be obtained in any desirable manner.

[3] Pointwise mutual information achieved correlation of 0.73 with WordSim353[36].

Cilibrasi and Vitanyi defined the *semantics* $S(t_1, \ldots, t_n)$ of the terms $t_1, \ldots, t_n$, as the set of all contexts in which they appear together. Than they defined the *normalized semantic distance (*NSD*)* between t_1, t_2 to be

$$\mathsf{NSD}\left(t_1, t_2\right) \triangleq \frac{\max\left\{\log\left(\left|S\left(t_1\right)\right|\right), \log\left(\left|S\left(t_2\right)\right|\right)\right\} - \log\left(\left|S\left(t_1, t_2\right)\right|\right)}{\log\left(Z\right) - \min\left\{\log\left(\left|S\left(t_1\right)\right|\right), \log\left(\left|S\left(t_2\right)\right|\right)\right\}},$$

where $Z \triangleq \sum_{t_1,t_2 \in D} |S(t_1, t_2)|$. The NSD function, like any other absolute scoring function for pairs, induces a permutation over all the term pairs, and therefore, can be utilized as a classifier for SR preferences, as required. However, this classifier is constructed blindly without any consideration of the user's subjective preferences. To incorporate user subjective preferences we introduce a novel extension of NSD that allows for assigning weights to contexts. Define the *weighted semantics* $WS(t_1, \ldots, t_n)$ of the terms $t_1, \ldots, t_n$ as $WS(t_1, \ldots, t_n) \triangleq \sum_{c \in S(t_1,\ldots,t_n)} w(c)$, where $w(c) \in \mathbb{R}^+$ is a weight assigned to the context c, where we impose the normalization constraint $\sum_{c \in \mathcal{C}} w(c) = |\mathcal{C}| = N$. Thus, given a BK corpus, $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$, and a set W of weights, $W \triangleq \{w(c_1), w(c_2), \ldots, w(c_N)\}$, we define *weighted normalized semantic distance (*WNSD*)* between t_1 and t_2 is,

$$\mathsf{WNSD}_W(t_1, t_2) \triangleq \frac{\max\{\log(WS(t_1)), \log(WS(t_2))\} - \log(WS(t_1, t_2))}{\log(Z) - \min\{\log(WS(t_1)), \log(WS(t_2))\}},$$

where Z is a normalization constant, $Z \triangleq \sum_{t_1,t_2 \in D} WS(t_1, t_2)$. We call the set W of weights a *semantic model*. Our goal is to learn a good model from labeled examples.

Recall that our objective is to quantify the relatedness of two terms regardless of the types of relations that link these terms. Is it really possible to learn a single model W that will encode coherent semantics universally for all terms and all relations?

At the outset, this objective might appear hard or even impossible to achieve. Additional special obstacle is the modeling of synonym relations. The common wisdom is that synonym terms, which exhibit a very high degree of relatedness, are unlikely to occur in the same context (see e.g., [28, 6]), especially if the context unit is very small (e.g., a sentence). Can our model capture also similarity relations? We empirically investigate this issue and answer this question in the affirmative.

5 The SemanticSort Algorithm

Let $f_W : D \times D \to \mathbb{R}^+$ be any adaptive co-occurrence measure satisfying: (i) each context has an associated weight in W; (ii) $f_W(t_1, t_2)$ monotonically increases with increasing weight(s) of context(s) in $S(t_1, t_2)$; and (iii) $f_W(t_1, t_2)$ monotonically decreases with (increasing) weight(s) of context(s) in $S(t_1) \setminus S(t_1, t_2)$ or $S(t_2) \setminus S(t_1, t_2)$.

We now present a learning algorithm that can utilize any such function. We later apply this algorithm while instantiating this function to WNSD, which clearly satisfies the required properties. Note, however, that many known co-occurrence measures can be extended (to include weights) and be applied as well.

Relying on f_W we would like utilize empirical risk minimization (ERM) to learn an appropriate model W of context weights so as to be consistent with the training set S_m.

Algorithm 1 $\mathrm{SemanticSort}(S_m, \alpha, \alpha_{max}, \epsilon, \lambda)$

```
1: Initialize:
2: $W \leftarrow \overrightarrow{1}, \quad \Delta_{prev} \leftarrow MaxDoubleValue$
3: repeat
4:    $\Delta \leftarrow 0$
5:    For all $e = ((\{t_1, t_2\}, \{t_3, t_4\}), y) \in S_m$ do
6:       If $(y == -1)$ then
7:          $(\{t_1, t_2\}, \{t_3, t_4\}) \leftarrow (\{t_3, t_4\}, \{t_1, t_2\})$
8:       $score_{12} \leftarrow f_W(t_1, t_2) \quad score_{34} \leftarrow f_W(t_3, t_4)$
9:       If $(score_{12} < score_{34})$ then
10:         {This is an unsatisfied example.}
11:         $\lambda_{up} \leftarrow \frac{\alpha \cdot \lambda(\Delta_e)+1}{\alpha \cdot \lambda(\Delta_e)}, \quad \lambda_{dn} \leftarrow \frac{1}{\lambda_{up}}$
12:         $\Delta \leftarrow \Delta + \Delta_e$
13:         for all $c \in S(t_1, t_2)$ do $w(c) \leftarrow w(c) \cdot \lambda_{up}$
14:         for all $c \in S(t_3, t_4)$ do $w(c) \leftarrow w(c) \cdot \lambda_{dn}$
15:         Normalize weights s.t. $\sum_{c \in C} w(c) = |\mathcal{C}|$
16:    If $(\Delta - \Delta_{prev} + \epsilon \geq 0)$ then
17:       $\alpha \leftarrow 2 \cdot \alpha$
18:       If $(\alpha \geq \alpha_{max})$ then Return
19:    $\Delta_{prev} \leftarrow \Delta$
20: until $\Delta == 0$
```

To this end we designed SemanticSort, an algorithm that minimizes the training error over S_m by fitting appropriate weights to f_W. A pseudocode is provided in Algorithm 1.

The inputs to SemanticSort are S_m, a learning rate factor α, a learning rate factor threshold α_{max}, a decrease threshold ϵ, and a learning rate function λ. When a training example is not satisfied, e.g., $e = (X = (\{t_1, t_2\}, \{t_3, t_4\}), y = +1)$ and $f_W(t_1, t_2) < f_W(t_3, t_4)$, we would like to increase the semantic relatedness score of t_1 and t_2 and decrease the semantic relatedness score of t_3 and t_4. SemanticSort achieves this by multiplicatively promoting/demoting the weights of the "good"/"bad" contexts in which t_1, t_2 and t_3, t_4 co-occur. The weight increase (resp., decrease) depends on λ_{up} (resp., λ_{dn}), which are defined as follows. $\lambda_{up} \triangleq \frac{\alpha \cdot \lambda(\Delta_e)+1}{\alpha \cdot \lambda(\Delta_e)}$, $\lambda_{dn} \triangleq \frac{1}{\lambda_{up}}$.

SemanticSort uses λ to update context weights in accordance with the error magnitude incurred for example e, defined as $\Delta_e \triangleq |f_W(t_1, t_2) - f_W(t_3, t_4)|$. Thus, we require that λ is a monotonically decreasing function so that the greater Δ_e is, the more aggressive λ_{up} and λ_{dn} will be. The learning speed of the algorithm depends on these rates, and overly aggressive rates might prevent convergence due to oscillating semantic relatedness scores. Hence, SemanticSort gradually refines the learning rates as follows. Define $\Delta \triangleq \sum_{\text{e is not satisfied}} \Delta_e$, as the total sum of the differences over unsatisfied examples. We observe that if Δ decreases at least in ϵ in each iteration, then SemanticSort converges and the learning rates remain the same. Otherwise, SemanticSort will update the learning rate to be less aggressive by doubling α. Therefore, we require that $0 < \epsilon$. Note that the decrease of Δ is only used to control convergence, but we test SemanticSort using the 0/1 loss function as described in Section 6. SemanticSort iterates over the examples until its hypothesis satisfies all of them, or α exceeds α_{max}. Thus, empirical risk minimization in our context is achieved by minimizing Δ.

6 Empirical Evaluation

To evaluate the effectiveness of SemanticSort we conducted several experiments. One of the barriers in designing these experiments is the lack of labeled dataset of term quadruples as required by our model. The common benchmark datasets are attractive because they were labeled by human annotators, but these datasets are rather small. When considering a small real world application involving even 500 vocabulary terms, we need to be able to compare the relatedness of many of the $\binom{500}{2} = 124,750$ involved pairs. However, the largest available dataset, WordSim353, contains only 353 pairs. Although we utilized all available datasets in our experiments (see below), we sought a benchmark of significantly larger scale in order to approach real world scenarios.

Gutenberg Semantic Score (GSS). Without access to a humanly annotated dataset of a large scale, we synthesized a labeled dataset as follows. Noting that a vocabulary of 1000-2000 words covers about 72%-80% of written English texts [38], we can envision practical applications involving vocabularies of such sizes. We therefore selected a dictionary D_n consisting of the n most frequent English words ($n = 500, 1000$). For each of the $\binom{n}{2}$ term pairs over D_n we used an independent corpus of English texts, namely the Gutenberg Project, to define the SR score of pairs, using the NSD method, applied with sentence based contexts. We refer to this as the *Gutenberg Semantics Score (GSS)*.

Project Gutenberg is a growing repository that gathers many high quality and classic literature that is freely available on the web. For example, among the books one can find `Alice's Adventures in Wonderland`, `The Art of War`, `The Time Machine`, `Gulliver's Travels`, and many well known fiction ebooks. Currently, Project Gutenberg offers over 36,000 ebooks.[4]

While GSS is certainly not as reliable as human generated score (for the purpose of predicting human scores), it is positively correlated with human annotation, achieving 0.58 Spearman correlation with the WordSim353 benchmark. Given a set of term pairs together with their SR scores (such as those generated by GSS), we construct a labeled set of preferences according to SR scores (see definitions in Section 3).

We emphasize that the texts of the Project Gutenberg were taken conclusively and as is, without any modifications, to avoid any selection bias.[5] Nevertheless, despite its statistical correlation to human annotation, our main objective isn't to evaluate absolute performance scores, but rather to see if generalization can be accomplished at this scale while using an extremely small fraction of the available training examples.

Background knowledge corpora. An integral part of the SemanticSort model is its BK corpus. We conducted experiments using two corpora. The first corpus is the snapshot of Wikipedia from 05/11/05 preprocessed using Wikiprep.[6] Following [8], in order

[4] In this work we used a complete older version of Project Gutenberg from February 1999 containing only 1533 texts bundled by Walnut Creek CDROM.

[5] The GSS dataset is available at `http://www.cs.technion.ac.il/~rani/semantic_relatedness`.

[6] Wikiprep is an XML preprocessor for Wikipedia, available at `http://www.cs.technion.ac.il/~gabr/resources/code/wikiprep`.

to remove small and overly specific articles, we filtered out articles containing either less than 100 non-stopword terms and/or less than 5 incoming links and/or less than 5 outgoing links. The second corpus we used is the Project Gutenberg mentioned above. We emphasize that in all experiments involving GSS scores only Wikipedia was used as the BK corpus. Also, in each experiment we either used Wikipedia or Gutenberg as a BK corpus and not both. In all the experiments we ignored stopwords and stemmed the terms using Porter's stemmer. Finally, We considered three types of contexts: sentences, paragraphs and whole documents. Sentences are parsed using '.' as a separator without any further syntax considerations; paragraphs are parsed using an empty line as a separator. No other preprocessing, filtering or optimizations were conducted.

Evaluation methodology. Consider a collection P of preferences, where each preference is a quadruple, as define in Section 3. When we evaluate performance of the algorithm w.r.t. a training set of size m, we choose an m-subset, $S_m \subseteq P$ uniformly at random. The rest of the preferences in $P \setminus S_m$ are taken as the test set.[7] However, if $P \setminus S_m$ remains very large, only 1,000,000 preferences, chosen uniformly at random from $P \setminus S_m$, are taken for testing. The training set S_m is fed to $\mathrm{SemanticSort}$, which generates an hypothesis h, consisting of a weights vector W that includes a component for each context in $\mathcal{C}$. Then we apply the hypothesis on the test set and calculate the resulting accuracy (using the 0/1 loss function). This quantity provides a relatively accurate estimate of (one minus) the true error $R(h)$. In order to obtain a learning curve we repeat this evaluation procedure for a monotonically increasing sequence of training set sizes. The popular performance measure in SR research is the Spearman correlation coefficient of the ranking obtained by the method to the ground truth ranking. Therefore, we also calculated and reported it as well. In addition, in order to gain statistical confidence in our results, we repeated the experiments for each training set size multiple times and reported the average results. For each estimated average quantity along the leaning curve we also calculated its standard error of the mean (SEM), and depicted the resulting SEM values as error bars.

Experiment 1: large scale. In order to evaluate $\mathrm{SemanticSort}$ on ambitious, large scale and quite realistic scenario, we conducted the following experiments. Taking D_{1000} (the top 1000 most frequent words in Wikipedia) we considered all possible preferences. Note that the number of preferences associated with D_{1000} is huge, containing about $10^{12}/4$ quadruples. We labeled the preferences according to GSS as described above. In generating the learning curve we were only able to reach $m = 2,000,000$ training examples, thus utilizing an extremely small fraction of the available preferences (the largest fraction is about 10^{-5}). Figure 1 presents 0/1 test accuracy and Spearman correlation learning curves. On this figure we also mark the results obtained by two unsupervised methods: (i) NSD using Wikipedia as BK corpus with paragraph level contexts; (ii) the well known ESA method using the same filtered Wikipedia snapshot mentioned above. Both these unsupervised performance scores were calculated by us

[7] Formally speaking, this type of sampling without replacement of the training set, is within a standard transductive learning model [39, Sec. 8.1,Setting 1] .

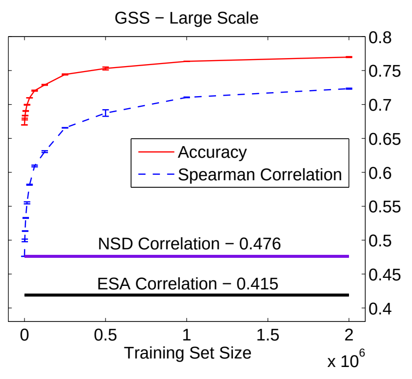

Fig. 1. Experiment 1 (large scale) - Learning curves for test accuracy (solid) and test correlation (dashed), with standard error bars.

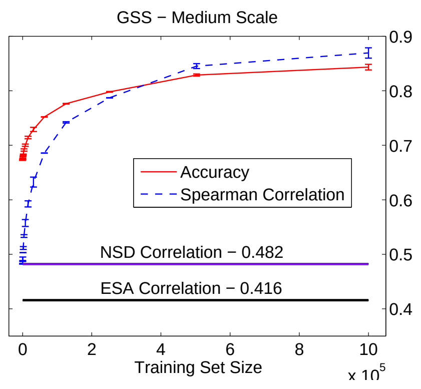

Fig. 2. Experiment 2 (medium scale) - Learning curves for test accuracy (solid) and test correlation (dashed),with standard error bars.

using our implementations of these methods. It is evident that $\mathrm{SemanticSort}$ successfully generalized the training sample and accomplished a notable improvement over its starting point. We believe that these results can serve as a proof of concept and confirm $\mathrm{SemanticSort}$'s ability to handle real world challenges.

Experiment 2: medium scale. We repeated the previous experiment, now with D_{500}, taken to as a random subset of D_{1000}. All other experiment parameters were precisely as in Experiment 1. The resulting learning curves are shown in Figure 2. Clearly, this medium scale problem gave rise to significantly higher absolute performance scores. We believe that the main reason for this improvement (over the large scale experiment) is merely the use of a larger fraction of preferences in training.

Experiment 3: small scale. As mentioned in Section 2, many of the known techniques, including the published supervised methods, evaluated performance with respect to the WordSim353 benchmark. In order to link the proposed approach to the current literature we also conducted an experiment using WordSim353 as a source for labeled preferences. This experiment serves three purposes. First, it can be viewed as a sanity check for our method, now challenging it with humanly annotated scores. Second, it is interesting to examine the performance advantage of our supervised approach vs. no systematic supervision as obtained by the unsupervised methods (we already observed in Experiments 1&2 that our supervised method can improve the scores obtained by ESA and NSD). Finally, using this experiment we are able compare between $\mathrm{SemanticSort}$ and the other known supervised methods that so far have been relying on SVMs.

Figure 3 shows the learning curves obtained by $\mathrm{SemanticSort}$ applied with paragraph contexts using either Wikipedia or Gutenberg (but not both together) as a BK corpus. The lower horizontal line, at the 0.82 level, marks the best known *unsupervised* result obtained for WordSim353 [23]. The upper horizontal line, at the 0.8654 level, marks the best known *supervised* result [27]. It is evident that quite rapid learning is

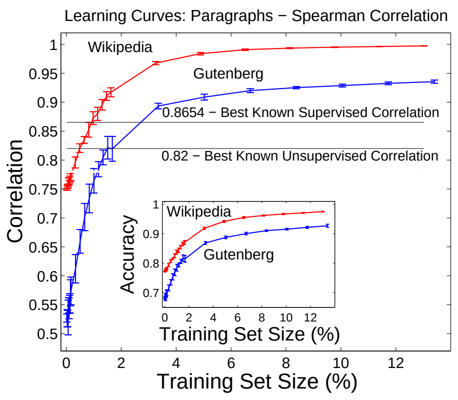

Fig. 3. Experiment 3 (small scale) - Learning curves for test correlation and test accuracy (sub-plot) with standard error bars using either Wikipedia or Gutenberg.

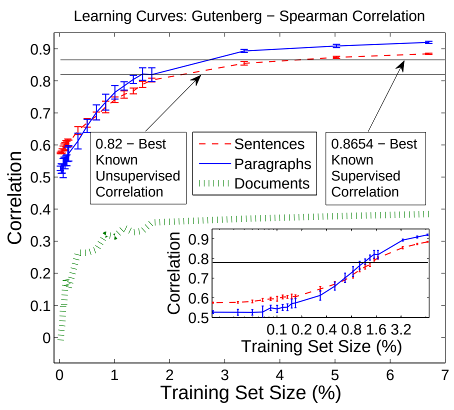

Fig. 4. Experiment 3 (small scale) - Learning curves for test correlation with standard error bars using Project Gutenberg. The sub-plot uses logarithmically scaled X-axis.

accomplished using either the Wikipedia or the Gutenberg models, but Wikipedia enables significantly faster learning and smaller sample complexity for each error level. The curves in the internal panel show the corresponding test *accuracies* (0/1 loss) for the same experiments. Note that meaningful comparisons between SemanticSort and the other (SVM based) supervised methods (described in Section 2) can only be made when considering the same train/test partition sizes. Unlike our experimental setting, both Agirre et al. [11], and Haralambous and Klyuev [27] achieved their reported results (0.78 and 0.8654 correlation with WordSim353, respectively) using 10-fold cross validation, thus utilizing 90% of the available labeled preferences for training. When considering only the best results obtained at the top of the learning curve, SemanticSort outperforms the best reported supervised performance after consuming 1.5% of all the available WordSim353 preferences using the Wikipedia model and after consuming 3% of the preferences using the Project Gutenberg model.

Figure 4 depicts three Gutenberg learning curves: one for each context type. The internal panel zooms into the same curves of sentence- and paragraph-based contexts, now with logarithmically scaled X-axis to emphasize their differences. As before, the lower (resp., upper) horizontal line at 0.82 (resp., 0.8654) marks the best known unsupervised (resp., supervised) result for WordSim353 [23] (resp., [27]). Clearly, paragraph contexts exhibit the best test performance for almost all training set sizes. In contrast, contexts consisting of whole documents perform poorly, to the extent that even after utilizing the largest training set size, they are still way behind sentences and paragraphs (even without using a single labeled example). A similar comparison (not presented) for Wikipedia contexts showed entirely different picture with all contexts exhibiting very similar (and almost indistinguishable) performance as shown for paragraphs in Figure 3.

Experiment 5: semantic similarity. Synonymous relations are considered among the most prominent semantic relations. *Semantic similarity* is a sub-domain of SR where one attempts to assess the strength of synonymous relations. A widely accepted approach to handle synonyms (and antonyms) is via distributional similarity [28,6]. In this approach, to determine the similarity of terms t_1 and t_2 we consider $D(t_1)$ and $D(t_2)$, the "typical" distributions of terms in close proximity to t_1 and t_2, respectively. It is well known that these distributions tend to resemble whenever t_1 is similar to t_2, and vice versa. In contrast, SemanticSort computes its similarity scores based on co-occurrence counts, and the conventional wisdom is that synonyms tend not to co-occur. Can we expect SemanticSort to handle synonymous relations?

We examine and analyze the behavior of a paragraph-based SemanticSort on a specialized semantic similarity task. To this end, we use the semantic similarity datasets, namely Rubenstein and Goodenough (R&G) [40] and Miller and Charles (M&C) [41].

Figure 5 depicts the results obtained for the M&C dataset. The upper (resp. lower) horizontal line, at the 0.92 (resp. 0.9) level, marks the best known *supervised* (resp. *unsupervised*) results obtained for the M&C dataset [11] (resp. [14,42]). Figure 6 depicts the results obtained for the R&G dataset. The upper (resp. lower) horizontal line, at the 0.96 (resp. 0.8614) level, marks the best known *supervised* (resp. *unsupervised*) results obtained for R&G dataset [11] (resp. [17]). The learning curves depicted in both figures clearly indicate that learning synonyms using our method is an achievable task, and in fact, can improve upon the distributional similarity methods. While synonyms and antonyms co-occur infrequently, they still do co-occur. It is a nice property of our model that it can leverage these sparse co-occurrence counts and accurately detect synonyms by sufficiently increasing the weights of their mutual contexts.

7 Model Interpretability

The model learned by SemanticSort is encoded in its weight vector W. In this section we summarize our preliminary study to explore the model W and gain some insight into its structure. Are the weights in W "arbitrarily" optimized to reduce the training error, or is it the case that they are organized in a meaningful and interpretable manner? Can we learn from W something about the human rater(s) who tagged the training set? Can something on their world knowledge and/or intellectual interests be inferred?

Trying to answer the above questions we conducted the following study. The results fall short of fully answering these questions, but they are indicative and suggest that the model W contains useful information that perhaps could be utilized in applications. In our experiments, due to the absence of human annotating resources, we again synthesized "human" raters whose knowledge is focused on specific topics.

Given a specific topic T in Wikipedia (e.g., sports) we extracted the set S_T of documents pertaining to T (using the Wikipedia topic tags), and partitioned S_T uniformly at random into two subsets, S_T^1 and S_T^2. We used S_T^1 for labeling, and S_T^2, as part of the BK corpus together with the rest of the Wikipedia corpus. Our synthetic rater annotated preferences based on NSD applied over S_T^1, whose articles were partitioned to paragraph units. We call the resulting semantic preferences the *T-semantics*.

Taking D_{1000} as a dictionary, we generated a training set by sampling uniformly at random $m = 2,000,000$ preferences, which were tagged using the T-semantics. We then applied SemanticSort to learn the T-semantics using this training set while

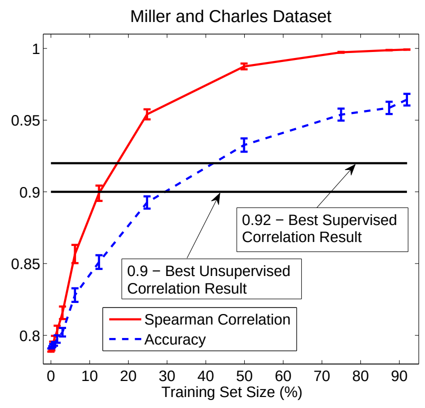

Fig. 5. Experiment 5 – Semantic similarity with Miller & Charles dataset.

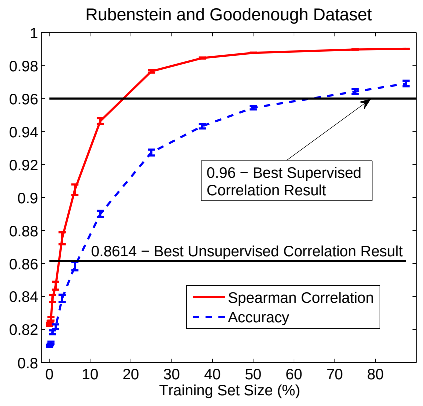

Fig. 6. Experiment 5 – Semantic similarity with Rubenstein and Goodenough dataset.

utilizing S_T^2 (as well as the rest of Wikipedia) as a BK corpus, whose documents were parsed to the paragraph level as well. We then examined the resulting W_T model.

#	play		player		record		club	
	Music	Sports	Music	Sports	Music	Sports	Music	Sports
1	band	game	instrument	play	release	set	dance	football
2	guitar	team	play	league	album	season	night	league
3	instrument	season	replace	game	label	win	heart	cup
4	perform	player	join	season	band	career	fan	play
5	time	football	guitar	born	song	finish	local	divis
6	year	first	technique	team	first	run	house	season
7	role	score	key	football	new	game	London	manage
8	tour	club	example	professional	studio	won	scene	success
9	two	year	football	baseball	production	score	mix	found
10	new	career	hand	major	sign	second	radio	player

Table 1. Model Interpretability - Top 10 related terms according to Music and Sports Semantics.

Two topics T were considered: Music and Sports, resulting in two models: W_{music} and W_{sports}. In order to observe and understand the differences between these two models, we identified and selected, before the experiment, a few target terms that have ambiguous meanings with respect to Music and Sports. The target terms are: `play`, `player`, `record`, `club`. Table 1 exhibits the top 10 most related terms to each of the target terms according to either W_{music} or W_{sports}. It is evident that the semantics portrayed by these lists are quite different and nicely represent their topics as we may intuitively expect. The table also emphasizes the inherent subjectivity in SR analysis, that should be accounted for when generating semantic models.

Given a topical category C in Wikipedia, and a hypothesis h, we define the *aggregate C-weight according to h*, to be the sum of the weights of all contexts that belong to an article that is categorized into C or its Wikipedia sub-categories. Also, given a topic T, we denote by h_{init}^T, its initial hypothesis and by h_{final}^T, its final hypothesis (after learning). In order to evaluate the influence of the labeling semantics on h_{final}^T, we calculated, for each topic T, the difference between its aggregate C-weights according to h^T_{init}, and according to h^T_{final}.

Figure 7 presents the increase/decrease in those aggregate C-weights for some of Wikipedia's major categories C. In both cases of labeling topics, Music or Sports, observe that by and large the aggregate weights of categories that are related to the labeling topic were increased, while weights of unrelated categories were decreased. Quite surprisingly, when considering the Music topic, many mathematical categories dramatically increased their weight. Various other interesting relations are highlighted by this process. For example, notice the sharp decrease of political topics in the Sports model (right), and the decrease of wars and military in the music model (left).

While these results aren't conclusive (and can certainly be viewed as anecdotal), we believe they do indicate that the weights in W are organized in a meaningful and interpretable manner, which encodes the labeling semantics as a particular weight distribution over the corpus topics. In addition, not only did SemanticSort identified the labeler BK, it also revealed some other unexpected topical relations.

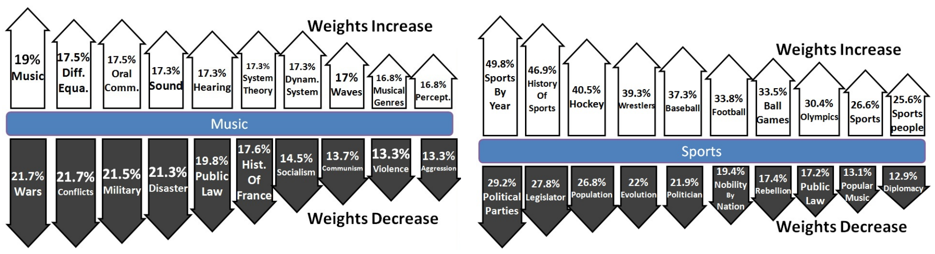

Fig. 7. Model Interpretability – Weights increase (upper/white) and decrease (lower/black) of Wikipedia's major categories according to Music hypotheses (left) and Sports (right)

8 Concluding Remarks

Building on successful and interesting ideas, we presented in this work a novel supervised method for learning SR. The proposed SemanticSort algorithm exhibits interesting performance over a large and medium scale problems and competitive performance on small scale problems. The SemanticSort algorithm performs feature re-weighting as done by multiplicative update algorithms such as Winnow. Moreover, any (kernel) inner product algorithm such as SVM can be applied on a vectorial representation of features corresponding to contexts as induced by the proposed semantic model. However, from a learning theoretic perspective, the use of general purpose classification algorithms over those features can be problematic due to the huge number of contexts in our model, which may unfortunately lead to overfitting. In contrast, by its construction with appropriate f_W function (such as the WNSD), SemanticSort can only output permutations over the set of word pairs of size $\binom{|D|}{2}$. Thus, the hypothesis space considered by SemanticSort is a subset of those permutations, whose VC-dimension can be shown to be precisely $\binom{|D|}{2} - 1$. Moreover, the utilization of the BK corpus through the WNSD measure, provides further capacity reductions by placing many constraints on the set of allowable permutations. For example, observe that SemanticSort only updates the weights of contexts that include both terms in a given pair, so it cannot change the semantic relatedness score of terms that do not co-occur.

It would be nice to gain further theoretical insights on SemanticSort that can explain its successful performance. Another interesting question is how to extend our SR model and the algorithm to allow for active learning. The performance of SemanticSort depends on appropriate choice of the learning rate function $\lambda(\cdot)$. It is desired to explore possibilities for automatically adapting this learning rate. Our results indicate that high quality SR can be learned with markedly different types of BK corpora. It would be very interesting to obtain better understanding on the role of the BK corpus, and perhaps even quantify the effectiveness of a given BK corpus. Finally, SemanticSort is a general ranking algorithm in the sense that it can be applied with any feature vector accompanied with appropriate f_W. It is interesting to explore applications of the algorithm for ranking problems in other domains such as media (images, music) ranking.

References

1. Broder, A., Fontoura, M., Josifovski, V., Riedel, L.: A semantic approach to contextual advertising. In: SIGIR, ACM (2007)
2. Guha, R., McCool, R., Miller, E.: Semantic search. In: WWW, ACM (2003)
3. Green, S.: Building hypertext links by computing semantic similarity. IEEE **11** (1999)
4. Gabrilovich, E., Markovitch, S.: Overcoming the brittleness bottleneck using wikipedia: Enhancing text categorization with encyclopedic knowledge. In: AAAI. (2006)
5. Sun, X., Wang, H., Yu, Y.: Towards effective short text deep classification. In: SIGIR, ACM (2011)
6. Budanitsky, A., Hirst, G.: Evaluating wordnet-based measures of lexical semantic relatedness. Comp. Ling. **32** (2006) 13–47
7. Bloehdorn, S., Moschitti, A.: Structure and semantics for expressive text kernels. In: CIKM, ACM (2007)
8. Gabrilovich, E., Markovitch, S.: Wikipedia-based semantic interpretation for natural language processing. AI Research (2009)
9. Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., Ruppin, E.: Placing search in context: the concept revisited. In: WWW. (2001)
10. Ponzetto, S.P., Strube, M.: Knowledge derived from wikipedia for computing semantic relatedness. JAIR (2007)
11. Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Pasca, M., Soroa, A.: A study on similarity and relatedness using distributional and wordnet-based approaches. In: NAACL. (2009)
12. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: IJCAI. (1995)
13. Jiang, J., Conrath, D.: Semantic similarity based on corpus statistics and lexical taxonomy. CoRR (1997)
14. Li, Y., Bandar, Z.A., McLean, D.: An approach for measuring semantic similarity between words using multiple information sources. IEEE **15** (2003) 871–882
15. Jarmasz, M., Szpakowicz, S.: S.: Roget's thesaurus and semantic similarity. In: RANLP. (2003)
16. Tsatsaronis, G., Varlamis, I., Vazirgiannis, M., Nørvåg, K.: Omiotis: A thesaurus-based measure of text relatedness. In: ECML PKDD. (2009)

17. Tsatsaronis, G., Varlamis, I., Vazirgiannis, M.: Text relatedness based on a word thesaurus. JAIR (2010)
18. Lesk, M.: Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: SIGDOC, ACM (1986) 24–26
19. Banerjee, S., Pedersen, T.: Extended gloss overlaps as a measure of semantic relatedness. In: IJCAI. (2003)
20. Patwardhan, S., Pedersen, T.: Using wordnet based context vectors to estimate the semantic relatedness of concepts. In: EACL. (2006)
21. Morris, J., Hirst, G.: Lexical cohesion computed by thesaural relations as an indicator of the structure of text. Comp. Ling. (1991)
22. Hirst, G., St-Onge, D.: Lexical chains as representations of context for the detection and correction of malapropisms. In: WordNet. (1998)
23. Radinsky, K., Agichtein, E., Gabrilovich, E., Markovitch, S.: A word at a time: computing word relatedness using temporal semantic analysis. In: WWW. (2011)
24. Milne, D., Witten, I.: An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: AAAI Workshop. (2008)
25. Yeh, E., Ramage, D., Manning, C., Agirre, E., Soroa, A.: Wikiwalk: random walks on wikipedia for semantic relatedness, ACL (2009)
26. Liberman, S., Markovitch, S.: Compact hierarchical explicit semantic representation. In: IJCAI Workshop. (2009)
27. Haralambous, Y., Klyuev, V.: A Semantic Relatedness Measure Based on Combined Encyclopedic, Ontological and Collocational Knowledge. Computing Research Repository (2011)
28. Lin, D.: An information-theoretic definition of similarity. In: ICML. (1998)
29. Dagan, I., Lee, L., Pereira, F.: Similarity-based models of cooccurrence probabilities. Machine Learning **34** (1999) 43–69
30. Terra, E., Clarke, C.: Frequency estimates for statistical word similarity measures. In: NAACL. (2003)
31. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. JASIST (1990)
32. Reisinger, J., Mooney, R.: Multi-prototype vector-space models of word meaning. In: NAACL. (2010) 109–117
33. Bollegala, D., Matsuo, Y., Ishizuka, M.: Measuring semantic similarity between words using web search engines. In: WWW. (2007)
34. Eyke, H., Johannes, F., Weiwei, C., Klaus, B.: Label ranking by learning pairwise preferences. AI **172**(16-17) (2008) 1897–1916
35. Das Sarma, A., Das Sarma, A., Gollapudi, S., Panigrahy, R.: Ranking mechanisms in twitter-like forums. In: WSDM. (2010)
36. Recchia, G., Jones, M.: More data trumps smarter algorithms: Comparing pointwise mutual information with latent semantic analysis. Behavior Research Methods (2009)
37. Cilibrasi, R., Vitanyi, P.: The google similarity distance. IEEE **19** (2007) 370–383
38. Francis, W.N., Kucera, H.: Frequency analysis of English usage: Lexicon and grammer. Houghton Mifflin (1982)
39. Vapnik, V.: Statistical Learning Theory. Wiley Interscience (1998)
40. Rubenstein, H., Goodenough, J.B.: Contextual correlates of synonymy. Commun. ACM (1965)
41. Miller, G.A., Charles, W.G.: Contextual correlates of semantic similarity. Language and Cognitive Processes **6** (1991)
42. Hughes, T., Ramage, D.: Lexical semantic relatedness with random graph walks. In: EMNLP-CoNLL. (2007)