Interpolated PLSI for Learning Plausible Verb Arguments *

Hiram Calvo[a, b], Kentaro Inui[a], and Yuji Matsumoto[a]

[a]Computational Linguistics, Nara Institute of Science and Technology,
Takayama, Ikoma, Nara 630-0192, Japan
{calvo, inui, matsu}@is.naist.jp

[b]Artificial Intelligence, Center for Computing Research, National Polytechnic Institute,
Mexico City, DF, 07738, Mexico
hcalvo@cic.ipn.mx

Abstract. Learning Plausible Verb Arguments allows to automatically learn what kind of activities, where and how, are performed by classes of entities from sparse argument co-occurrences with a verb; this information it is useful for sentence reconstruction tasks. Calvo *et al.* (2009b) propose a non language-dependent model based on the Word Space Model for calculating the plausibility of candidate arguments given one verb and one argument, and compare with the single latent variable PLSI algorithm method, outperforming it. In this work we replicate their experiments with a different corpus, and explore variants to the PLSI method in order to explore further capabilities of this latter widely used technique. Particularly, we propose using an interpolated PLSI scheme that allows the combination of multiple latent semantic variables, and validate it in a task of identifying the real dependency-pair triple with regard to an artificially created one, obtaining up to 83% recall.

Keywords: Plausible Verb Arguments, *K*-Nearest Neighbors algorithm, KNN, Distributional Thesaurus, Probabilistic Latent Semantic Indexing, PLSI.

1 Introduction

Plausible Verb Arguments information is helpful in sentence reconstruction tasks. For example: The boy plays with the ____ in the ____; A _____ eats grass; and I drank _____ in a glass. Several tasks have to deal with this common problem, for example, anaphora resolution would consist on finding the referenced objects: The boy plays with *it there*, *It* eats grass, I drank *it* in a glass. Information Retrieval applications look for answers to 5W questions such as 'Who eats grass?', "Where?", "When?" (Parton *et al.*, 2009). The answers to these questions are not always explicitly stated in a text, such as 'Where do boys play usually using what?', 'What do boys usually play with?', 'What is usually drunk in a glass?' therefore, additional common sense knowledge is needed to answer these questions. Our goal is to create a large database of this information so that plausibility can be tested for performing a variety of tasks. For other tasks that can use this kind of information, see Section 1.1.

This problem can be seen as collecting a large database of semantic frames with detailed categories and examples that fit these categories. For this purpose, recent works take advantage

* We thank the support of the Japanese Government and the Mexican Government (SNI, SIP-IPN, COFAA-IPN, and PIFI-IPN). Second author is a JSPS fellow. We also thank our anonymous reviewers for their useful comments and discussion.

of existing manually crafted resources such as WordNet, Wikipedia, FrameNet, VerbNet or PropBank. For example, Reisinger and Paşca (2009) annotate existing WordNet concepts with attributes, and extend *is-a* relations, based on Latent Dirichlet Allocation on Web documents and Wikipedia. Yamada *et al.* (2009) explore extracting hyponym relations from Wikipedia using pattern-based discovery and distributional similarity clustering. Nevertheless, the approach of using handcrafted resources prevents from obtaining information for languages where those resources do not exist.

Calvo *et al.* (2009a, 2009b) propose a non language-dependent model based on K-Nearest Neighbors for calculating the plausibility of candidate arguments given one verb and one argument, and compare with the traditional PLSI method, outperforming it. In this work we replicate their experiments with a different corpus, and explore variants to the PLSI method in order to attest better capabilities for this latter widely used technique. Particularly, we propose using an interpolated PLSI scheme that allows the combination of multiple latent semantic variables.

In Section 2 we present three different models for plausible argument estimation, the K-Nearest Neighbors Model proposed by Calvo *et al.* (2009b) is presented in Section 2.1; the single-variable PLSI model is presented in Section 2.2; and our proposal of multiple latent semantic variable based on interpolation is presented in Section 2.3. Then we present several experiments: in Section 3.1 and 3.2 we replicate previous experiments with a different corpus, being the latter section an analysis of the learning rate. Previous results are heavily affected by pre-filtering, as we show in experiments shown in Section 3.3; our proposal model overcomes this and results are shown in Section 3.4. Afterwards, we explore how our model works with n-grams instead of dependency triple relationships in experiments from Section 3.5. Finally, we draw our conclusions in Section 4 mentioning future work and possible applications.

1.1 Possible applications

Correct Plausible Verb Argument identification can be used for several tasks, such as improving parsing. Since latent variables group the kind of arguments expected for a sentence, it is possible to infer the meaning of unknown words, as in the well-known example about *tezguino* (Lin, 1998b) where it is possible to know what is tezguino from the sentences: *A bottle of tezguino is on the table*; *Everybody likes tezguino*; *Tezguino makes you drunk*; and *We make tezguino out of corn*. Another application is Semantic Role Labeling, since grouping verb arguments and measuring their plausibility increases performance, as shown by Merlo and Van Der Plas (2009) and Deschacht and Moens (2009). Some other applications are metaphora recognition, since we are able to know common usages of arguments, and an uncommon usage would suggest its presence, or a coherence mistake (*v. gr. to drink the moon in a glass*). Malapropism detection can use the measure of the plausibility of an argument to determine misuses of words (Bolshakov, 2005) as in *hysteric* center, instead of historic center; *density* has brought me to you; It looks like a *tattoo* subject; and Why you say that with *ironing*?.

2 Models for Plausible Argument Estimation

We explore the models proposed in Calvo *et al.* (2009a, 2009b); then, we propose a new model called interpolated PLSI, which allows using multiple latent semantic variables.

We can regard the task of finding the plausibility of a certain argument for a set of sentences as estimating a word given a specific context. Since we want to consider argument co-relation, we want to estimate $P(v,r_1,n_1,r_2,n_2)$ where v is a verb, r_1 is the relationship between the verb and n_1 (noun) as subject, object, preposition or adverb. r_2 and n_2 are analogous. If we assume that n has a different function when used with another relationship, then we can consider that r and n make a new symbol, called a. So that we can express the 5−tuple $P(v,r_1,n_1,r_2,n_2)$ as $P(v,a_1,a_2)$.

We want to know, given a verb and an argument a_1, which a_2 is the most plausible argument, *i.e.* $P(a_2|v,a_1)$. The probability of finding a particular verb and two of its syntactic relationships can be expressed as:

$$P(v,a_1,a_2) = P(v,a_1) \cdot P(a_2|v,a_1), \qquad (1)$$

which can be estimated in several ways.

2.1 K-Nearest Neighbors Model

Uses the *k* nearest neighbors of each argument to find the plausibility of an unseen triple given its similarity to all triples present in the corpus, measuring this similarity between arguments. See Figure 1 for the pseudo-algorithm of this model.

```
for each triple $<v,a_1,a_2>$ with observed count c,
 for each argument $a_1,a_2$
  Find its k most similar words $a_{1s1}…a_{1sk}$,  $a_{2s1}…a_{2sk}$
   with similarities $s_{1s1}$, ..., $s_{1sk}$ and $s_{2s1}$,...,$s_{2sk}$.
  Add votes for each new triple $<v,a_{1si},a_{2sj}>$ += $c \cdot s_{1s}i \cdot s_{2sj}$
```

Figure 1: Pseudo-algorithm for the K-nearest neighbors DLM algorithm

As votes are accumulative, triples that have words with many similar words will get more votes.

Common similarity measures range from Euclidean distance, cosine and Jaccard's coefficient (Lee, 1999), to measures such as Hindle's measure and Lin's measure (Lin, 1998a). Weeds and Weir (2003) show that the distributional measure with best performance is the Lin similarity, so this measure is used for smoothing the co-occurrence space, following the procedure as described by Lin (1998a).

2.2 PLSI – Probabilistic Latent Semantic Indexing

The probabilistic Latent Semantic Indexing Model (PLSI) was introduced in Hofmann (1999), arose from Latent Semantic Indexing (Deerwester *et al.*, 1990). The model attempts to associate an unobserved class variable $z \in Z = \{z_1, ..., z_k\}$, (in our case a generalization of correlation of the co-occurrence of *v,a_1* and *a_2*), and two sets of observables: arguments, and verbs+arguments. In terms of generative model it can be defined as follows: a *v,a_1* pair is selected with probability *$P(v,a_1)$*, then a latent class *z* is selected with probability *$P(z|v,a_1)$* and finally an argument *a_2* is selected with probability *$P(a_2|z)$*. Calvo *et al.* (2009a) propose using PLSI (Hoffmann, 1999) this said way, expressed also as (2).

$$P(v,a_1,a_2) = \sum_{Z} P(z_i) \cdot P(a_2|z_i) \cdot P(v,a_1|z_i) \qquad (2)$$

z is a latent variable capturing the correlation between *a_2* and the co-occurrence of (*v,a_1*) simultaneously. Using a single latent variable to correlate three variables may lead to a poor performance of PLSI, so that in next section we explore different ways of exploiting the smoothing by latent semantic variables.

2.3 iPLSI – interpolated PLSI

The previous PLSI formula originally used crushes the association of information from *a_2*, and *v,a_1* simultaneously into one single latent variable. This caused two problems: first, data sparseness, and second, it fixed the correlation between two variables. Hence we propose a variation for this calculation by using interpolation based on each pair of arguments for a triple.

The following formula shows an interpolated way of estimating the probability of a triple based on the co-occurrences of its different pairs.

$$P_E(v,a_1,a_2) \propto f_m(v,a_1)f(a_2) + f_n(v,a_2)f(a_1) + f_o(a_1,a_2)f(a_2) \qquad (3)$$

$$f_m(v,a_1) = \sum_m P(m_i) \cdot P(v|m_i) \cdot P(a_1|m_i)$$

$$f_n(v,a_2) = \sum_n P(n_i) \cdot P(v|n_i) \cdot P(a_2|n_i)$$

$$f_o(a_1,a_2) = \sum_o P(o_i) \cdot P(a_1|o_i) \cdot P(a_2|o_i)$$

where $f(v)$, $f(a_1)$, and $f(a_2)$ are the observed probabilities of v, a_1 and a_2 respectively.

Additionally we test a model that considers additional information. See Eq. (4). Note that a_i (the latent variable topics) should not be confused with a_1 and a_2 (the arguments).

$$P_E(v,a_1,a_2) \approx f_m(v,a_1)f(a_2) + f_n(v,a_2)f(a_1) + f_o(a_1,a_2)f(a_2) + f_a(v,a_1,a_2) + f_b(v,a_1,a_2) + f_c(v,a_1,a_2) \qquad (4)$$

$$f_a(v,a_1,a_2) = \sum_a P(a_i) \cdot P(v,a_2|a) \cdot P(a_1|a)$$

$$f_b(v,a_1,a_2) = \sum_b P(b_i) \cdot P(a_1,a_2|b_i) \cdot P(v|b_i)$$

$$f_c(v,a_1,a_2) = \sum_c P(c_i) \cdot P(v,a_1|c_i) \cdot P(a_2|c_i)$$

See the Figure 2 for a graphical representation of this concept. Each latent variable is represented by a letter in a small circle. Big circles surround the components of the dependency triple to be estimated. A black dot shows the co-occurrence of two variables. All of them contribute for the estimation of the triple v,a_1,a_2.

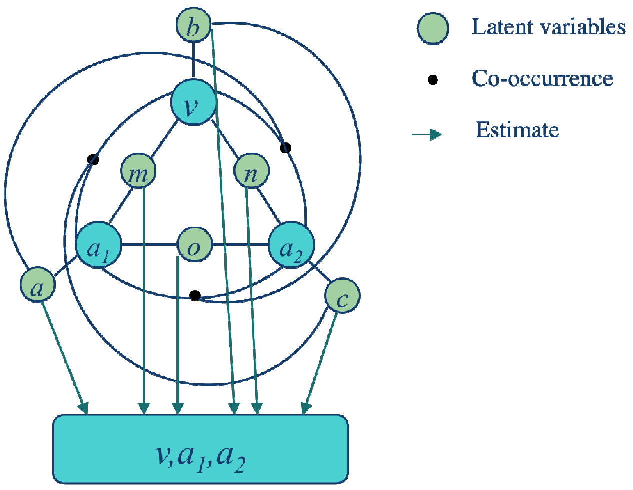

Figure 2: Graphical representation of iPLSI. The tuple v,a1,a2 is estimated by using latent variables based on pairs of two variables, and/or the pair of a variable and the co-ocurrence of two variables. See eqs. (3) and (4).

3 Experiments

We compare these two models in a pseudo-disambiguation task following Weeds and Weir (2003). First we obtain triples $\langle v, a_1, a_2 \rangle$ from the corpus. Then, we divided the corpus in training (80%) and testing (20%) parts. With the first part we trained the PLSI model. We use this part also for creating the distributional thesaurus used by the KNN model, *i.e.*, the similarity measure used for pairs of arguments a_2, a'_2. Then we are able to calculate *Plausibility* (v, a_1, a_2). For evaluation we created artificially 4-tuples: $\langle v, a_1, a_2, a'_2 \rangle$, formed by taking all the triples $\langle v, a_1, a_2 \rangle$ from the testing corpus, and generating an artificial tuple $\langle v, a_1, a'_2 \rangle$ choosing a random a'_2 with $r'_2 = r_2$, and making sure that this new random triple $\langle v, a_1, a'_2 \rangle$ was not present in the training corpus. The task consisted of selecting the correct tuple. Ties occur when both tuples are given the same score (and both are different from zero).

For these evaluations we used the UkWaC corpus (Ferraresi *et al.*, 2008.) This corpus is a large balanced corpus of English from the UK Web with more than 2 billion tokens[1]. We created two wordsets for the verbs: *play, eat, add, calculate, fix, read, write, have, learn, inspect, like, do, come, go, see, seem, give, take, keep, make, put, send, say, get, walk, run, study, need,* and *become*. These verbs were chosen as a sample of highly frequent verbs, as well as not so frequent verbs. They are also verbs that can take a great variety of arguments, such as *take* (i.e., ambiguity is high). Each wordset contains 1000 or 2500 verb dependency triples per each verb. The first wordset is evaluated against 5,279 verb dependency triples, while the second wordset is evaluated against 12,677 verb dependency triples, corresponding roughly to 20% of the total number of triples in each wordset.

3.1 Results of original algorithm with new corpus

In this section we present our results for this new corpus of the original PLSI and the KNN algorithms with the new corpus. Tests were carried out with one 7-topic variable for PLSI, and a 100 nearest neighbors expansion for KNN. Calvo *et al.* (2009b) have shown that for estimating the probability of an argument a_2, $P(a_2|v, a_1)$ works better than $P(a_2|v)$. The following table confirms this for different wordset sizes. These experiments were performed on a subcorpus of UKWaC made of 1000 or 2500 triples per verb for the verbs mentioned in Section 3.

Table 1: Results of the original PLSI and KNN algorithms for a test with the UKWaC corpus

Mode	Algorithm	Wordset size	Prec.	Recall	F-score
$P(a_2\|v)$	PLSI	1000	0.5333	0.2582	0.3479
	KNN	1000	0.7184	0.5237	0.6058
	PLSI	2500	0.5456	0.2391	0.3325
	KNN	2500	0.7499	0.5032	0.6023
$P(a_2\|v, a_1)$	PLSI	1000	0.4315	0.1044	0.1681
	KNN	1000	0.8236	0.5492	0.6590
	PLSI	2500	0.3414	0.0611	0.1036
	KNN	2500	0.8561	0.6858	0.7615

In almost cases KNN performs better than original PLSI in precision and recall (the best of the KNN variations is better than the best of the PLSI variations). Contrary to KNN, PLSI's performance increases as the wordset size is increased probably due to more confusion in using the same number of topics. This can be seen also in Figures 3 and 4: recall improves slightly for bigger data sets and more topics.

[1] A tool including queries to this corpus can be found at http://sketchengine.co.uk

3.2 Measuring the learning rate

This experiment consisted on gradually increasing the number of triples from 125 to 2000 dependency triples per verb to examine the effects of using smaller corpora. Results are shown in Figure 3. In this figure KNN outperforms PLSI when adding more data. KNN precision is higher as well in all experiments. The best results for PLSI were obtained with 7 topics, while for KNN the best results were obtained using 200 neighbors.

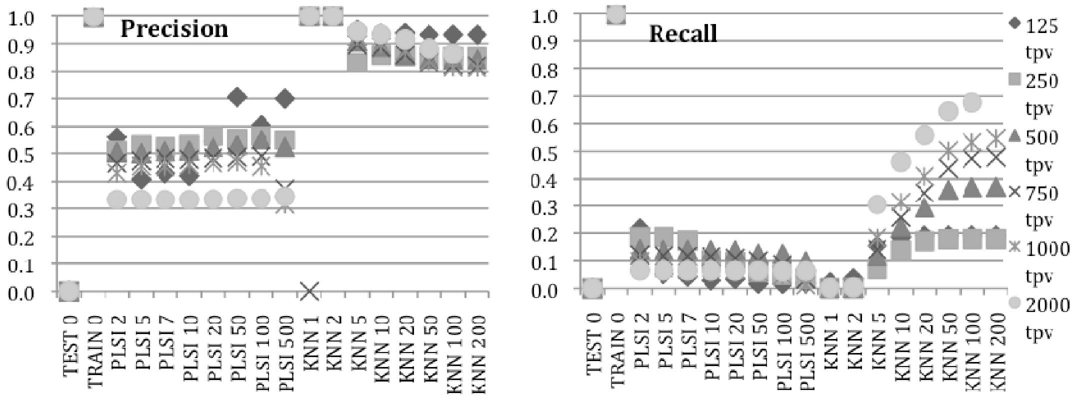

Figure 3: Precision and Recall for the original PLSI and KNN with learning rate (each series has different number of triples per verb, tpv). The frequency threshold for triples was set to 4. The numbers and the lower part show the number of topics for PLSI and the number of neighbors for KNN.

3.3 Results with no pre-filtering

Previous results used a pre-filtering threshold of 4, that is, triples with less than 4 occurrences were discarded. Here we present results with no pre-filtering. In **Figure** 4 results for KNN fall dramatically. PLSI is able to perform better with 20 topics. This suggests that PLSI is able to smooth better single occurrences of certain triples. KNN is better for working with frequently occurring triples. We require a method that can handle occurrences of un-frequent words, since pre-filtering implies a loss of data that could be useful afterwards. For example, consider that *tezgüino* is mentioned only once in the training test. We consider that it is important to be able to learn information for scarcely mentioned entities too. The next section presents results regarding to the improvement of using PLSI to handle non-filtered items.

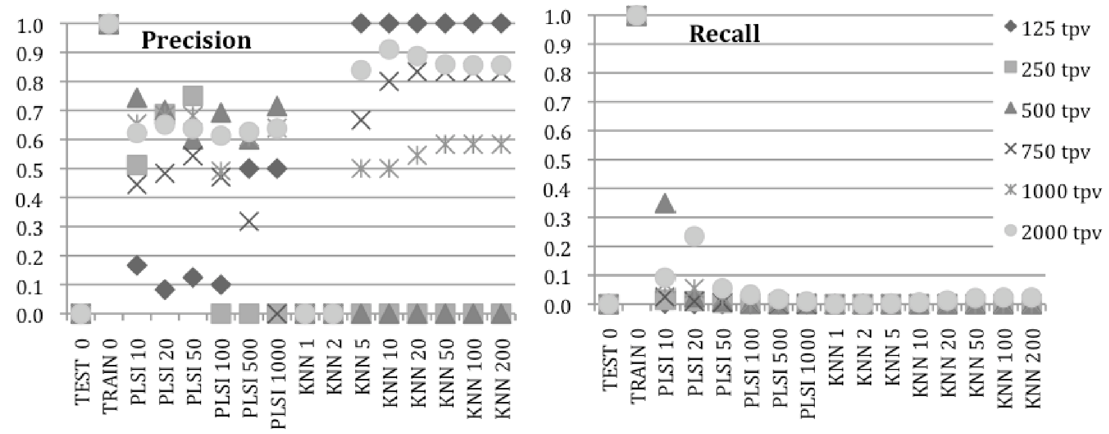

Figure 4: Average of Precision and Recall for the original PLSI and KNN showing learning rate (each series has different number of triples per verb, *tpv*). No frequency threshold was used. The numbers and the lower part show the number of topics for PLSI and the number of neighbors for KNN.

3.4 iPLSI results

As presented in Section 2.3, we test different models for combining the Latent Semantic Variables. The *mode* part shows the latent variables that were used for these tests. For example, for the *a,c* row, the estimation was carried using (5). Results are presented in Table 2.

$$P_E(v,a_1,a_2) \approx f_a(v,a_1,a_2) + f_c(v,a_1,a_2) \qquad (5)$$

In Table 2**,** the best results were obtained for *o,* (using only the information from a_1,a_2) followed by *m,o*, which is combining the information from v,a_1 and a_1,a_2. The *m,n,o* and *n,o* modes include *n*, which has no impact in this test because it is always fixed, and helps little for deciding which triple is better. However, as we show in the following section, a test with pure n-grams (non dependency triples, as in all previous tests) the three components (in this case *m*, *n*, and *o*), are contributing to the estimation.

Table 2: Comparison of different iPLSI modes, consisting on selecting different estimators. KNN is shown in the last row for reference.

mode	Precision	Recall	mode	Precision	Recall
a,b,c	0.78	0.78	m,n,o	0.83	0.83
a	0.67	0.60	m	0.78	0.77
b	0.44	0.44	n	0.50	0.48
c	0.77	0.77	o	0.84	0.84
a,c	0.62	0.62	m,n	0.77	0.77
a,b	0.78	0.78	m,o	0.83	0.83
b,c	0.76	0.76	n,o	0.84	0.84
KNN	0.74	0.51	a,b,c,m,n,o	0.80	0.80

3.5 N-grams test

We conducted this test to attest that the three components are contributing to the interpolation, as well as avoiding the bias the parser might induce. The n-grams test was conducted by selecting trigrams of bigrams from the UKWaC corpus in a similar manner than the previous experiments, however in this case we did not use dependency relationships, but sliding windows of hexagrams distributed in trigrams in order to mimic the way function words (*v.gr.* prepositions or determiners) affect triplets in the dependency model. The n-grams were extracted for n-grams related to the same verbs described in Section 3. The task consisted, as with the dependency triples task, to choose one amongst two options of Pair 1. The correct case is the always first pair, although the system does not know about this. We used 80% of the trigrams as a base for prediction (training), and 20% for testing. Tests were conducted for 500 triples per verb to 5000 triples per verb, in the best performance models of the previous experiment (*m,n* and *m,n,o*).

Table 3: Results of iPLSI for hexagrams grouped as trigrams of bigrams. It shows that it is possible to select the correct trigram amongst two in 75% of the cases.

Size, Mode	Prec.	Recall	Size, Mode	Prec.	Recall
500 m,n	0.75	0.70	2000 m,n,o	0.77	0.77
500 m,n,o	0.78	0.74	3000 m,n	0.70	0.70
1000 m,n	0.70	0.70	3000 m,n,o	0.75	0.75
1000 m,n,o	0.76	0.76	5000 m,n	0.72	0.72
2000 m,n	0.73	0.72	5000 m,n,o	0.76	0.76

From **Table 3** we can see that *m,n,o* is always having the best performance.

4 Conclusions

We have confirmed previous results that show that the KNN algorithm outperforms single-variable PLSI, and we study the learning rate of both algorithms, showing that KNN increases recall when more data is added, without trading much recall; however, KNN requires strongly a pre-filtering phrase which eventually leads to an important loss of scarcely occurring words.

These words are important to our purposes, because filtering them out would prevent us to generalizing rare words for measuring their plausibility. The iPLSI (interpolated PLSI) algorithm proposed here deals with that issue, yielding better results than single-variable PLSI. We have found that it is possible to select the most feasible hexagram out of two with a 75% of recall for raw n-grams grouped as trigrams of bigrams, and up to 83% recall for dependency trigrams. The conducted tests prove that it is possible to select the correct candidate for a triple, which can be regarded as part of a sentence. This allows calculating the most plausible argument in a sentence, using a broader context given by a verb and other argument.

iPLSI has outperformed the previous KNN model, but still there is room for improvement. As a future work, we should explore with true three-variable PLSI instead of a two-way interpolation, as well as other variants of iPLSI such as two-staged iPLSI, which would consist on relating two latent semantic variables with a latent variable in a second stage. Finally, since the test we conducted creates random alternatives, our system might select more probable candidates than the actual one, such as given "cow eats hay in yard", selecting the randomly created "cow eats grass in yard" would count as negative results. Although the effect of this is expected to be low, it should be considered on further analyses.

References

Bolshakov, I. A. 2005. An Experiment in Detection and Correction of Malapropisms through the Web, *LNCS 3406*, pp. 803-815.

Calvo, H., K. Inui, Y. Matsumoto. 2009a. Learning Co-Relations of Plausible Verb Arguments with a WSM and a Distributional Thesaurus. *Procs. of the 14th Iberoamerican Congress on Pattern Recognition, CIARP 2009*, Springer, Verlag. To appear.

Calvo, H., K. Inui, Y. Matsumoto. 2009b. Dependency Language Modeling using KNN and PLSI. *Procs. of the 8th Mexican International Conference on Artificial Intelligence, MICAI 2009*, Springer, Verlag, to appear.

Deerwester, S., S. T. Dumais, G. W. Furnas, Thomas K. L, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, pp. 391–407.

Deschacht, K. and M. Moens. 2009. Semi-supervised Semantic Role Labeling using the Latent Words Language Model. *Procs. 2009 Conf. on Empirical Methods in Natural Language Processing*, pp. 21–29.

Ferraresi, A., E. Zanchetta, M. Baroni and S. Bernardini. 2008. Introducing and evaluating ukWaC, a very large web-derived corpus of English. *Procs. of the WAC4 Workshop at LREC*. Marrakech, pp. 45–54.

Hoffmann, T. 1999. Probabilistic Latent Semantic Analysis, *Uncertainity in Artificial Intelligence, UAI*.

Lee, L., 1999. Measures of Distributional Similarity, *Procs. 37th ACL*.

Lin, D. 1998a. Automatic Retrieval and Clustering of Similar Words. *Procs. 36th Annual Meeting of the ACL and 17th International Conference on Computational Linguistics*.

Lin, D. 1998b. Dependency-based Evaluation of MINIPAR, *Proc. Workshop on the Evaluation of Parsing Systems*.

Merlo, P. and L. Van Der Plas. 2009. Abstraction and Generalisation in Semantic Role Labels: PropBank, VerbNet or both? *Procs. 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pp. 288–296.

Parton, K., K. R. McKeown, B. C., M. T. Diab, R. Grishman, D. Hakkani-Tür, M. Harper, H. Ji, W. Y. Ma, A. Meyers, S. Stolbach, A. Sun, G. Tur, W. Xu and S. Yaman. 2009. Who, What, When, Where, Why? Comparing Multiple Approaches to the Cross-Lingual 5W Task. 2009. *Procs. 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pp. 423–431.

Reisinger, J and M. Paşca. 2009. Latent Variable Models of Concept-Attribute Attachment. *Procs. 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pp. 620–628.

Weeds, J. and D. Weir. 2003. A General Framework for Distributional Similarity, *Procs. conf on EMNLP*, Vol. 10:81-88.

Yamada I., K. Torisawa, J. Kazama, K. Kuroda, M. Murata, S. de Saeger, F. Bond and A. Sumida. 2009. Hypernym Discovery Based on Distributional Similarity and Hierarchical Structures. *Procs. 2009 Conf. on Empirical Methods in Natural Language Processing*, pp. 929–937.