Corpus-Based Neural Network Method for Explaining Unknown Words by WordNet Senses

Bálint Gábor, Viktor Gyenes, and András Lőrincz*

Eötvös Loránd University, Pázmány P. sétány 1/C, Budapest 1117
{gbalint, gyenesvi}@inf.elte.hu, andras.lorincz@elte.hu
http://nipg.inf.elte.hu/

Abstract. This paper introduces an unsupervised algorithm that collects senses contained in WordNet to explain words, whose meaning is unknown, but plenty of documents are available that contain the word in that unknown sense. Based on the widely accepted idea that the meaning of a word is characterized by its context, a neural network architecture was designed to reconstruct the meaning of the unknown word. The connections of the network were derived from word co-occurrences and word-sense statistics. The method was tested on 80 TOEFL synonym questions, from which 63 questions were answered correctly. This is comparable to other methods tested on the same questions, but using a larger corpus or richer lexical database. The approach was found robust against details of the architecture.

1 Introduction

The Internet is an immensely large database; large amount of domain specific text can be found. Intelligent tools are being developed to determine the meaning of documents, and manually created lexical databases are intended to provide help for such tools. However, manually assembled lexical databases are unable to cover specific, emerging subjects, thus documents may contain words of unknown meanings; words that are not contained in the lexical databases, or the contained meanings do not fit into the context found in the documents. However, the meaning of these words can often be inferred from their contexts of usage. The aim of our method is to explain words that are unknown to a human or machine reader, but are contained in many documents in the same sense.

To achieve this goal, our method looks for WordNet senses that are semantically close to the unknown meaning of the word. We rely on the common practice of measuring the similarity of words based on their contextual features, and designed a neural network architecture by means of three databases as sources of information. The first is WordNet[1], where the words are grouped into synonym sets, called *synsets*. The second source of information that our method exploits is SemCor[2], which is a corpus tagged with WordNet senses. SemCor was used

* Corresponding author

[1] http://www.cogsci.princeton.edu/∼wn/

[2] http://www.cs.unt.edu/∼rada/downloads/semcor/semcor2.0.tar.gz

A. Jorge et al. (Eds.): PKDD 2005, LNAI 3721, pp. 470–477, 2005.

to obtain information on the statistical distributions of the senses of the words. The third database used is the British National Corpus (BNC[3]), a collection of English texts of 100 million words. Our assumption is that the meaning of a word is similar to the meaning of a *sense*, if they appear in the same context, where we define context by the *senses* that they often co-occur with[4].

Testing of any new method requires a controllable benchmark problem. Our method was evaluated on 80 synonym questions from the Test of English as a Foreign Language (TOEFL[5]). The system scored 78.75% (63 correct answers). Many other studies had also chosen this TOEFL benchmark problem: Landauer and Dumais's Latent Semantic Analysis (LSA) [1] is based on co-occurrences in a corpus, and it provides generalization capabilities. It was able to answer 64.4% of the questions correctly. Turney's Pointwise Mutual Information Information Retrieval (PMI-IR) algorithm [2] performed 73.25% on the same set of questions. This is also a co-occurrence based corpus method, which examines noun enumerations. It uses the whole web as a corpus and exploits AltaVista's special query operator, the *NEAR* operator. Terra and Clarke [3] compared several statistical co-occurrence based similarity measures on a one terabyte web corpus, and scored 81.25%. Jarmasz and Szapakovicz constructed a thesaurus-based method [4], which performed 78.75% on these questions. They utilized Roget's Thesaurus to calculate path lengths in the semantic relations graph between two words, from which a semantic similarity measure could be derived.

This paper is organized as follows: Section 2 details the neural network method. Section 3 describes the various test cases and presents the results. Discussion is provided in Section 4, conclusions are drawn in Section 5.

2 Methods

As it was already mentioned, our method exploits three databases (Fig. 1(A)). BNC is used to obtain word co-occurrence statistics. The following estimations are also required: given a synset, how frequently is one of its words used to express that sense, and, on the other hand, if a word is used, how frequently is it used in one of its senses. Database SemCor was used to obtain these statistics. Since all these pieces of information are needed, we only used words and synsets which occur in SemCor at least once. This means 23141 words and 22012 synsets. Later, we extended these sets by adding the trivial synsets, which contained single words. Then we could experiment with 54572 words and 53443 synsets.

2.1 Co-occurrence Measures

The aim of our method is to find semantically close synsets to an unknown word. Two words that occur in similar contexts can be considered as similar in

[3] `http://www.natcorp.ox.ac.uk/`
[4] The words *sense* and *synset* is used interchangeably in this paper.
[5] `http://www.ets.org/`

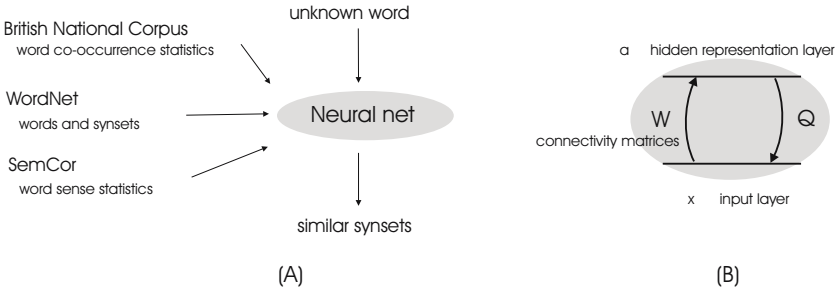

Fig. 1. Databases and reconstruction network architecture (A): Scheme of our method. (B): Basic computational architecture. Input layer (x) and hidden representation (a) are connected by bottom-up (W) and top-down (Q) matrices.

meaning. Therefore, we need to express the measures of co-occurrence between words and synsets; i.e. values indicating how often words and synsets co-occur.

The probability that word w_1 occurs near word w_2 can be estimated as follows: $P(w_1|w_2) = \frac{f(w_1,w_2)}{f(w_2)}$, where $f(w_1, w_2)$ is the number of times w_1 and w_2 co-occur in a 5 wide context window[6] and $f(w)$ is the frequency of word w. We say that w_1 and w_2 are *near words* of each other, if both $P(w_1|w_2)$ and $P(w_2|w_1)$ are high, meaning that w_1 and w_2 are likely to co-occur. The following measure derived from mutual co-occurrences expresses this idea: $N(w_1, w_2) = min(P(w_1|w_2), P(w_2|w_1))$. It is expected that this co-occurrence measure describes the contexts of the words. Given a word w, we call the *near word list* of w is the 100 words w_i for which the $N(w, w_i)$ values are the highest. This near word list can be represented as a *feature vector* of the word, the entries of the vector are the $N(w, w_i)$ values. Then the co-occurrence information about the words can be summarized in a quadratic and symmetric matrix N_W, where the i^{th} row of the matrix is the feature vector of the i^{th} word: $N_W(i, j) = N(w_i, w_j)$, where w_j is the j^{th} word in our vocabulary.

In SemCor, every occurrence of a word is tagged with a WordNet synset that expresses the meaning of the actual occurrence of the word. By counting these tags we can compute the desired probabilities. The probability that for a given word w, the expressed sense is s, can be estimated as $P(s|w) = \frac{f(w,s)}{f(w)}$, where $f(w, s)$ is the frequency of word w in sense s and $f(w)$ is the frequency of word w in any of its senses. We also need the probability that a given sense s is expressed by word w, which can be estimated as: $P(w|s) = \frac{f(w,s)}{f(s)}$, where $f(s)$ is the frequency of sense s, whichever word it is expressed by. These probabilities can also be summarized in matrix forms, denoted by S_W and W_S: $S_W(i, j) = P(s_i|w_j)$ and $W_S(i, j) = P(w_i|s_j)$.

Using the measures introduced, a co-occurrence measure between synsets can be derived. The idea is the following: given a synset s, the *near synsets* of s are the synsets of the near words of the words expressing s. This idea is

[6] Increasing the context window by a factor of 2 had no significant effects.

expressed by the appropriate concatenation of the three matrices introduced above: $N_S = S_W N_W W_S$.

2.2 Reconstruction Networks

The basic reconstruction network model has two neuron layers. Connections bridge these layers. The lower layer is the input layer of the network, and the upper layer is called the hidden or internal representation layer (Fig. 1(B)). The network reconstructs its input by optimizing the hidden representation. For this reason, we call it reconstruction network. Formally, the following quadratic cost function is involved [5]:

$$J(a) = ||x - Qa||_2^2 \ , \tag{1}$$

where Q is the connectivity matrix, x is the input vector, a is the hidden representation vector. The columns of the connectivity matrix can be thought of as basis vectors, which must be linearly combined with the appropriate coefficients so that the combination falls close to the input. The optimization can either be solved directly

$$a = (Q^T Q)^{-1} Q^T x \ , \tag{2}$$

or iteratively

$$\Delta a \propto W(x - Qa) \ , \tag{3}$$

where $W = Q^T$, which can be derived from the negative gradient of cost function (1). The form of (3) is more general than required by (1) but it still suitable as long as WQ is positive definite. Both methods have advantages and disadvantages. Directly solving the optimization returns the exact solution, but might require a considerable amount of memory, while the iterative solution requires less resources, but is computationally intensive.

The reconstruction network described above shall be called 'one-tier' network. We designed both 'one-tier' and 'two-tier' networks for the word-sense reconstruction. The two-tier network has two one-tiers on the top of each other. The internal representation layer of the first tier serves as input for the second tier. There are differences between the two architectures in computation speed and in numerical precision.

The feature vector representation of the context of the unknown word serves as input for the network. In the hidden layer, the neurons represent the candidate synsets. In the one-tier network (Fig. 2(A)), the top-down and bottom-up matrices are defined as follows: $Q = W_S N_S$ and $W = N_S^T S_W$[7]. Thus, in the one-tier method, hidden synset activities are (a) transformed to near synset activities and then (b) the activities of the near words are generated. An illustrative iteration is depicted in Fig. 2(B): The activities in the topmost layer change during the reconstruction of the input word *frog*. It can be seen that only a few activities become high, others remain small. Note the horizontal scale: there were about 23,000 neurons in the topmost layer in this iteration.

[7] However, we found that $\tilde{Q} = N_W W_S$ and $\tilde{W} = \tilde{Q}^T = W_S^T N_W^T$ are simpler, express the same relations and converge faster, thus these were used in the computations.

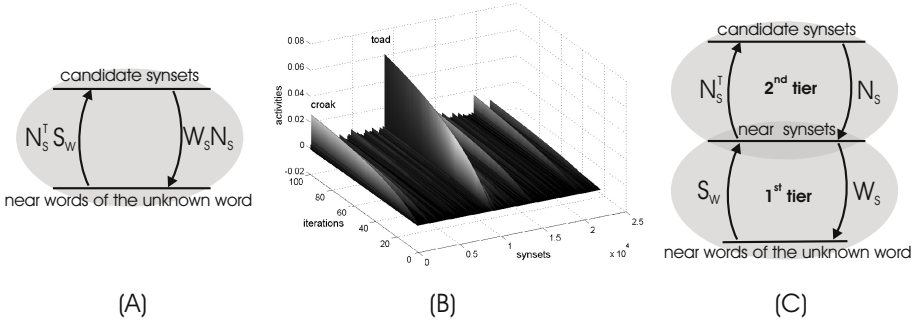

Fig. 2. One-tier and two-tier networks (A) and (C): one and two tier networks. The input of the i^{th} node of the lowest layer is $N(w, w_i)$, where w is the unknown word. $S_W(i,j) = P(s_i|w_j)$, $W_S(i,j) = P(w_i|s_j)$, $N_S = S_W N_W W_S$, where $N_W(i,j) = N(w_i, w_j)$ for all i, j. The *result* of the computation is the synset represented by the highest activity unit of the top layer after running the network. (B): Convergence of the iterative approximation. Input word is *frog*. High activity nodes correspond to sense *toad* and *croak*.

In the 'two-tier' network (Fig. 2(C)) the two steps of the transformation are separated. The nodes in the intermediate layer correspond to the near synsets of the unknown word. The bottom-up and top-down matrices are the S_W and the W_S matrices, respectively. In the second tier the connectivity matrix is N_S in both directions, which contains the synset near values. This captures the idea mentioned in the introduction; the meaning of the unknown word is similar to the meaning of a synset if they have the same near *synsets*.

3 Tests and Results

We tested our method on 80 TOEFL synonym questions, each question consisted of a *question word* (for example *grin*) and four *candidate answer words*, (for example: *exercise, rest, joke, smile*). The task was to find the candidate word that was the most similar in meaning to the question word (*smile*).

To meet our goals, we considered the question word the unknown word. We simulated the situation of the question word being *unknown* by erasing all information about the question word and its meaning from the SemCor statistics and WordNet synsets. After running the network for the context of the question word as input, we examined the activities corresponding to the synsets of the candidate words, and assigned a value to each candidate word equal to the highest activity of the synsets of the candidate word in the upper layer of the network. The candidate word with the highest value was the chosen answer.

In the one-tier network and in the second tier of the two-tier network, the huge number of connections between the nodes required the application of the less precise iterative method. However, the first tier of the two-tier network could be optimized directly, because the connectivity matrices were very sparse.

In some cases we have additional – top-down (TD) – information about the unknown word, for example its part-of-speech or the candidate answer words, this reduces the set of candidate synsets. The implementation of this filtering is simple in our system, since we can simply leave out the unnecessary synsets.

In order to test whether or not using synsets increases the efficiency of the system, we constructed a one tier *control* network which used only words. The input of the network was the same, however, the nodes in the upper layer corresponded to words. The connectivity matrix between the two layers is N_W, as defined in 2.1. The activities of the upper layer were examined; words having similar context as the input word were returned.

The number of correct answers in the various cases can be seen in Table 1. The best result, 63 correct answers (78.75%), was produced with part-of-speech constraint with the 23141 word data set and also without TD constraint with the 54572 word data set, utilizing the one-tier network. However, the best first iterations were achieved by the two-tier network. It can be seen, that iteration has improved the precision in almost all the cases. The control network started from 53 and 54 correct answers for the two word sets and reached 60 and 59 correct answers, respectively, by iterations. These results are considerably smaller than those of without the synsets, which supports our starting assumptions. We should note that if information related to the question words are not deleted from the database, then the number of correct answers is 68, which *amounts to 85%*.

Table 1. Results *No constr*: No constraint is applied. *PoS*: Synsets in the upper layer correspond to the part-of-speech of the word. *Candidate*: Only the synsets of the candidate words are used. *Control*: Single tier control network. *Direct*: non-iterative solution.

23141 words	No constr		PoS		Candidate		Control
	1 tier	2 tiers	1 tier	2 tiers	1 tier	2 tiers	1 tier
1 iter	55	58	55	58	55	58	53
10 iter	60	57	61	59	55	58	56
100 iter	61	60	63	58	58	56	60
direct	-	-	-	-	57	55	-

54572 words	No constr		PoS		Candidate		Control
	1 tier	2 tiers	1 tier	2 tiers	1 tier	2 tiers	1 tier
1 iter	56	59	56	59	56	59	54
10 iter	62	60	59	58	56	59	56
100 iter	63	61	62	58	57	56	59
direct	-	-	-	-	57	57	-

By examining the activities corresponding to the candidate answer words, decision points can be incorporated. Then the system may deny to answer a question, if the answer is uncertain. We could improve precision but the number of answers decreased considerably. Still, this property should be useful in multiple expert schemes, where experts may be responsible for different domains.

4 Discussion

Compared to the other methods, LSA performs relatively poorly (64.5%). However, the original intention of LSA was not to serve as an efficient TOEFL solver, but to model human memory. LSA reads the dictionary (the text database of the experiment) and runs the singular value decomposition only once and without knowing anything about the questions beforehand. After this procedure LSA can immediately answer the questions. While the first phase in LSA models a person's general learning process, this second phase imitates how someone solves questions without relying on any external aid [6]. By contrast, many other methods are allowed to use their databases after they have observed the questions.

Our method resembles LSA. Alike LSA, it works by the optimization of reconstruction using hidden variables over Euclidean norm. We also build a kind of memory model (the connectivity matrices of the neural network) before the questions are observed. When the questions are observed, the answer can be produced by running the network. Alike to LSA, our method was not developed for solving TOEFL questions, but for explaining unknown words. Considering this, our comparably high score (78.75%) is promising. True though, the original network incorporates information contained in WordNet and SemCor, however, the strength of the approach is shown by the *control network*, which did not use any lexical information, and gave 60 correct answers (i.e., 75%).

The Hyperspace Analogue to Language (HAL) model [7] works in high dimensions alike to our method. According to the HAL model, the strength of a term–term association is inversely proportional to the Euclidean distance between the context and the target words. Alike to HAL, our method makes use of the whole table of co-occurrences. This seems important; the larger table gave better result for us. Our method combines the advantages of LSA and HAL: it makes use of all information like HAL and adopts hidden variables like LSA.

We also included other information, the uncertainty of the answer, that goes beyond the statistics of co-occurrences. It may be worth noting here that our approach can be generalized to hidden, overcomplete, and sparse representations [8]. Such non-linear generalizations can go beyond simple computational advantages when additional *example based information* [9] or *supervisory training* are to be included.

A recent paper on meaning discovery using Google queries [10] thoroughly details the development of semantic distances between words. The method uses first order co-occurrence counts (Google page counts) to determine the semantic distance of two words. The article describes a semantic distance called Normalized Google Distance (NGD), derived from the same formula that we use to calculate the co-occurrence measure of two words. However, in our case, the formula was used to examine second order co-occurrences instead of first order co-occurrences. We conducted two studies with NGD. First, we solved the 80 TOEFL synonym questions using NGD as described in the paper; we measured the distance of the question word and each candidate word, and chosen the one with the smallest distance. Depending on the database we used to collect word frequencies, the results were different, however, surprisingly low: 30 correct an-

swers (37.5%) when BNC was used, and 40 correct answers (50.0%) when Google was used to return the page counts needed for NGD. In the other study we used our neural network method based on NGD instead of our co-occurrence measure. Results in this case were almost identical to the original setting, when we used our own co-occurrence measure, indicating the robustness of our solution against these details.

5 Conclusions

We have studied neural network architectures for explaining unknown words by known senses, senses that are contained in our lexical databases. We tested the method on TOEFL synonym questions. It was found that the networked solution provided good results, and was found robust against the details. At the cost of decreasing recall, the precision of the system can be improved. These features make our method attractive for various circumstances.

Acknowledgments

We are grateful to Prof. Landauer for providing us the TOEFL examples. We thank one of the referees for calling our attention to Burgess' HAL model.

References

[1] Landauer, T.K., Dumais, S.T.: A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. Psychol. Rev. **104** (1997) 211–240
[2] Turney, P.: Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In: ECML Proceedings, Freiburg, Germany (2001) 491–502
[3] Terra, E., Clarke, C.L.A.: Choosing the word most typical in context using a lexical co-occurrence network. In: Proc. of Conf. on Human Language Technol. and North American Chapter of Assoc. of Comput. Linguistics. (2003) 244–251
[4] Jarmasz, M., Szpakowicz, S.: Roget's Thesaurus and semantic similarity. In: Proc. of the Int. Conf. on Recent Advances in Natural Language Proc. (RANLP-03). (2003)
[5] Haykin, S.: Neural Networks: A comprehensive foundation. Prentice Hall, New Jersey, USA (1999)
[6] Landauer, T.K. (personal communication)
[7] Burgess, C.: From simple associations to the building blocks of language: Modeling meaning in memory with the HAL model. Behav. Res. Methods, Instr. and Comps. **30** (1998) 188–198
[8] Olshausen, B.A.: Learning linear, sparse factorial codes. A.I. Memo 1580, MIT AI Lab (1996) C.B.C.L. Paper No. 138.
[9] Szatmáry, B., Szirtes, G., Lőrincz, A., Eggert, J., Körner, E.: Robust hierarchical image representation using non-negative matrix factorization with sparse code shrinkage preprocessing. Pattern Anal .and Appl. **6** (2003) 194–200
[10] Cilibrasi, R., Vitanyi, P.: Automatic meaning discovery using google. arXiv:cs.CL/0412098 v2 (2005)