Identifying semantic relations in a specialized corpus through distributional analysis of a cooccurrence tensor

Gabriel Bernier-Colborne
OLST (Université de Montréal)
C.P. 6128, succ. Centre-Ville
Montréal (Québec) Canada H3C 3J7
gabriel.bernier-colborne@umontreal.ca

Abstract

We describe a method of encoding cooccurrence information in a three-way tensor from which HAL-style word space models can be derived. We use these models to identify semantic relations in a specialized corpus. Results suggest that the tensor-based methods we propose are more robust than the basic HAL model in some respects.

1 Introduction

Word space models such as LSA (Landauer and Dumais, 1997) and HAL (Lund et al., 1995) have been shown to identify semantic relations from corpus data quite effectively. However, the performance of such models depends on the parameters used to construct the word space. In the case of HAL, parameters such as the size of the context window can have a significant impact on the ability of the model to identify semantic relations and on the types of relations (e.g. paradigmatic or syntagmatic) captured.

In this paper, we describe a method of encoding cooccurrence information which employs a three-way tensor instead of a matrix. Because the tensor explicitly encodes the distance between a target word and the context words that co-occur with it, it allows us to extract matrices corresponding to HAL models with different context windows without repeatedly processing the whole corpus, but it also allows us to experiment with different kinds of word spaces. We describe one method whereby features are selected in different slices of the tensor corresponding to different distances between the target and context words, and another which uses SVD for dimensionality reduction. Models are evaluated and compared on reference data extracted from a specialized dictionary of the environment domain, as our target application is the identification of lexico-semantic relations in specialized corpora. Preliminary results suggest the tensor-based methods are more robust than the basic HAL model in some respects.

2 Related Work

The tensor encoding method we describe is based on the Hyperspace Analogue to Language, or HAL, model (Lund et al., 1995; Lund and Burgess, 1996), which has been shown to be particularly effective at modeling paradigmatic relations such as synonymy. In the HAL model, word order is taken into account insofar as the word vectors it produces contain information about both the cooccurrents that precede a word and those that follow it. In recent years, there have been several proposals that aim to add word order information to models that rely mainly on word context information (Jones and Mewhort, 2007; Sahlgren et al., 2008), including models based on multi-way tensors. Symonds et al. (2011) proposed an efficient tensor encoding method which builds on unstructured word space models (i.e. models based on simple cooccurrence rather than syntactic structure) by adding order information. The method we describe differs in that it explicitly encodes the distance between a target word and its cooccurrents.

Multi-way tensors have been used to construct different kinds of word space models in recent years. Turney (2007) used a word-word-pattern tensor to model semantic similarity, Van de Cruys (2009) used a tensor containing corpus-derived subject-verb-object triples to model selectional preferences, and Baroni and Lenci (2010) proposed a general, tensor-based framework for structured word space models. The tensor encoding method we describe differs in that it is based on an unstructured word space model, HAL.

3 HAL

The HAL model employs a sliding context window to compute a word-word cooccurrence matrix, which we will note $\mathbf{A}$, in which value a_{ij} is based on the number of times context word w_j appears within the context window of target word w_i. Thus, words that share cooccurrents will be closer in word space. If equal weight is given to all context words in the window, regardless of distance, we call the context window *rectangular*. In the original HAL model, the values added to $\mathbf{A}$ are inversely proportional to the distance between the target word and context word in a given context. In this case, the context window is *triangular*.

In the HAL model, the cooccurrence matrix is computed by considering only the context words that occur before the target word. Once the matrix has been computed, row vector $\mathbf{a}_{i:}$ contains cooccurrence information about words preceding w_i, and column vector $\mathbf{a}_{:i}$ contains information about those that follow it. The row vector and column vector of each target word are concatenated, such that the resulting word vectors contain information about both left-cooccurrents and right-cooccurrents. We call this type of context window *directional*, following (Sahlgren, 2006), as opposed to a *symetric* context window, in which cooccurrence counts in the left and right contexts are summed. In our experiment, we only use one type of context window (directional and rectangular), but models corresponding to different types of context windows can be derived from the cooccurrence tensor we describe in section 4.

Once the values in $\mathbf{A}$ have been computed, they can be weighted using schemes such as TF-ITF (Lavelli et al., 2004) and Positive Pointwise Mutual Information (PPMI), which we use here as it has been shown to be particularly effective by Bullinaria and Levy (2007). Finally, a distance or similarity measure is used to compare word vectors. Lund and Burgess (1996) use Minkowski distances. We will use the cosine similarity, as did Schütze (1992) in a model similar to HAL and which directly influenced its development.

4 The Cooccurrence Tensor

In the following description of the cooccurrence tensor, we follow the notational guidelines of (Kolda, 2006), as in (Turney, 2007; Baroni and Lenci, 2010). Let W be the vocabulary[1], which we index by i to refer to a target word and by j for context words. Furthermore, let P, indexed by k, be a set of positions, relative to a target word w_i, in which a context word w_j can co-occur with w_i. In other words, this is the signed distance between w_j and w_i, in number of words. For instance, in the sentence "a dog bit the mailman", we would say that "dog" co-occurs with "bit" in position -1. If we only consider the words directly adjacent to a target word, then $P = \{-1, +1\}$. If the tensor encoding method is used to generate HAL-style cooccurrence matrices corresponding to different context windows, then P would include all positions in the largest window under consideration.

In a cooccurrence matrix $\mathbf{A}$, a_{ij} contains the frequency at which word w_j co-occurs with word w_i in a fixed context window. Rather than computing matrices using fixed-size context windows, we can construct a cooccurrence tensor $\mathcal{X}$, a labeled three-way tensor in which values x_{ijk} indicate the frequency at which word w_j co-occurs with word w_i in position p_k. Table 1 illustrates a cooccurrence tensor for the sentence "dogs bite mailmen" using a context window of 1 ($P = \{-1, +1\}$), in the form of a nested table.

In tensor $\mathcal{X}$, $\mathbf{x}_{i:k}$ denotes the row vector of w_i at position p_k, $\mathbf{x}_{:jk}$ denotes the column vector of word w_j at position p_k and $\mathbf{x}_{ij:}$ denotes the tube vector indicating the frequency at which w_j co-occurs with w_i in each of the positions in P.

HAL-style cooccurrence matrices corresponding to different context windows can be extracted from the tensor by summing and concatenating various slices of the tensor. A frontal slice $\mathbf{X}_{::k}$ represents a $I \times J$ cooccurrence matrix for position p_k. A cooccurrence matrix corresponding to a symetric context window of size n can be extracted by summing the slices $\mathbf{X}_{::k}$ for $p_k \in \{-n, -n+1, \ldots, n\}$. For a directional window, we first sum the slices for $p_k \in \{-n, \ldots, -1\}$, then sum the slices for $p_k \in \{1, \ldots, n\}$, then concatenate the 2 resulting matrices horizontally.

Thus, summing and concatenating slices allows us to extract HAL-style cooccurrence matrices. A different kind of model can also be obtained by concatenating slices of the tensor. For instance, if we concatenate $\mathbf{X}_{::k}$ for $p_k \in \{-2, -1, +1, +2\}$ horizontally, we obtain a matrix containing a vec-

[1] We assume that the target and context words are the same set, but this need not be the case.

	j=1:dog		*j=2:bite*		*j=3:mailman*	
	k=1:−1	*k*=2:+1	*k*=1:−1	*k*=2:+1	*k*=1:−1	*k*=2:+1
i=1:dog	0	0	0	1	0	0
i=2:bite	1	0	0	0	0	1
i=3:mailman	0	0	1	0	0	0

Table 1: A $3 \times 3 \times 2$ cooccurrence tensor.

tor of length $4J$ (instead of the $2J$-length vectors of the HAL model) for each target word, which encodes cooccurrence information about 4 specific positions relative to that word. We will refer to this method as the tensor slicing method. Note that if $P = \{-1, 1\}$ the resulting matrix is identical to a HAL model with context size 1

As the size of the resulting vectors is KJ, this method can result in very high-dimensional word vectors. In the original HAL model, Lund et al. (1995) reduced the dimensionality of the vectors through feature selection, by keeping only the features that have the highest variance. Schütze (1992), on the other hand, used truncated SVD for this purpose. Both techniques can be used with the tensor slicing method. In our experiment, SVD was applied to the matrices obtained by concatenating tensor slices horizontally[2]. As for feature selection, a fixed number of features (those with the highest variance) were selected from each slice of the tensor, and these reduced slices were then concatenated.

It must be acknowledged that this tensor encoding method is not efficient in terms of memory. However, this was not a major issue in our experimental setting, as the size of the vocabulary was small (5K words), and we limited the number of positions in P to 10. Also, a sparse tensor was used to reduce memory consumption.

5 Experiment

5.1 Corpus and Preprocessing

In this experiment, we used the PANACEA Environment English monolingual corpus, which is freely distributed by ELDA for research purposes[3] (Catalog Reference ELRA-W0063). This corpus contains 28071 documents (∼50 million tokens) dealing with different aspects of the environment domain, harvested from web sites using a focused crawler. The corpus was converted from XML to raw text, various string normalization operations were then applied, and the corpus was lemmatized using TreeTagger (Schmid, 1994). The vocabulary (W) was selected based on word frequency: we used the 5000 most frequent words in the corpus, excluding stop words and strings containing non-alphabetic characters. During computation of the cooccurrence tensor, OOV words were ignored (rather than deleted), and the context window was allowed to span sentence boundaries.

5.2 Evaluation Data

Models were evaluated using reference data extracted from DiCoEnviro[4], a specialized dictionary of the environment. This dictionary describes the meaning and behaviour of terms of the environment domain as well as the lexico-semantic relations between these terms. Of the various relations encoded in the dictionary, we focused on a subset of three paradigmatic relations: near-synonyms (terms that have similar meanings), antonyms (opposite meanings), and hyponyms (kinds of). 446 pairs containing a headword and a related term were extracted from the dictionary. We then filtered out the pairs that contained at least one OOV term, and were left with 374 pairs containing two paradigmatically-related, single-word terms. About two thirds (246) of these examples were used for parameter selection, and the rest were set aside for a final comparison of the highest-scoring models.

[2] We also tried concatenating slices vertically (thus obtaining a matrix where rows correspond to <target word, position> tuples and columns correspond to context words) before applying SVD, then concatenating all row vectors corresponding to the same target word, but we will not report the results here for lack of space. Concatenating slices horizontally performed better and seems more intuitive, and the size of the resulting vectors is not dependent on the number of positions in P.

[3] http://catalog.elra.info/product_info.php?products_id=1184

[4] http://olst.ling.umontreal.ca/cgi-bin/dicoenviro/search_enviro.cgi (under construction).

5.3 Automatic Evaluation

Each model was automatically evaluated on the reference data as follows. For each <headword, related term> pair in the training set, we computed the cosine similarity between the headword and all other words in the vocabulary, then observed the rank of the related term in the sorted list of neighbours. The score used to compare models is recall at k (R@k), which is the percentage of cases where the related term is among the k nearest neighbours of the headword. It should be noted that a score of 100% is not always possible in this setting (depending on the value of k), as some headwords have more than 1 related term in the reference data. Nonetheless, since most ($\sim$70%) have 1 or 2 related terms, R@k for some small value of k (we use k = 10) should be a good indicator of accuracy. A measure that explicitly accounts for the fact that different terms have different numbers of related terms (e.g. R-precision) would be a good alternative.

5.4 Models Tested

We compared HAL and the tensor slicing method using either feature selection or SVD[5], as explained in section 4. We will refer to each of these models as HAL_{SEL}, TNSR_{SEL}, HAL_{SVD} and TNSR_{SVD}. Context sizes ranged from 1 to 5 words. For feature selection, the number of features could take values in $\{1000, 2000, \ldots, 10000\}$, 10000 being the maximum number of features in a HAL model using a vocabulary of 5000 words. In the case of TNSR_{SEL}, to determine the number of features selected per slice, we took each value in $\{1000, 2000, \ldots, 10000\}$, divided it by K (the number of positions in P), and rounded down. This way, once the slices are concatenated, the total number of features is equal to (or slightly less than) that of one of the HAL_{SEL} models, allowing for a straightforward comparison. When SVD was used instead of feature selection, the number of components could take values in $\{100, 200, ..., 1000\}$. In all cases, word vectors were weighted using PPMI and normalized[6].

[5]We used the SVD implementation (ARPACK solver) provided in the scikit-learn toolkit (Pedregosa et al., 2011).

[6]For HAL_{SEL} and TNSR_{SEL}, we apply PPMI weighting *after* feature selection. In the case of TNSR_{SEL}, we wanted to avoid weighting each slice of the tensor separately. We decided to apply weighting after feature selection in the case of HAL_{SEL} as well in order to enable a more straightforward comparison. We should also note that, in our experiments using HAL, PPMI weighting performed better when applied after feature selection, especially for low numbers of features.

absorb	**extreme**	**precipitation**
emit	severe	rainfall
sequester	intense	snowfall
convert	harsh	temperature
produce	catastrophic	rain
accumulate	unusual	evaporation
store	seasonal	runoff
radiate	mild	moisture
consume	cold	snow
remove	dramatic	weather
reflect	increase	deposition

Table 2: 10 nearest neighbours of 3 environmental terms using the HAL_{SEL} model.

6 Results

Table 2 illustrates the kinds of relations identified by the basic HAL_{SEL} model. It shows the 10 nearest neighbours of the verb *absorb*, the adjective *extreme* and the noun *precipitation*. If we compare these results with the paradigmatic relations encoded in DiCoEnviro, we see that, in the case of *absorb*, 3 of its neighbours are encoded in the dictionary, and all 3 are antonyms or terms having opposite meanings: *emit*, *radiate*, and *reflect*. As for *extreme*, the top 2 neighbours are both encoded in the dictionary as near-synonyms. Finally, *rain* and *snow* are both encoded as kinds of *precipitation*. Most of the other neighbours shown here are also paradigmatically related to the query terms. Thus, HAL seems quite capable of identifying the three types of paradigmatic relations we hoped to identify.

Table 3 shows the best R@10 achieved by each model on the training set, which was used to tune the context size and number of features or components, and their scores on the test set, which was only used to compare the best models. In the case of HAL_{SEL}, the best model has a context window size of 1 and uses 9K out of 10K available features. As for TNSR_{SEL}, the best model had a context size of 2 ($P = \{-2, -1, +1, +2\}$) and 10000 features (2500 per slice). It performed only slightly better on the training set, however it beat the HAL model with a wider margin on the test set.

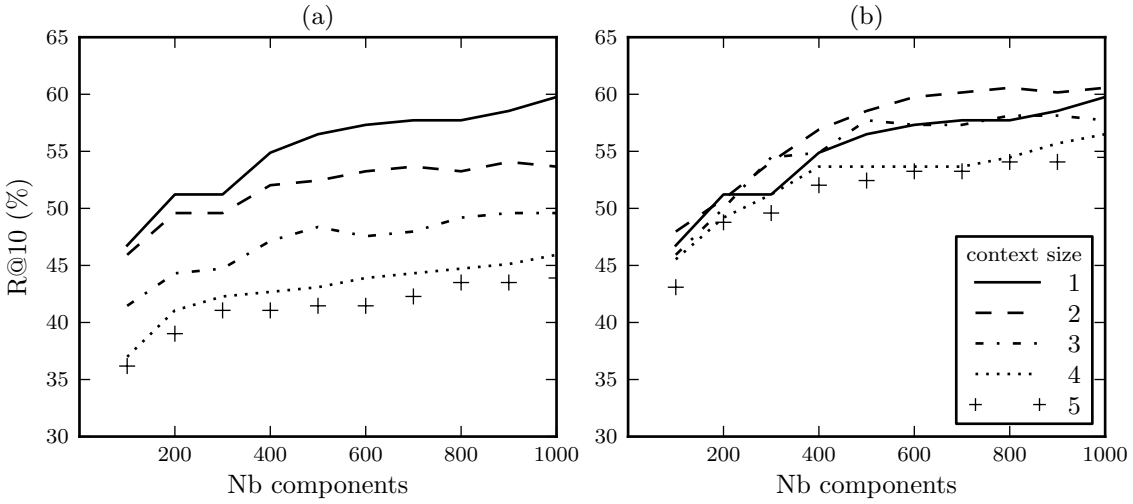

Figure 1: HAL vs. tensor slicing method using SVD for dimensionality reduction. R@10 is plotted against number of components. Models are identical when context size is 1. **(a)** HAL_{SVD} **(b)** TNSR_{SVD}

Model	**Train**	**Test**
HAL_{SEL}	60.57	57.03
TNSR_{SEL}	60.98	60.94
HAL_{SVD}	59.76	56.25
TNSR_{SVD}	60.57	60.16

Table 3: R@10 (%) of best models.

The best HAL_{SVD} model used a 1-word window and 1000 components, whereas the best TNSR_{SVD} model had a context size of 2 and 800 components. Again, the tensor-based model slightly edged out the HAL model on the training set, but performed considerably better on the test set.

Further analysis of the results indeed suggests that the tensor slicing method is more robust in some respects than the basic HAL model. Figure 1 compares the performance of HAL_{SVD} and TNSR_{SVD} on the training set, taking into account context size and number of components. It shows that the HAL model is quite sensitive to context size, narrower context performing better in this task. The tensor-based method reduces this gap in performance between context sizes, the gain being greater for larger context sizes. Furthermore, using the tensor-based method with a slightly wider context (2) raises R@10 for most values of the number of components. Results obtained with HAL_{SEL} and TNSR_{SEL} follow the same trend, the tensor-based method being more robust with respect to context size. For lack of space, we only show the plot comparing HAL_{SVD} and TNSR_{SVD}.

7 Concluding Remarks

The work presented in this paper is still in its exploratory phase. The tensor slicing method we described has only been evaluated on one corpus and one set of reference data. Experiments would need to be carried out on common word space evaluation tasks in order to compare its performance to that of HAL and other word space models. However, our results suggest that the tensor-based methods are more robust than the basic HAL model to a certain extent, and can improve accuracy. This could prove especially useful in settings where no reference data are available for parameter tuning.

Various possibilities offered by the cooccurrence tensor remain to be explored, such as weighting the number of features selected per slice using some function of the distance between words, extracting matrices from the tensor by applying various functions to the tube vectors corresponding to each word pair, and applying weighting functions that have been generalized to higher-order tensors (Van de Cruys, 2011) or tensor decomposition methods such as those described in (Turney, 2007).

Acknowledgements

We would like to thank the anonymous reviewers for their helpful and thorough comments. Funding was provided by the Social Sciences and Humanities Research Council of Canada.

References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510–526.

Michael N Jones and Douglas JK Mewhort. 2007. Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, 114(1):1–37.

Tamara Gibson Kolda. 2006. Multilinear operators for higher-order decompositions. Technical Report SAND2006-2081, Sandia National Laboratories.

Thomas K Landauer and Susan T Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211.

Alberto Lavelli, Fabrizio Sebastiani, and Roberto Zanoli. 2004. Distributional term representations: An experimental comparison. In *Proceedings of the 13th ACM International Conference on Information and Knowledge Management*, pages 615–624. ACM.

Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208.

Kevin Lund, Curt Burgess, and Ruth Ann Atchley. 1995. Semantic and associative priming in high-dimensional semantic space. In *Proceedings of the 17th Annual Conference of the Cognitive Science Society*, volume 17, pages 660–665.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Magnus Sahlgren, Anders Holst, and Pentti Kanerva. 2008. Permutations as a means to encode order in word space. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society*, pages 1300–1305. Cognitive Science Society.

Magnus Sahlgren. 2006. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Ph.D. thesis, Stockholm University.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*.

Hinrich Schütze. 1992. Dimensions of meaning. In *Proceedings of the 1992 ACM/IEEE Conference on Supercomputing (Supercomputing'92)*, pages 787–796. IEEE Computer Society Press.

Michael Symonds, Peter D Bruza, Laurianne Sitbon, and Ian Turner. 2011. Modelling word meaning using efficient tensor representations. In *Proceedings of 25th Pacific Asia Conference on Language, Information and Computation*.

Peter Turney. 2007. Empirical evaluation of four tensor decomposition algorithms. Technical Report ERB-1152, National Research Council of Canada, Ottawa.

Tim Van de Cruys. 2009. A non-negative tensor factorization model for selectional preference induction. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 83–90. ACL.

Tim Van de Cruys. 2011. Two multivariate generalizations of pointwise mutual information. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 16–20. ACL.