A Preliminary Functional Analysis of Memory in the Word Sense Disambiguation Task

Nate Derbinsky[1] **and John E. Laird**[1]

Abstract. We focus on the problem of efficiently retrieving knowledge from large memories given ambiguous cues. First, we analyse the word sense disambiguation task in context of memory model comparison and evaluation. Then, in this task, we demonstrate the functional benefit of two forms of memory retrieval bias, recency and frequency of memory access, and present a preliminary evaluation of heuristics to efficiently support these biases in memory systems.

1 INTRODUCTION

One advantage the human brain demonstrates over the current generation of artificially intelligent agents is its ability to extract diverse, useful experiences from its interactions with the world; store large amounts of this information in memory for long periods of time; and later retrieve this knowledge from memory when it is relevant to making decisions and taking action. There is evidence that extending agents with long-term memory supports many functional cognitive capabilities [1]; however, maintaining and querying large memories poses significant computational challenges that currently make it impossible to task these agents with real-world problems.

The focus of this paper is one specific challenge facing long-term memory: given a large store of knowledge, how should the system respond to an ambiguous cue, one that pertains to multiple previously encoded memories. Anderson and Schooler [2], positing that human memory optimally solves this problem with respect to the history of past memory access, have developed and validated memory models that are widely used in the cognitive modelling community. However, existing computational implementations of these long-term declarative memory models do not scale to large bodies of knowledge [3].

Previously [4] we developed and evaluated computational techniques to efficiently support queries of large declarative memory stores; however, this work supported only a limited class of bias in the case of ambiguous cues. In this paper, we extend our prior work and evaluate methods for efficiently incorporating recency and frequency of memory access as functional models of memory retrieval bias. We evaluate our methods in the word sense disambiguation (WSD) task, an important and well-studied problem in the Natural Language Processing (NLP) community [5]. We are not attempting to solve the word sense disambiguation problem, but instead our work is intended to provide evidence that (1) the WSD task is an appropriate benchmark when evaluating and comparing memory models and that (2) agents whose long-term memory systems incorporate historical regularities of past memory access will benefit in this task.

We begin with an introduction to the word sense disambiguation task, including an analysis of WordNet [6] and SemCor [7], the datasets we use in our evaluation. We then present results of how simple baseline algorithms perform on the WSD task, including the relative advantage of memory-based algorithms that incorporate the recency and frequency of memory access. Given this performance advantage, we evaluate the WSD performance of the base-level model of memory bias [8], a commonly used model based upon the rational analysis of memory [2] that combines recency and frequency of memory access. As the base-level model performs relatively well in this task and dataset, we motivate, describe, and evaluate preliminary heuristics to efficiently implement this model in a long-term memory system. Finally, we conclude with a discussion of this and future work.

2 WORD SENSE DISAMBIGUATION

Many words in the English language are *polysemous*, that is they have multiple, distinct meanings, or *senses*, which are interpreted differently based upon the context in which they occur. For instance, consider the following sentences:

a) Deposit the check at the *bank*.
b) After canoeing, they rested at the *bank*.

The occurrences of the word *bank* in the two sentences clearly denote different meanings: 'financial institution' and 'side of a body of water,' respectively. Word sense disambiguation is the ability to identify the meaning of words in context in a computational manner [5]. The task of WSD is critical to the field of NLP and has been studied for decades. There are several formulations of and many approaches to this problem.

As the focus of this work is memory, not language processing, we simplify components of the general WSD problem and adopt a variant of the *lexical sample* formulation of the problem. As input, the agent receives a sequence of sentences, each composed of a sequence of words. For simplicity, each word in the input is tagged with its appropriate part of speech (noun, verb, adjective, or adverb). Additionally, the agent has access to a machine-readable dictionary (MRD), such that each lexical word/part of speech pair in the input corresponds to a list of word senses within the MRD. For each sense, the MRD contains a definition and tag frequency from a representative corpus. Thus, for each word in each input sentence, the agent's task is to select the most appropriate sense from the MRD.

3 TASK ANALYSIS

The data source we use is version 3 of the SemCor [9] *semantic concordance*. A semantic concordance is a textual corpus and lexicon linked such that every substantive word in the text is linked to its appropriate sense in the lexicon [7]. SemCor is the largest and most used sense-tagged corpus, which includes 352

[1] Computer Science and Engineering Division, University of Michigan, 2260 Hayward St., Ann Arbor, MI, 48109-2121. Email: {nlderbin, laird}@umich.edu.

texts from the Brown corpus [10]. We use the 186 Brown corpus files that have all content words tagged, which includes more than 185,000 sense references to version 3 of the WordNet lexicon [6]. WordNet 3, the most utilized resource for WSD in English, includes more than 212,000 word senses.

To understand the task at hand, it is useful to examine certain properties of WordNet and SemCor. We begin with aggregate sense size in WordNet, which measures the number senses per lexical word/part of speech pair, and thus the average *ambiguity* faced by an agent attempting to disambiguate an arbitrary input word. In total, the average number of senses per word is 1.33 (std. deviation 1.12), with a minimum of 1 and a maximum of 59. If we aggregate these statistics with respect to word part of speech, we see the breakdown reported in Table 1 (sorted in order of increasing maximum sense size).

Part of Speech	Min.	Max.	Avg.	Std. Dev.
Adverb	1	13	1.2453	0.7379
Adjective	1	27	1.3968	1.0731
Noun	1	33	1.2421	0.8563
Verb	1	59	2.1725	2.5128

Table 1. Part of Speech Sense Size Statistics in WordNet 3.

This summary statistic, however, refers only to WordNet, and thus does not take into account the distribution of words within the SemCor texts. Table 2 indicates this proportion of words in the SemCor data set, aggregated by part of speech. These statistics reveal that nearly 73% of the words fall into the two most ambiguous parts of speech (nouns and verbs).

Part of Speech	Proportion of SemCor
Adverb	10.2%
Adjective	17.1%
Noun	47%
Verb	25.7%

Table 2. SemCor Part of Speech Proportion.

For about 0.33% of SemCor words, multiple senses are equally appropriate within the linguistic context, as annotated by a human interpreter. Thus we introduce *effective* sense size, computed as one divided by ambiguity in a given input context. For example, consider the following sentence from SemCor:

> *Pansies are supposed to like it cool, but those great velvety flowers were healthy and perky in the glaring sun.*

The verb “to like” has five senses in WordNet and in this context, two are considered equally appropriate (“find enjoyable or agreeable” and “want to have”). So while the sense size of this word is 5, its *effective* sense size in context is (1/[2/5]) = 2.5.

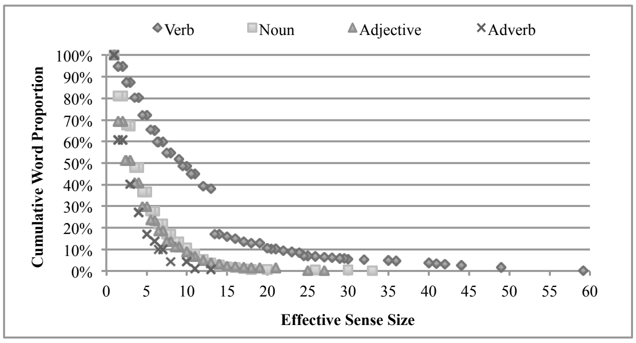

Figure 1. Cumulative SemCor Word Proportion vs. Effective Sense Size

Figure 1 synthesizes the data from Tables 1 and 2, plotting cumulative proportion of words against effective sense size, aggregated by part of speech. This chart is rich with useful trends and statistics that we consider here. First, the data exhibits a strong right skew, containing mostly words with few effective senses. Next, we draw out the proportion of words that require no disambiguation, as their effective sense size is 1, by reading the second plotted point for each part of speech. While for adverbs and adjectives this value is about 40% and 30%, respectively, for nouns and verbs it is about 20% and 5%. Next, we can assess the median effective sense size for each part of speech by reading the x-axis as each part of speech intersects 50% on the y-axis. For adverbs, adjectives, and nouns, this value is between 2 and 4, while for verbs this it is about 10. In summary, the average effective sense size in the SemCor dataset is between 2 and 3 and the expected performance on the WSD task, given a random selection strategy, is 38.73%.

	SemCor Task Performance
Random	38.73%
Static Frequency	76.39%
Lesk	63.40%
Simplified Lesk	65.52%

Table 3. Non-Memory Baseline Results.

4 NON-MEMORY BASELINE ALGORITHMS

To contextualize performance of memory-based algorithms on the WSD task, we first implemented some non-memory baseline algorithms. The results from these baselines are summarized in Table 3, including *random* selection, derived in the previous section. All algorithms we implement select a word sense for all input words, and thus precision and recall are identical for all results, so we report them jointly as “task performance.” Each task performance result we report is the true accuracy of the algorithm on this dataset, as opposed to the sample average of individual probabilistic runs; consequently, even small differences in performance should be considered relevant.

The first baseline was a frequency-biased random selection strategy. As previously described, WordNet includes, for each sense, a static tag frequency from the Brown corpus. As the SemCor textual corpus is a subset of the Brown corpus, we expected this information to be highly informative during sense disambiguation, and unsurprisingly this algorithm yielded nearly twice the performance of pure random selection.

The remainder of the non-memory baselines were variants of the Lesk algorithm for word sense disambiguation [11]. The Lesk algorithm, a commonly used baseline metric [12], assumes that words in a given “neighbourhood” (such as a sentence) will tend to share a common topic, and thus biases sense selection based upon shared terms in sense definitions and context. The classic algorithm finds the maximum overlap between all definitions of all candidate senses in the neighbourhood, and is thus computationally intractable as the size of the neighbourhood grows, so it is common to introduce constant-sized neighbourhood “windows” to reduce search time. A “simplified” Lesk algorithm [13] defines word context as simply the terms in the neighbourhood, as opposed to their definitions, and thus has more tractable growth, scaling with the sense size for the word to be disambiguated. The performance of the Lesk family of algorithms is highly sensitive to the exact wording of sense definitions, and so it is common to supplement Lesk with heuristics and additional sources of semantic meaning (ex. [14]).

For the classic algorithm, we evaluated neighbourhood windows of size 2 and 4 in each sentence. For both classic and simplified, we also evaluated four heuristics. The first was the use of a *stop* list, which excludes definition terms that are common to the target language, such as "a" and "the." The second was to exclude example sentences from sense definitions, as the example terms might pollute overlapping computations. The third was the use of the Porter Stemming [15] algorithm to strip word suffixes with the intention of facilitating overlap of words with common linguistic roots. Finally, we included a bias towards the corpus frequency information described above. We evaluated the full combinatorial set of these parameters across both algorithms. The maximal results (see Table 3) for both classic and simplified algorithms occurred using the stop list, pruned definitions, and frequency bias, but not the Stemming algorithm. For the classic algorithm, the neighbourhood size that yielded greatest task performance was 2. Again, these results are simplistic, very specific to our implementation and data sets, and are not intended for representation of or comparison to modern NLP techniques, but instead to provide a baseline for later memory-based results. These results suggest that we can apply basic techniques, which do not incorporate memory-based methods, and expect to disambiguate up to 76.39% of input words in SemCor.

5 MEMORY BASELINE ALGORITHMS

Inspired by non-memory frequency bias results (see Table 3) and the rational analysis of memory [2], this section investigates algorithms for maintaining a dynamic history, or *memory*, of information presentation and evaluates the degree to which these memories facilitate effective sense selection.

In context of the WSD task, the algorithms described below differ in what information constitutes a history of past sense assignment, and how this information is maintained over time, as well as how, when presented with a word to disambiguate, this history is resolved to select a word sense. Therefore, to evaluate these algorithms, we applied a common evaluation sequence. First, for each word in each input sentence, we performed a read-only *query*, the result of which was scored. We then *presented* to the algorithm the correct sense (or senses) that had been annotated within the SemCor test set. Revealing the correct sense(s) to the memory system eliminates the possibility of unintended divergent learning, which could occur without truthful feedback and would obfuscate the algorithm results.

Unlike the non-memory baselines, these algorithms have the potential to improve with added exposure to the corpus, and thus we performed 10 sequential runs of each. The results are summarized in Table 4 and report task performance on the 1st and 10th run on the SemCor test set.

The first algorithm we evaluated was *recency* of presentation. This algorithm maintains only the most recently presented sense for each lexical word/part of speech pair, which is returned at the time of the next query of the same pair (selecting randomly from amongst multiple simultaneous presentations). This algorithm performs well if the same word sense is used repeatedly in immediate succession.

The next algorithm was *frequency* of presentation. This algorithm maintains the number of presentations of each word sense and then selects the most frequent sense at the time of query. This algorithm performs well if particular senses of words are generally more common than others in a corpus, as opposed to being highly dependent upon sentence context. As an experimental condition, we initialized the frequency of each word sense to its absolute frequency within the full Brown corpus. We found that this initialization provided more than 4% improvement on the first run, but the improvement was only about 0.1% after 10 runs. This condition is labelled "Frequency*" in Table 4. This is comparable to the "Frequency Bias" result in Table 3, with the added improvement in Table 4 coming from updated frequency values as the algorithm gains exposure to the corpus.

Finally, to establish an upper bound on the degree to which recency and frequency can individually contribute to WSD performance, we implemented an *oracle* algorithm. For each word query, this algorithm scores both the *recency* and *frequency* algorithms described above and returns the result that provided the greater score. As with *frequency*, we label as "Oracle*" the variant that initializes frequency with overall corpus frequency. This algorithm performs well for a word query when either recency or frequency is informative to effective sense selection.

	Run 1	**Run 10**
Recency	72.34%	74.43%
Frequency	71.69%	76.53%
Frequency*	75.97%	76.62%
Oracle	79.51%	84.08%
Oracle*	83.87%	84.18%

Table 4. Memory Baseline Results.

We draw three conclusions from the data summarized in Table 4. First, we note that with the exception of pure *recency*, which does not achieve *frequency bias* performance, all memory-based algorithm results for run 10 are greater than all non-memory baselines (see Table 3). This result suggests that memory access history in SemCor, with very little corpus exposure, yields a performance benefit in the WSD task, an advantage that is not dependent upon MRD definition quality (unlike Lesk and its variants). Second, based upon the run 10 results, we can expect memory-endowed agents to disambiguate up to about 84% of SemCor words, simply via memory retrievals, with the potential to improve performance with additional reasoning. Finally, in comparing the run 10 results of "Frequency" vs. "Frequency*" and "Oracle" vs. "Oracle*" we have preliminary evidence that it is unnecessary to bootstrap learning in frequency-biased memories with corpus-specific initialization information, as the empirical history of presentation within the text corpus quickly captures these regularities.

6 MEMORY BIAS MODEL

We have presented evidence that recency and frequency of memory access yield performance benefits in the WSD task on the SemCor dataset. However, to apply these findings to a memory system, we require a model of how these properties combine to bias selection of word senses (recall that the *oracle* algorithm in the previous section is not possible to implement, as

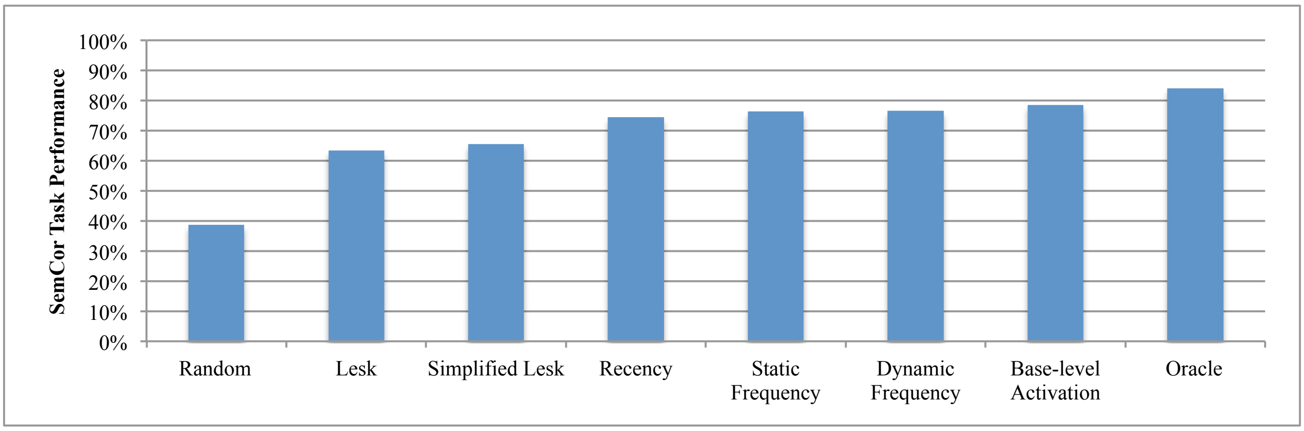

Figure 2. SemCor task performance comparison of non-memory baseline algorithms and run 10 memory-based algorithms.

it requires the memory system to evaluate correct sense assignments during word sense selection).

The base-level activation component of the declarative memory module in the ACT-R cognitive architecture [8], based upon the rational analysis of memory [2], offers one such model that has been widely used within the cognitive modelling community. This model computes activation bias of a memory according to the following equation:

$$\ln\left(\sum_{j=1}^{n} t_j^{-d}\right)$$

where n is the number of presentations of the memory, t_j is the time since the j^{th} presentation, and d is a free decay parameter.

We evaluated this model in the same fashion as the memory baseline algorithms. The only experimental condition was the value of the decay parameter, over which we performed an exploratory sweep of 12 values between, but not including, 0 and 1, and found that d=0.7 resulted in the best run 10 performance. We found that base-level activation yielded 74.45% task performance on SemCor in the first run and 78.47% after 10 runs. This run 10 task performance bests all non-memory and memory baselines, and the run 1 task performance is an improvement to the *recency* and *frequency* run 1 results.

Figure 2 summarizes SemCor task performance. It includes all non-memory baselines, as well as those memory baselines that are not initialized using corpus-specific information. Note that for clarity, we have re-named the memory form of *frequency* as "Dynamic Frequency."

7 EFFICIENT MEMORY BIAS

Data in the previous section demonstrates that the base-level model is relatively effective in the WSD task on the SemCor dataset. Additionally, base-level activation is the dominant model used for cognitive models of human memory phenomena. However, it has been shown to not scale to large bodies of knowledge for long-lived agents [16]. Thus we consider here the challenges associated with efficiently integrating this model as a source of bias within a long-term memory system.

One scaling issue arises when calculating base-level bias as presentation history (n) grows large. However, there are known methods to mitigate this problem (such as a constant-sized history) and so we do not discuss this issue further.

The primary scaling challenge occurs because the base-level model includes a sum over memory presentations (t_j), and that these temporal distances change at *each* point in time for *every* memory. Thus, a naïve integration of the base-level model as a source of bias in memory retrieval must calculate bias values for each candidate memory, a potentially expensive computation given large memory stores and ambiguous cues. Our prior work [4] details methods for efficiently biasing memory retrievals, assuming that only a *constant* number of memories change bias value at any point in time. To better satisfy this assumption for the base-level model, the remainder of this section provides a preliminary evaluation, within the WSD task, of two novel heuristics that seek to identify *only* those memories for which bias *must* be calculated during a memory retrieval. We begin with a description of evaluation metrics and baselines, describe the heuristics, and analyse our results (Table 5).

The first metric, *updates*, refers to the number of memories that require bias calculation during retrieval (lower is better). The *average* is a measure of expected efficiency and the *maximum* refers to expected reactivity. The second metric, *validation*, is a measure of quality and refers to the proportion of queries in the WSD task that result in the same word sense as a *naïve* baseline, wherein bias calculations are performed for all candidate memories. Table 5 also includes a *stable* baseline, wherein memory bias calculations only occur at the time of presentation. This heuristic exploits a regularity of the base-level model: from the time that bias is calculated for a memory, this value is guaranteed to *over-estimate* the true bias value until the memory is presented again in the future. Note that while this heuristic requires no updates, validation suffers by 27.15%.

Our first novel heuristic, *NT*, refers to the (N)umber of memory accesses and most recent (T)ime of access. Both of these statistics can be efficiently maintained incrementally and we reasoned that if *neither* of these values of memory A is greater than that of memory B, it is unlikely that the bias value of A is greater than that of B, and therefore it is unnecessary to compute the bias value of A. This heuristic reduces the average number of updates by 55%, as compared to *naïve*, and maintains a very high level of validation, as compared to *stable*, but has no impact on maximum updates.

To reduce maximum updates, we developed a second novel heuristic, *NTM*, which augments *NT* with incremental (M)aintenance of memories: NTM clears the presentation history, and updates the bias value, of those memories for which the time since the most recent presentation is greater than a threshold (τ). This type of incremental maintenance can be implemented efficiently [17] and we reasoned that the result over time would be many fewer memory candidates with substantive presentation histories. We found that increasing the maintenance threshold (i.e. permitting "older" histories) had the effect of decreasing average updates, while increasing maximum updates and validation; however, due to the exponential decay of the base-level model, all three metrics exhibited "knees". We performed a preliminary sweep of the threshold parameter on SemCor and report the data in Table 5 for the setting that yielded the fewest maximum updates and greatest validation (τ=10). The result is a more than 80% reduction in maximum updates, as compared to *NT*, while maintaining a high level of validation, as compared to *stable*, and a moderate average number of updates, as compared to *NT* and *naïve*. This data represents initial evidence that a high-fidelity base-level model can be efficiently implemented in a memory system, even as the number of memories grows large.

	Avg Updates	Max Updates	Validation
Naïve	2.94	31	100%
Stable	0	0	72.85%
NT	1.32	31	100%
NTM	1.74	6	99.87%

Table 5. Heuristic Results (τ=10).

8 DISCUSSION

We have analysed a formulation of the WSD task on the SemCor data set and have shown preliminary evidence that recency and frequency of sense assignment, biases common to human-inspired computational memory models, are beneficial to task performance. We have also presented evidence that in this task and data set, the base-level model [8] is effective at combining these properties as a source of retrieval bias, and we described and evaluated preliminary heuristics to efficiently incorporate base-level activation within a long-term memory system.

Our over-arching goal is to develop long-term memory mechanisms that are efficient and effective across a wide variety of tasks. We have made progress towards evaluating one class of memory bias on one data set for the word sense disambiguation task, but this leaves much future work to be done. First, to make sure we are not over fitting for the SemCor dataset, we plan to evaluate additional WSD datasets (ex. SENSEVAL [13]). Furthermore, to gather evidence that recency and frequency of memory access are generally useful in a long-term memory system, we plan to evaluate these biases in tasks other than WSD. While the base-level model has been shown to be effective, there is additional room for improvement, as illustrated by the task performance of the *oracle* algorithm, and thus we also plan to develop and evaluate additional memory bias models. We also plan to evaluate the computational run-time of bias algorithm implementations within real agents, such as to understand the trade-offs between computational efficiency, model validity, and bias functionality. Finally, our results demonstrate that memory retrievals alone can successfully disambiguate up to 84% of the words in SemCor, but what of the remaining words? We plan to integrate this work into running agents and explore the interactions of memory retrievals and other, complimentary processing mechanisms and sources of knowledge.

ACKNOWLEDGEMENTS

The authors acknowledge the funding support of the Air Force Office of Scientific Research.

REFERENCES

[1] Nuxoll, A., Laird, J. E. Extending Cognitive Architecture with Episodic Memory. In: *Procs. of AAAI* (2007).

[2] Anderson, J., Schooler, L. Reflections of the Environment in Memory. In: *Psychological Science*, 2 (6), pp. 396-408 (1991).

[3] Douglass, S., Ball, J., Rodgers, S. Large Declarative Memories in ACT-R. In: *Procs. of ICCM* (2009).

[4] Derbinsky, N., Laird, J. E., Smith, B. Towards Efficiently Supporting Large Symbolic Declarative Memories. In: *Procs. of ICCM* (2010).

[5] Navigli, R. Word Sense Disambiguation. *ACM Computing Surveys*, 41 (2), pp. 1-69 (2009).

[6] Miller, G. A. WordNet: A Lexical Database for English. *Communications of the ACM*, 38 (11), pp. 39-41 (1995).

[7] Miller, G. A., Leacock, C., Tengi, R., Bunker, R. T. A Semantic Concordance. In: *Procs. of DARPA Workshop on Human Language Technology* (1993).

[8] Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., Qin, Y. An Integrated Theory of the Mind. In: *Psychological Review*, 111 (4), pp. 1036-1060 (2004).

[9] Mihalcea, R., http://www.cse.unt.edu/~rada/downloads.html

[10] Francis, W. N., Towell, G., Voorhees, E. M. Towards Building Contextual Representations of Word Senses Using Statistical Models. In: *Procs. of Workshop on the Acquisition of Lexical Knowledge from Text* (1993).

[11] Lesk, M. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In: *Procs. of SIGDOC* (1986).

[12] Vasilescu, F., Langlais, P., Lapalme, G. Evaluating Variants of the Lesk Approach for Disambiguating Words. In: *Procs. of LREC* (2004).

[13] Kilgarriff, A., Rosenzweig, J. English SENSEVAL: Report and Results. In: *Procs of LREC* (2000).

[14] Banerjee, S., Pedersen, T. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In: *Lecture Notes in Computer Science*, 2276:136-145 (2002).

[15] Porter, M. F. An Algorithm for Suffix Stripping. In: *Program: Electronic Library and Information Systems*, 40 (3), pp. 211-218 (2006).

[16] Kennedy, W. G., Trafton, J. G. Long-Term Symbolic Learning. In: *Cognitive Systems Research*, 8 (3), pp. 237-247 (2007).

[17] Nuxoll, A. M., Laird, J. E., James, M. Comprehensive Working Memory Activation in Soar. In: *Procs. of ICCM* (2004).