OntoNotes: Sense Pool Verification Using Google N-gram and Statistical Tests

Liang-Chih Yu[†*], Chung-Hsien Wu[*], Andrew Philpot[†], and Eduard Hovy[†]

[†] Information Sciences Institute, University of Southern California,
4676 Admiralty Way, Marina del Rey, CA 90292
{liangchi, philpot, hovy}@isi.edu
[*] Department of Computer Science and Information Engineering,
National Cheng Kung University,
No. 1, Ta-Hsueh Road, Tainan,Taiwan, ROC.
{lcyu, chwu}@csie.ncku.edu.tw

Abstract. The OntoNotes project has developed a methodology for producing a large multilingual corpus with annotation of predicate-argument structure, word senses, ontology linking, and coreference. The underlying semantic model of OntoNotes involves word senses that are grouped into so-called sense pools, i.e., sets of near-synonymous senses of words. Such information is useful for many applications, including query expansion for information retrieval (IR) systems, (near-)duplicate detection for text summarization systems, and alternative word selection for writing support systems. Once senses have been created and verified by annotation, sense pools are formed by an expert. Verification of sense pools is the topic of this paper. This paper describes a two-stage framework that combines machine and human verification of sense pools. The machine verification acts as a filter to select candidate pool members based on n-gram frequencies obtained from Google and subjected to appropriate statistical measures. The remaining candidates are then passed to humans for final verification. Our experimental results demonstrate that the machine verification can save much human verification work and thus facilitate the development of sense pools.

Keywords: Lexical semantics, word sense disambiguation, sense pool verification, ontology linking

1 Introduction

Richly annotated corpora are useful resources for many natural language applications. Various machine learning algorithms can then be trained on these corpora to improve the applications' effectiveness. The OntoNotes [1] project has developed a methodology for producing a large multilingual corpus with annotation of predicate-argument structure, word senses, ontology linking, and coreference[1]. In word sense

[1] Year 1 of the OntoNotes corpus has been released by Linguistic Data Consortium (LDC) (http://www.ldc.upenn.edu) in early 2007. The Year 2 corpus will be released in late 2007.

creation, sense creators generate sense definitions by grouping fine-grained sense distinctions obtained from WordNet and dictionaries into more coarse-grained senses. There are two reasons for this grouping instead of using WordNet senses directly. First, people have trouble distinguishing many of the WordNet-level distinctions in real text, and make inconsistent choices; thus the use of coarse-grained senses can improve inter-annotator agreement (ITA) [2], [3]. Second, improved ITA enables machines to more accurately learn to perform sense tagging automatically. Sense grouping in OntoNotes has been calibrated to ensure that ITA averages at least 90%.

But creating senses alone is not usually enough for many NLP applications. Knowing that the word "arm" has (at least) the senses *weapon* and *bodypart* enables systems to perform word sense disambiguation. But knowing in addition that the weapon sense of "arm" corresponds to the weapon sense of "weapon" and of "arsenal" means that systems can in addition perform term expansion (for IR) [4], [5], [6], (near-)duplicate detection (for summarization), alternative word selection (for writing support systems) [7], [8], [9], and so on.

In OntoNotes we call such collections of (near-)synonymous word senses (which correspond to WordNet synsets) *sense pools*. Based on the sense definitions, and using thesauri and other resources, OntoNotes specialists assign each word sense to a sense pool by combining each sense of a word in the OntoNotes corpus with the senses of other OntoNotes words that carry similar meaning. Finally, each sense pool (and hence its corresponding senses) are linked to a concept node in the Upper Model of the Omega ontology (http://omega.isi.edu) [10]. Omega is a 120,000-node terminological ontology constructed by merging a variety of resources, including WordNet [11], [12], Mikrokosmos [13], [14], and some upper models such as SUMO [15] and DOLCE [16]. Table 1 shows the concept nodes, sense definitions, and sense pools for the word "arm" (noun sense).

Table 1. OntoNotes senses and sense pools. The WordNet version is 2.1.

Omega concept	OntoNotes sense	WordNet sense	Sense pool
The forelimb of an animal	arm_1	WN.1	{arm_1, limb_1}
A weapon	arm_2	WN.2	{arm_2, weapon_1, arsenal_1}
A subdivision or branch of an organization	arm_3	WN.3	{arm_3, branch_1}
A projection, a narrow extension of a structure	arm_4	WN.4 WN.5	{arm_4, limb_4, branch_5}

A proposed sense pool needs to be verified. We do this by checking whether a sense in a pool can be substituted by other senses in the same pool without (too much) loss of semantic meaning. This process is useful for sense-pool creators to refine pools and improve their quality. Consider the following noun sense pool and sentence.

Pool: {bridge_1, overpass_1, viaduct_1, tunnel_1}

Sentences:

(1a) The **tunnel** under the bay is closed because of an accident.

(1b) The railroad passes under the mountain through a **tunnel**.

(1c) They are building a railroad **tunnel** right through the mountain.

In the above example, the elements of the pool share the meaning of a physical structure that connects separate places by traversing an obstacle. The verification process can then verify each substitution pair, e.g., (tunnel_1, bridge_1), in the pool based on the sentences annotated with the target sense "tunnel_1". Obviously, all the substitutions of "bridge" for "tunnel" above are semantically implausible. Therefore, the sense-pool creators can refine the pool according to the substitution information.

The verification process can be performed either by human experts or machine learning algorithms. This study takes both approaches to developing a two-stage framework that adopts the machine verification as a filter to save human verification work. Fig. 1 presents the two-stage framework.

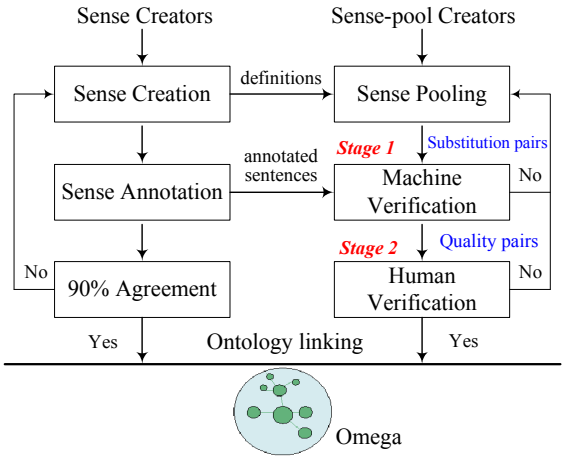

Fig. 1. Two-stage framework for sense pool verification.

The left part of Fig. 1 shows the sense creation process, which is detailed in [1]. In the right part of Fig. 1, the machine verification first takes as input each substitution pair of the pools and a set of sentences annotated with the target senses. For each pair, the n-grams that contain the target sense of the pair are extracted from the annotated sentences. The target sense in each n-gram is then substituted by the replacing sense. The frequencies of these n-grams are then obtained from Google to calculate the substitution score for the pair. The score is evaluated using a statistical test to determine if the target sense can be substituted by the replacing sense. Only the pairs accepted by the statistical test, called *quality pairs*, are passed to humans for final verification. The rest is returned to sense-pool creators for further refinement. Finally, the senses and their pools are store in the Omega ontology.

The rest of this work is organized as follows. Section 2 describes the principles for sense pooling and ontology linking. Section 3 describes the framework for sense pool verification, including machine and human verification, respectively. Section 4 summarizes the experimental results. Conclusions are finally drawn in Section 5.

2 Sense Pooling and Ontology Linking

OntoNotes specialists create a sense pool by the following procedure:
(1) Select a noun (or verb) and identify one of its senses as the target sense. Use a thesaurus to identify a set of words (typically between 3 and 20) that have at least one sense synonymous with the target sense.
(2) Create a new pool for the target sense if it is not already in an existing pool.
(3) Take each word in the synonym set and consider their every sense, deciding whether it belongs in the new pool, should be included in some other existing pool, or should be pooled in a new pool later.
(4) Repeat step **(3)** until all words in the synonym set have their relevant senses assigned into a pool.

Once a sense pool has been created, it is verified (as discussed in detail below). Verification can either accept the sense pool without changes, indicate the need to drop a sense, or suggest that the whole pool is inconsistent. In the latter two cases, experts perform changes and resubmit the changed pool for renewed verification.

When a sense pool is accepted, it is added into version 5 of the Omega ontology [10]. The Omega ontology is a long-term project whose function is to capture and organize semantically-anchored terms to support NL processing. Version 5 uses as terms (i.e., concepts) the verified sense pools. Omega contains an Upper Model of some 200 concepts that provide a very high-level categorization of the Objects, Events, and Properties encountered in (text about) the world.

Inserting a sense pool into Omega is accomplished by locating the most specific Upper Model concept that logically subsumes the pool's meaning. An easy example is the pool that denotes "car/automobile", which is inserted under *artifact*. A more difficult example is the pool denoting "teacher", which is inserted under *person*, and to which the Role "operator" relation is attached, to link "teacher" also to *profession*. In addition to Role, other operator relations are Part, Function, and Location.

3 Sense Pool Verification

3.1 Machine Verification

The main steps of the machine verification are described as follows.

(1) Input data: Given a sense pool, all possible substitution pairs are first generated for verification. For instance, the pool {a, b, c} will produce 6 substitution pairs, i.e., (a, b), (b, a), (a, c), (c, a), (b, c), (c, b).
Each pair has the format:

(target-sense_senseID, replacing-sense_senseID)

Additionally, a set of sentences annotated with the target sense is also selected from the OntoNotes corpus.

(2) N-gram generation: For each pair, the n-grams (bigram to 4-gram) with the target sense are generated from the annotated sentences. The target sense in each n-gram is then substituted by the replacing sense. The columns A and E of Table 2 list the n-grams with the target sense "tunnel_1" and replacing sense "overpass_1", respectively, generated from sentence (1a), (1b), and (1c).

Table 2. Example of machine verification with the pool (tunnel_1, overpass_1) and sentences (1a)—(1c)

Target sense: tunnel_1				Replacing sense: overpass_1				
A	B	C	D	E	F	G	H	I
N-gram	freq (ngram)	freq (tunnel)	Z (B/C)	N-gram	freq (ngram)	freq (overpass)	Z (F/G)	Score (H/D)
the tunnel	1,670,000	66,200,000	0.025	the overpass	568,000	3,190,000	0.178	1.000
tunnel under	789,000	66,200,000	0.012	overpass under	1,950	3,190,000	0.001	0.051
a tunnel	1,260,000	66,200,000	0.019	an overpass	368,000	3,190,000	0.115	1.000
railroad tunnel	115,000	66,200,000	0.002	railroad overpass	88,000	3,190,000	0.028	1.000
tunnel right	30,800	66,200,000	0.000	overpass right	3,110	3,190,000	0.001	1.000
							$Score_{bigram}$(tunnel_1, overpass_1)	**0.763**
the tunnel under	88,400	66,200,000	0.001	the overpass under	318	3,190,000	0.000	0.075
tunnel under the	390,000	66,200,000	0.006	overpass under the	107	3,190,000	0.000	0.006
through a tunnel	458,000	66,200,000	0.007	through an overpass	866	3,190,000	0.000	0.039
a railroad tunnel	15,100	66,200,000	0.000	a railroad overpass	16,200	3,190,000	0.005	1.000
railroad tunnel right	9	66,200,000	0.000	railroad overpass right	23	3,190,000	0.000	1.000
tunnel right through	3,250	66,200,000	0.000	overpass right through	4	3,190,000	0.000	0.026
							$Score_{trigram}$(tunnel_1, overpass_1)	**0.358**
the tunnel under the	546,000	66,200,000	0.001	the overpass under the	64	3,190,000	0.000	0.024
tunnel under the bay	21,700	66,200,000	0.000	overpass under the bay	0	3,190,000	0.000	0.000
mountain through a tunnel	112	66,200,000	0.000	mountain through an overpass	0	3,190,000	0.000	0.000
building a railroad tunnel	319	66,200,000	0.000	building a railroad overpass	202	3,190,000	0.000	1.000
a railroad tunnel right	2	66,200,000	0.000	a railroad overpass right	2	3,190,000	0.000	1.000
railroad tunnel right through	2	66,200,000	0.000	railroad overpass right through	0	3,190,000	0.000	0.000
tunnel right through the	434	66,200,000	0.000	overpass right through the	1	3,190,000	0.000	0.048
							$Score_{4gram}$(tunnel_1, overpass_1)	**0.296**

(3) Google queries: For each n-gram with the target sense and replacing sense, their frequencies can be obtained by querying Google (columns B and F). The frequency is then normalized as

$$Z(ngram^i_{target}) = \frac{freq(ngram^i_{target})}{freq(target)},$$
$$Z(ngram^j_{replacing}) = \frac{freq(ngram^i_{replacing})}{freq(replacing)}, \quad (1)$$

where $Z(ngram^i)$ denotes the normalization values for an n-gram with the target sense or replacing sense (columns D and H); $freq(ngram^i)$ denotes the frequency of an n-gram with the target sense or replacing sense, and $freq(\bullet)$ denotes the frequency of the target sense or replacing sense (columns C and G).

(4) Substitution score: The substitution score for an n-gram is defined as follows.

$$Score(ngram^i) = \frac{Z(ngram^j_{replacing})}{Z(ngram^i_{target})}. \quad (2)$$

This score (column I) is used to measure the degradation of the normalization value after the substitution of the target sense. A greater score indicates a lower degradation level, which means that these two senses are more substitutable in the context of the given n-gram. Additionally, the upper bound of the score is restricted to 1 for simplicity consideration. The substitution score for a pair can then be calculated as the average of n-gram scores (bigram, trigram or 4-gram), as shown below.

$$Score_{ngram}(pair) = \frac{\sum_{i=1}^{N} Score(ngram^i)}{N}, \quad (3)$$

where N denotes the number of n-grams. In the remainder of this paper, we report only 4-gram scores to represent the substitution scores for the pairs, since they are more discriminative than bigram and trigram scores.

(5) Randomized score: Although the substitution scores indicate the degradation levels, for instance, $Score_{4gram}$(tunnel_1, overpass_1) = 0.296, it is difficult to decide if the scores are significantly higher, since they may vary with different substitutions. One possible solution is the *thresholding* method, but choosing a suitable threshold for each pair is still nontrivial. To address the problem, this study proposes the use of *randomized scores*. A randomized score for a pair is calculated by randomly selecting k semantically unrelated senses from the OntoNotes corpus as the replacing senses,

thus producing k scores (by steps (2)—(4)) for the random pairs. The randomized score is then calculated by averaging the k scores. Therefore, each pair has its own randomized score, representing the baseline value for the substitution. Table 3 shows an example of the randomized score (k=3) for the input pair (tunnel_1, overpass_1).

Table 3. Example of randomized score computation. The replacing senses *transportation_1*, *problem_1* and *capital_1* are randomly selected from the OntoNotes corpus.

Input pair	Substitution score	Random pair	Substitution score	Randomized Score
(tunnel_1,overpass_1)	0.296	(tunnel_1,transportation_1)	0.174	0.171
		(tunnel_1,problem_1)	0.181	
		(tunnel_1,capital_1)	0.157	

(6) Quality pair selection: Each pair is associated with a substitution score and a randomized score. The quality pairs are then selected by comparing these two scores. The strategies are described as follows.

Strategy 1: Selection with difference: A pair is called a quality pair if and only if the substitution score is greater than the randomized score.

Strategy 2: Selection with statistical test: A pair is called a quality pair if and only if the substitution score is significant greater than the randomized score. An independent *t-test* with confidence level 0.95 is used to determine if the difference of the two scores is statistically significant.

3.2 Human Verification

It was our intention to avoid human verification altogether. However, even 4-grams included some examples that are neutral enough to accept semantically unrelated replacement words, for example, "the old [tunnel/capital/transportation/problem] was", causing acceptability scores above threshold. For this reason, and to evaluate the effectiveness of the automated procedure, we implemented human verification.

Human verification evaluates each quality pair in the following steps. First, a set of sentences annotated with the target sense is selected, and then the target sense in each sentence is substituted by the replacing sense. These after-substitution sentences along with the definitions of the target sense and replacing sense are presented to a native English speaker for verification. The verifier examines the substitutions in the sentences with meaning equivalence (not fluency), and then rates them by the following criteria.

Positive-1: Exactly equivalent.

Positive-2: Essentially equivalent.

Negative-1: Mostly equivalent, but differ in one important aspect.

Negative-2: Not really equivalent.

Neutral: Don't know.

The verifier is instructed to ignore disfluency and syntactic awkwardness, and to pay attention only to semantic substitutability. Each sentence is associated with one of the five ratings. For each quality pair, if the number of sentences with positive ratings is greater than those with negative ratings, then the pair is accepted, otherwise it will be returned to sense-pool creators for refinement. If the two numbers are equal, then this pair will be judged by an adjudicator.

4 Experimental Results

4.1 Experiment setup

This experiment focuses on the verification for sense pools derived from noun senses. A total of 139 noun sense pools were selected, which produced 517 substitution pairs for verification. The annotated sentences were selected from the English corpus of OntoNotes v1.0, i.e., Wall Street Journal (WSJ). Table 4 presents the statistics for the experimental data.

Table 4. Statistics of experimental data.

Avg. number of senses per pool	2.53
Min. number of senses per pool	2
Max. number of senses per pool	7
Avg. number of sentences per pair	32
Min. number of sentences per pair	6
Max. number of sentences per pair	50

The maximum number of annotated sentences that could be selected for each pair (target sense) was restricted to 50 to limit computational complexity. Additionally, the pairs with 5 or fewer sentences were excluded from the experiment, since a smaller number of sentences may produce an unstable substitution score. Therefore, a total of 177 pairs were evaluated in this experiment.

In machine verification, the substitution scores and randomized scores were calculated using steps (1)—(5) described in Section 3.1. Additionally, the randomized scores were generated by randomly selecting 3 noun senses from the corpus. Both strategies for quality pair selection, i.e., selection with difference and statistical test, were implemented for comparison. In the human verification, the quality pairs were evaluated by presenting 7 sentences to a native English speaker who is not involved in the sense-pool creation.

The evaluation metric *accuracy* was used to compare the performance of the strategies for quality pair selection. The accuracy measures the percentage of the pairs correctly accepted and correctly rejected in the machine verification, defined by Eq. (4). Additionally, both strategies were considered as a filter to facilitate the human

verification process, thus the *saving* rate and *missing* rate were adopted as the evaluation metrics. The saving rate measures the percentage of the pairs rejected in the machine verification, defined by Eq. (5). However, some of the rejected pairs may be accepted in the human verification, and such false rejection can be reflected by the missing rate, defined by Eq. (6).

$$accuracy = \frac{\text{number of pairs correctly accepted and rejected by machine}}{\text{total number of pairs}}. \quad \textbf{(4)}$$

$$saving = \frac{\text{number of pairs rejected by machine}}{\text{total number of pairs}}. \quad \textbf{(5)}$$

$$missing = \frac{\text{number of pairs rejected by machine but accepted by human}}{\text{total number of pairs}}. \quad \textbf{(6)}$$

4.2 Results

Table 5 shows the results of machine and human verification for "arm" (pairs 1–7) and "corporation" (pairs 8–11) families. Some of the pairs, e.g., (limb_1, arm_1), were not verified due to the lack of annotated sentences. The selection strategy *Difference* correctly accepted 6 pairs (1, 2, 3, 6, 8 and 9), correctly rejected 2 pairs (5 and 10), and missed 2 pairs (4 and 11), while the *t-test* correctly accepted 5 pairs (2, 3, 6, 8 and 9), correctly rejected 3 pairs (5, 7, and 10), and missed 3 pairs (1, 4 and 11).

Table 5. Results of machine and human verification. The columns QP and Human indicate whether an input pair is a quality pair or not.

No	Input pair	# of sentences	Subs. Score	Rand. Score	Difference	QP	t-test (p-value)	QP	Human
1	(arm_1,limb_1)	11	0.159	0.072	0.087	Y	0.113	N	Y
2	(arm_2,weapon_1)	23	0.315	0.053	0.262	Y	0.000	Y	Y
3	(weapon_1,arm_2)	38	0.097	0.026	0.071	Y	0.000	Y	Y
4	(arm_2,arsenal_1)	23	0.032	0.053	(0.021)	N	—	N	Y
5	(weapon_1,arsenal_1)	38	0.012	0.026	(0.014)	N	—	N	N
6	(arm_3,branch_1)	17	0.474	0.131	0.343	Y	0.000	Y	Y
7	(branch_1,arm_3)	50	0.119	0.097	0.022	Y	0.376	N	N
8	(corporation_1,company_1)	50	0.624	0.190	0.434	Y	0.000	Y	Y
9	(company_1,corporation_1)	50	0.135	0.056	0.079	Y	0.000	Y	Y
10	(corporation_1,syndicate_2)	50	0.041	0.190	(0.149)	N	—	N	N
11	(company_1,syndicate_2)	50	0.005	0.056	(0.051)	N	—	N	Y

P-value < 0.05 represents the difference is statistically significant

Table 6 presents the accuracy, saving rate and missing rate of both strategies. Of the 177 test pairs, *Difference* correctly accepted 50 pairs and correctly rejected 37 pairs, thus yielding 49.15% accuracy. Additionally, *Difference* rejected a total of 50 pairs, of which 13 pairs were the missing cases, which means that it can save 28.25% (50/177) work of human verification with 7.34% (13/177) missing rate. The performance of *Difference* was low because it accepted more pairs that should be rejected (77/177). The *t-test* instead compared the substitution scores and randomized scores with significance tests, but not just the difference of the scores, thus it can correctly reject more pairs. The results show that the *t-test* yielded a higher accuracy and saving rate than did *Difference*, but the missing rate also increased accordingly.

Table 6. Performance of strategies for quality pair selection.

	Difference				t-test			
Accuracy	49.15%				63.84%			
Saving	28.25%				57.63%			
Missing	7.34%				14.69%			
	Human				Human			
		True	False			True	False	
Machine	True	50	77	127	True	37	38	75
	False	13	37	50	False	26	76	102
		63	114	177		63	114	177

The choice of strategy may depend on the application domain. For our application, the statistical approach can save much work of human verification with a higher accuracy. The weak filter (*Difference*) with a lower missing rate is also acceptable since human verification is applied in the later stage for final decision. For many other applications such as information retrieval and writing support systems, human intervention is often not possible or practical. In this circumstance, the statistical approach is more useful because it can automatically suggest more correct substitution information for query expansion and alternative word selection.

5 Conclusion

Ontology construction can proceed along several different principles and methodologies [17]. This paper describes a method that includes a step of multi-person verification, which provides a level of validity and replicability to the resulting ontology that is, unfortunately, not as common as it should be.

Verification follows a two-stage framework, machine and human verification, for sense pool verification. The Google n-gram frequencies are used for measuring the degradation of substitutions. A statistical test can then be applied to determine whether the degradation is statistically significant or not. Experimental results show that the machine verification can save the work of human verification and thus facilitate the development of sense pools.

Future work will focus on three directions. First, a new corpus will be annotated to evaluate more sense pools. Second, more significant features such as predicate-argument structure will be combined with n-grams to boost verification performance. Finally, pairwise substitutions will be extended to investigate the transitive property of substitution; that is, to evaluate if (a, c) is true when both (a, b) and (b, c) are true.

References

1. Hovy, E.H., Marcus, M., Palmer, M., Ramshaw, L., Weischedel, R.: OntoNotes: The 90% Solution. In: Proceedings of HLT/NAACL-06, pp. 57–60 (2006)
2. Palmer, M., Babko-Malaya, O., Dang, H.T.: Different Sense Granularities for Different Applications. In: Second Workshop on Scalable Natural Language Understanding Systems at HLT/NAACL-04 (2004)
3. Palmer, M., Dang, H. T., Fellbaum, C.: Making Fine-grained and Coarse-grained Sense Distinctions, Both Manually and Automatically. Journal of Natural Language Engineering. 13, 137–163 (2006)
4. Yu, L.-C., Wu, C.-H, Lin, C.-Y., Hovy, E.H., Lin, C.-L.: Topic Analysis for Psychiatric Document Retrieval. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07), pp. 1024–031 (2007)
5. Wu, C.-H., Yeh, J.-F., Lai. Y.-S.: Semantic Segment Extraction and Matching for Internet FAQ Retrieval. IEEE Trans. Knowledge and Data Engineering. 18, 930–940 (2006)
6. Wu, C.-H., Yeh, J.-F., and Chen., M.-J.: Domain-Specific FAQ Retrieval Using Independent Aspects. ACM Trans. Asian Language Information Processing. 4, 1–17 (2005)
7. Inkpen, D. and Hirst, G.: Building and Using a Lexical Knowledge-base of Near-Synonym Differences. Computational Linguistics. 32, 1–39 (2006)
8. Inkpen, D.: Near-Synonym Choice in an Intelligent Thesaurus. In: Proceedings of NAACL/HLT-07, pp. 356–363 (2007)
9. Inkpen, D.: A Statistical Model for Near-Synonym Choice. ACM Trans. Speech and Language Processing. 4, 1–17 (2007)
10. Philpot, A., Hovy, E.H., Pantel, P.: The Omega Ontology. In: Proceedings of OntoLex-05, pp. 59–66 (2005)
11. Miller, G.: WordNet: An Online Lexical Database. International Journal of Lexicography. 3, 235–312 (1990)
12. Fellbaum, C.: WordNet: An Electronic Lexical Data-base. MIT Press, Cambridge, MA. (1998)
13. Mahesh, K.: Ontology Development for Machine Translation: Ideology and Methodology. Technical Report MCCS 96-292, Computing Research Laboratory, New Mexico State University (1996)
14. O'Hara, T., Mahesh, K., Nirenburg, S.: Lexical Acquisition with WordNet and the Mikrokosmos Ontology. In: Proceedings of the Workshop on Usage of Word-Net in Natural Language Processing Systems at COLING/ACL-98, pp. 94–101 (1998)
15. Niles, I., Pease, A.: Towards a Standard Upper Ontology. In: Second International Conference on Formal Ontology in Information Systems (FOIS-01), pp. 2–9 (2001)
16. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening Ontologies with DOLCE. In: 13th International Conference on Knowledge Engineering and Knowledge Management, pp. 166–181 (2002)
17. Hovy, E.H. Methodologies for the Reliable Construction of Ontological Knowledge. In F. Dau, M.-L. Mugnier, and G. Stumme (eds), Conceptual Structures: Common Semantics for Sharing Knowledge. In: Proceedings of the 13th Annual International Conference on Conceptual Structures (ICCS 2005). Springer Lecture Notes in AI 3596, pp. 91–106 (2005)