Encoding Sequential Information in Vector Space Models of Semantics: Comparing Holographic Reduced Representation and Random Permutation

Gabriel Recchia (grecchia@indiana.edu)
Cognitive Science Program, 1910 E 10th St.
Indiana University, Bloomington, Indiana USA

Michael N. Jones (jonesmn@indiana.edu)
Department of Psychological and Brain Sciences
Indiana University, Bloomington, Indiana USA

Magnus Sahlgren (mange@sics.se)
Swedish Institute of Computer Science
Box 1263, SE-164 29 Kista, Sweden

Pentti Kanerva (pkanerva@berkeley.edu)
Redwood Center for Theoretical Neuroscience
University of California, Berkeley, California, USA

Abstract

Encoding information about the order in which words typically appear has been shown to improve the performance of high-dimensional semantic space models. This requires an encoding operation capable of binding together vectors in an order-sensitive way, and efficient enough to scale to large text corpora. Although both circular convolution and random permutations have been enlisted for this purpose in semantic models, these operations have never been systematically compared. In Experiment 1 we compare their storage capacity and probability of correct retrieval; in Experiments 2 and 3 we compare their performance on semantic tasks when integrated into existing models. We conclude that random permutations are a scalable alternative to circular convolution with several desirable properties.

Keywords: semantic representation, semantic space models, binding, convolution, permutation, random indexing.

Introduction

Vector-space models of lexical semantics have seen considerable recent attention in the psychological literature both as automated tools to estimate semantic similarity between words, and as psychological models of how humans learn and represent word meaning from repeated contextual co-occurrences. In general, these models build semantic representations for words from statistical redundancies observed in a large corpus of text (e.g., Landauer & Dumais, 1997; Lund & Burgess, 1996). As tools, the models have provided invaluable metrics of semantic similarity for stimulus selection and control in behavioral experiments using words, sentences, and larger units of discourse. As psychological models, the vectors derived from distributional models serve as useful representations in computational models of word recognition, priming, and higher-order comprehension (Landauer et al., 2007). In addition, the abstraction algorithms themselves are often proposed as models of the cognitive mechanisms humans use to learn meaning from repeated episodic experience.

A classic example of a vector-space model is Landauer and Dumais' (1997) Latent Semantic Analysis (LSA). LSA begins with a word-by-document co-occurrence matrix representation of a text corpus. A lexical association function is applied to dampen the importance of each word proportionate to its entropy over documents. Finally, an algorithm is applied to reduce the matrix's dimensionality; words are represented as vectors whose dimensions refer to the largest eigenvalues of the reduced representation.

Despite their successes both as tools and as psychological models, vector-space models suffer from several shortcomings. Most prominently, the models have been criticized as "bag of words" models that encode only the contexts in which words co-occur, but ignore *word-order information*. The role of word order was traditionally thought to apply only to the rules of word usage (grammar) rather than to the lexical meaning of the word itself. However, temporal information is now taking a more prominent role in the lexical representation of a word's meaning. Recently, Elman (2009) has convincingly argued that an inherent part of a word's lexical representation is information about its common temporal context, event knowledge, and habits of usage (cf. McKoon & Ratcliff, 2003; see also Hare et al., 2009).

A second issue for these models is *lack of scalability* (Recchia & Jones, 2009; Kanerva, Kristofersson, & Holst, 2000), due to reliance on computationally complex decomposition techniques to reveal the latent components in a word-by-document matrix (e.g., singular value decomposition). Not only is decomposition computationally expensive, the entire word-by-document matrix must be stored in memory during the operation. The problem is exacerbated by the fact that as the size of the corpus increases, the number of both rows and columns in the matrix increase significantly, the number of columns growing linearly with added documents, and the number of rows growing approximately in proportion to the square root of the number of tokens (Heap's law). The corpora that vector-space models like LSA are most commonly trained upon in the literature contain approximately the number of tokens that children are estimated to have experienced before age 3, not counting words that they produce during this time (Riordan & Jones, 2007; Risley & Hart, 2006). Recently, Recchia and Jones (2009) demonstrated that although simple semantic metrics such as pointwise mutual information (PMI) are outperformed by more complex models such as LSA on small corpora, PMI is capable of much better correspondence to human-derived semantic similarity judgments due to its ability to scale to large corpora. This led the authors to favor simple and scalable

algorithms to more complex non-scalable algorithms, concordant with approaches that have met with success in the computational linguistics literature (e.g. Banko & Brill, 2001).

Encoding Word Order

Two recent vector-space models that directly address the concerns of word order and scalability are Jones and Mewhort's BEAGLE (2007) and the "random permutation" model of Sahlgren, Holst, and Kanerva (2008) (henceforth referred to as RPM). Rather than starting with a word-by-document matrix, BEAGLE and RPM maintain static, randomly generated *signal vectors* intended to represent the invariant properties of each word (such as its orthography or phonology), as well as dynamic *memory vectors* that store information about each word's semantic representation. To represent statistical information about the word, BEAGLE and RPM bind together collections of signal vectors into *order vectors* that are added to memory vectors during training. Integrating word-order information has yielded greater success at fitting a variety of human semantic data than encoding only contextual information (e.g., Jones, Kintsch, & Mewhort, 2006; Jones & Mewhort, 2007). Because they require neither the overhead of a large word-by-document matrix nor computationally intensive matrix decomposition techniques, both models are significantly more scalable than traditional vector-space models.

Although BEAGLE and RPM differ in several ways, arguably the most important difference lies in the nature of the binding operation used to create order vectors. BEAGLE uses circular convolution, a binary operation (henceforth denoted as *) performed on two vectors such that every element i of $(x * y)$ is given by:

$$\sum_{j=0}^{D-1} x_j \cdot y_{(i-j) \bmod D}, \qquad (1)$$

where D is the dimensionality of x and y. Circular convolution can be seen as a modulo-n variation of the tensor product of two vectors x and y such that $(x * y)$ is of the same dimensionality as x and y. Furthermore, although $(x * y)$ is dissimilar from both x and y by any distance metric, approximations of x and y can be retrieved via the inverse operation of correlation.

In contrast, RPM employs *random permutations*, henceforth referred to as RPs. True to their name, RPs are functions that map input vectors to output vectors such that the outputs are simply randomly shuffled versions of the inputs. Just as $(x * y)$ yields a vector that differs from x and y but from which approximations of x and y can be retrieved, the sum of two RPs of x and y, $\Pi x + \Pi^2 y$ (where $\Pi^2 y$ is defined as $\Pi(\Pi y)$) yields a vector dissimilar from x and y but from which approximations of the original x and y can be retrieved via the inverse permutations Π^{-1} and Π^{-2}.

Both systems offer efficient storage properties, compressing order information into a single composite vector representation, and both encoding operations are reversible. However, RPs are much more efficient to compute. In language applications of BEAGLE, the computationally expensive convolution operation is what limits the size of a text corpus that the model can encode. As Recchia and Jones (2009) have demonstrated, scaling a semantic model to more data produces much better fits to human semantic data. Hence, both order information and magnitude of linguistic input have been demonstrated to be important factors in human semantic learning. If RPs have similar characteristics to convolution, they may afford encoding very large-scale order information, and much better approximations to human semantic structure.

This work is further motivated by the cognitive implications of circular convolution and RPs. Vector representations constructed by means of circular convolution have been frequently described as psychologically or neurally plausible (Levy, 2007; Jones & Mewhort, 2007), due to several features that they share with connectionist networks: distributed encoding, robustness to noise, affordance of generalization, error correction, pattern completion, and easy associative access (Plate, 2003). Furthermore, implementing neural networks that instantiate convolution-like operations is straightforward (Plate, 2000; but compare Pike, 1986). Similarly, RPs possess many properties relevant to human cognition. Not only have they been proposed as a particularly versatile multiplication operator for constructing vector representations that are highly distributed, tolerant of noise in the input, robust to error and component failure, and mathematically compatible with several known properties of neural circuitry (Kanerva, 2009), RPs are trivially easy to implement in connectionist terms; a RP can simply be thought of as a two-layer network connected by randomly placed one-to-one copy connections. Thus, comparing circular convolution and RPs affords us a better understanding of two psychologically plausible operations for encoding semantic information that have never been systematically compared.

We conducted three experiments intended to compare convolution and RPs as means of encoding word-order information with respect to performance and scalability. In Experiment 1, we conducted an empirical comparison of the storage capacity and the probability of correct decoding under each method. In Experiment 2, we compared RPs with convolution in the context of a simple vector accumulation model equivalent to BEAGLE's "order space" (Jones and Mewhort, 2007) on a small battery of semantic evaluation tasks when trained on a Wikipedia corpus. The model was trained on both the full corpus and a random subset; results improved markedly when RPs are allowed to scale up to the full Wikipedia corpus, which proved to be intractable for the convolution-based model. Finally, in Experiment 3, we specifically compared BEAGLE to RPM, which differs from BEAGLE in several important ways other than its binding operation, to assess whether using RPs in the context of RPM improves their performance further. We conclude that random permutations are a promising and scalable alternative to circular convolution.

Experiment 1

Plate (2003) made a compelling case for circular convolution in the context of holographic reduced representation, demonstrating its utility in constructing distributed representations with high storage capacity and high probability of correct retrieval. However, the storage capacity and probability of correct retrieval with RPs has not been closely investigated. This experiment compared the probability of correct retrieval of RPs with circular convolution under varying dimensionality and number of vectors stored.

Method

As a test of the capacity of convolution-based associative memory traces, Plate (2003, Appendix D) describes a simple paired-associative retrieval task in which the algorithm must select, from set E of m possible random vectors, the vector x_i that is bound to its associate y_i . The retrieval algorithm is provided with a trace vector of the form $t = (x_1 * y_1) + (x_2 * y_2) + (x_3 * y_3) + \ldots$ that stores a total of k vectors. All vectors are of dimensionality D, and each x_i and y_i is a vector with elements independently drawn from $N(0, 1/D)$. The retrieval algorithm is provided with the trace t and the probe y_i , and works by first calculating $a = (y_i \# t)$, where # is the *correlation operator* described in detail in Plate (2003, pp. 94-97). It then retrieves the vector in the "clean-up memory" set E that is the most similar to a. This is accomplished by calculating the cosine between a and each vector in the set E, and retrieving the vector from E for which the cosine is highest. If this vector is not equal to x_i , this counts as a retrieval error. We replicated Plate's method to empirically derive retrieval accuracies for a variety of choices of k and D, keeping m fixed at 1,000.

Sahlgren et al. (2008) essentially bind signal vectors to positions by means of successive self-composition of a permutation function Π, and construct trace vectors by superposing the results. Because the signal vectors are random, any permutation function that maps each element of the input onto a different element of the output will do; we adopt Sahlgren et al.'s suggestion of using rotation of a vector by one position for Π for the sake of simplicity. We also use their notation of $\Pi^n x$ to mean "Π composed with itself n times;" thus, $\Pi^2 x = \Pi(\Pi x)$, $\Pi^3(x) = \Pi(\Pi^2 x)$), and so forth. The notion of a trace vector of paired associations can then be recast in RP terms as follows:

$$t = (\Pi y_1 + \Pi^2 x_1) + (\Pi^3 y_2 + \Pi^4 x_2) + (\Pi^5 y_3 + \Pi^6 x_3) + \ldots$$

where the task again is to retrieve some y_i's associate x_i when presented only with y_i and t. A retrieval algorithm for accomplishing this can be described as follows: Given a probe vector y_i , the algorithm applies the inverse of the initial permutation to trace vector t, yielding $\Pi^{-1}t$. Next, the cosine between $\Pi^{-1}t$ and the probe vector y_i is calculated, yielding a value that represents the similarity between y_i and $\Pi^{-1}t$. These steps are then iterated: the algorithm calculates the cosine between y_i and $\Pi^{-2}t$, between y_i and $\Pi^{-3}t$, etc., until this similarity value exceeds some high threshold; this indicates that the algorithm has probably "found" y_i in the trace. At that point, t is permuted one more time, yielding x', a noisy approximation of y_i's associate x_i. This approximation x' can then be compared with clean-up memory to retrieve the original associate x_i.

Alternatively, rather than selecting a threshold, t may be permuted some finite number of times[1] and its cosine similarity to y_i calculated for each permutation. Let n indicate the inverse permutation for which $\cos(\Pi^{-n}t, y_i)$ is the highest. We can permute one more time to get $\Pi^{-n-1}t$, that is, our noisy approximation x'. This method is appropriate if we always want our algorithm to return an answer (rather than, say, timing out before the threshold is exceeded), and is the method we used for this experiment.

The final clean-up memory step is identical to that used by Plate (2003): we calculate the cosine between x' and each vector in the clean-up memory E, and retrieve the vector in E for which this cosine is highest. As when evaluating convolution, we keep m (the number of vectors in E) fixed at 1,000 while varying the number of stored vectors k and the dimensionality D.

Results

Figure 1 reports retrieval accuracies for convolution-based associative memory traces, while Figure 2 reports retrieval accuracies for the RP formulation of the task. 500 vector pairs were sampled randomly from a pool of 1,000 possible random vectors with replacement and the proportion of correct retrievals was computed. All 1,000 vectors in the pool were potential candidates; thus, an accuracy of 0.1% would represent chance performance. The horizontal axes of all figures indicate the total number of pairs stored in the trace (i.e., half the total number of vectors in the trace).

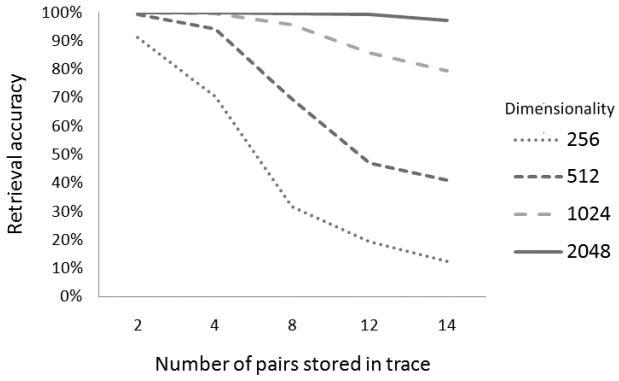

Figure 1. Retrieval accuracies for convolution-based associative traces.

[1] In Plate's (2003, p. 252) demonstration of the capacity of convolution-based associative memories, the maximal number of pairs stored in a single trace was 14; we likewise restrict the maximal number of pairs in a single trace to 14 (28 items total).

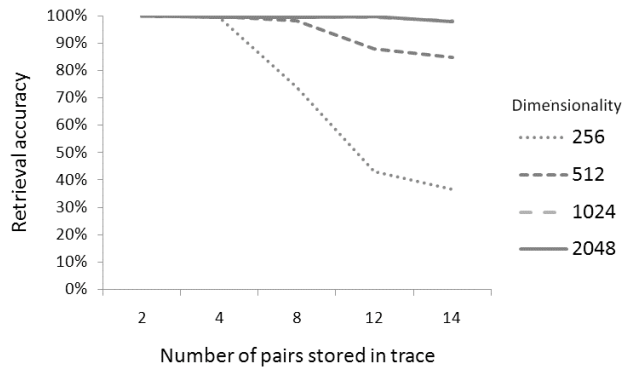

Figure 2. Retrieval accuracies for RP-based associative traces.

Discussion

Circular convolution has an impressive storage capacity and excellent probability of correct retrieval at high dimensionalities; the results were comparable to those reported by Plate (2003, p. 252) in his test of convolution-based associative memories. However, RPs seem to share these desirable properties as well. In fact, the storage capacity of RPs seems to drop off significantly more slowly than does the storage capacity of convolution as dimensionality is reduced.

This information capacity is particularly interesting given that, with respect to basic encoding and decoding operations, RP is computationally more efficient than convolution. Encoding *n*-dimensional bindings with circular convolution using equation (1) is a very slow $O(n^2)$ operation. This can be sped to $O(n)$ by means of the Fast Fourier transform (Jones, 2007; Plate, 2003). The algorithm to bind two vectors *a* and *b* in $O(n)$ time involves calculating discrete Fourier transforms of *a* and *b*, multiplying them pointwise to yield a new vector *c*, and calculating the inverse discrete Fourier transform of *c*. Encoding with RPs can also be accomplished in $O(n)$ time, but with steps that are not as computationally expensive. To bind two vectors *a* and *b,* the elements of *a* are permuted by directly copying then into a new vector, but with the mapping of their indices determined by the permutation function. For example, if the permutation function were chosen to be *rotation by one position* and vectors were of dimensionality *D*, each value at index *i* in the vector *a* would be copied to index $(i + 1) \bmod D$ in the new vector. The vector *b* is permuted in the same way, but using a different permutation function (e.g., $(i + 2) \bmod D$). Finally, *a* and *b* are added to yield a final binding *c*.

Noisy decoding—the retrieval of a noisy version of one or more bound associates from a trace (which may then be passed to clean-up memory to unambiguously determine the identity of the associate)—also operates in $O(n)$ time in both representations. As with encoding, the operation is $O(n)$, but fewer operations are required (a single permutation decodes one associate, rather than an involution + two discrete Fourier transforms + an elementwise multiplication + one inverse discrete Fourier transform).

Experiment 2

In order to move from the paired-associates problem of Experiment 1 to a real language task, we evaluated how a simple vector accumulation model akin to Jones & Mewhort's (2007) encoding of order-only information in BEAGLE would perform on a set of semantic tasks if RPs were used in place of circular convolution. In Experiment 2, we replaced the circular convolution component of BEAGLE with RPs so that we could quantify the impact that the choice of operation alone had on the results. Due to the computational efficiency of RPs, we were able to scale them to a larger version of the same textbase, and simultaneously explore the effect of scalability on order.

Method

Order information was trained using both the BEAGLE model and a modified implementation of BEAGLE in which the circular convolution operation was replaced with RPs as they are described in Sahlgren at al. (2008). A brief example will illustrate how this replacement changes the algorithm. Recall that in BEAGLE, each word *w* is assigned a static "environmental" signal vector e_w as well as a dynamic memory vector m_w that is updated during training. Recall also that the memory vector of a word *w* is updated by adding the sum of the convolutions of all n-grams (up to some maximum length λ) containing *w*. Upon encountering the phrase "one two three" in a corpus, the memory vector for "one" would normally be updated as follows:

$$m_{\text{one}} = m_{\text{one}} + (\Phi * e_{\text{two}}) + (\Phi * e_{\text{two}} * e_{\text{three}})$$

where Φ is a placeholder signal vector that represents the word whose representation is being updated. In the modified BEAGLE implementation used in this experiment, the memory vector for "one" would instead be updated as:

$$m_{\text{one}} = m_{\text{one}} + \Pi e_{\text{two}} + \Pi^2 e_{\text{three}}$$

The modified BEAGLE implementation was trained on a 2.33 GB corpus (418 million tokens) of documents from Wikipedia. Training on a corpus this large proved intractable for the slower convolution-based approach. Hence, we also trained both models on a 35 MB, six-million-token subset of this corpus constructed by sampling random 10-sentence documents from the larger corpus without replacement. Accuracy was evaluated on two synonymy tests: the English as a Second Language (ESL) and the Test of English as a Foreign Language (TOEFL) synonymy assessments. Rank correlations to human judgments of the semantic similarity of word pairs were calculated using the similarity judgments obtained from Rubenstein and Goodenough (G, 1965), Miller and Charles (M&C, 1991), Resnik (R, 1995), and Finkelstein et al. (F&al, 2002). A description of these measures can be found in Recchia and Jones (2009).

Results and Discussion

Table 3 provides a comparison of two variants of the BEAGLE model, each trained on order information only.

"Convolution" refers to the original BEAGLE as described in Jones & Mewhort, while "Random Permutations" refers to a version in which order information is encoded using RPs rather than circular convolution. Three points about these results merit special attention. First, there are no significant differences between the performance of convolution and RPs on the small corpus. Both performed nigh-identically on F and TOEFL; neither showed any significant correlations with human data on R&G, M&C, R, nor performed better than chance on ESL.

Table 3. Comparisons of variants of BEAGLE that differ by binding operation. Accuracy scores are reported for ESL & TOEFL; remaining tasks are Spearman rank correlations.

Criterion	Wikipedia subset		Full Wikipedia
	Convolution	Random Permutations	Random Permutations
ESL	.20	.26	.32
TOEFL	.46†	.46†	.63†
R&G	.07	-.06	.32*
M&C	.08	-.01	.33*
R	.06	-.04	.35*
F&al	.13*	.12*	.33*

*Significant correlation, $p < .05$, one-tailed.
†Accuracy score differs significantly from chance, $p < .05$, one-tailed.

Second, both models performed the best by far on the TOEFL synonymy test, supporting Sahlgren's et al. (2008) claim that order information may indeed be more useful for synonymy tests than tests of semantic relatedness, as paradigmatic rather than syntagmatic information sources are most useful for the former. However, it is unclear exactly why neither model did particularly well on ESL[2], as many models have achieved scores on it comparable to their scores on TOEFL (Recchia & Jones, 2009). Finally, only RPs were able to scale up to the full Wikipedia corpus, and doing so yielded extreme benefits for every task. This is a very strong point in favor of RPs, and suggests that sequential information can even be useful for tasks that involve semantic relatedness but not synonymy per se (R&G, M&C, R, F), provided that the model is trained at a sufficiently large scale.

Experiment 3

In Experiment 2 we saw that importing RPs into BEAGLE yielded comparable results on a small corpus and considerable improvement in scalability. Here we compare BEAGLE to the original model of Sahlgren et al., which we have been referring to as RPM. In many ways the two are similar: Like BEAGLE, RPM can construct a semantic space by (1) adding only order vectors to memory vectors during training, yielding an "order space," and (2) by adding order vectors *as well as* "context vectors," yielding a "composite space." Besides using RPs in place of circular convolution, the specific implementation of RPM reported by Sahlgren et al. differs from BEAGLE in several ways (signal-vector representation, window size, lexicon size, and the stoplist). This experiment aims to assess RPM's performance with another corpus and on other semantic tasks besides TOEFL, and to determine if performance improves under RPM parameter settings (compared to the BEAGLE settings in Experiment 2).

Method

The same evaluation method was applied as in Experiment 2, but with BEAGLE being compared directly to RPM. Both models were trained in order and composite space.

Results and Discussion

Table 4 reports the results of BEAGLE and RPM trained in order space, while Table 5 reports results in composite space (context + order information). As in Experiment 2, RPs but not convolution proved capable of scaling up to the full Wikipedia corpus. We replicated Sahlgren et al.'s (2008) performance on TOEFL in order space at this dimensionality, but this Wikipedia implementation of RPM fell short of the ~.73 accuracy they reported on TOEFL at a dimensionality of 2000 in composite space; the difference is most likely due to the different corpora used in the two evaluations. On the small corpus, switching from order space to composite space did not yield significant differences for either model when contrasted with the use of order space alone. On the large corpus, however, when contrasted with RPs in Experiment 2 (Table 3), RPM performed far better on several evaluations, most notably the correlations to the R&G, M&C, and R similarity judgments. It is intriguing that the version of RPM trained on the full Wikipedia in order space was able to perform well on several tasks that are typically conceived of as tests of associative relatedness and not tests of synonymy per se—for example, .70 on the Miller & Charles pairs (Table 4).

Table 4. BEAGLE and RPM in order space.

Criterion	Wikipedia subset		Full Wikipedia
	BEAGLE	RPM	RPM
ESL	.20	.27	.38†
TOEFL	.46†	.37†	.65†
R&G	.07	.15	.50*
M&C	.08	.16	.70*
R	.06	.11	.63*
F&al	.13*	.18*	.32*

* Significant correlation, $p < .05$, one-tailed.
†Accuracy score differs significantly from chance, $p < .05$, one-tailed.

[2] Note that the absolute performance of these models is irrelevant to the important comparisons. Many factors (e.g., frequency thresholding, morphological normalization, corpus siz/type) are known to improve performance on synonymy tests; we held these constant, which produced poor absolute performance (but see Sahlgren et al., 2008). The key comparisons are the consistency of the operations on the same textbase, and the relative performance boost when data are scaled up.

Table 5. BEAGLE and RPM in composite space.

Criterion	Wikipedia subset		Full Wikipedia
	BEAGLE	RPM	RPM
ESL	.24	.27	.42†
TOEFL	.47†	.40†	.66†
R&G	.10	.10	.49*
M&C	.09	.12	.70*
R	.09	.03	.60*
F&al	.23*	.19*	.32*

* Significant correlation, $p < .05$, one-tailed.
†Accuracy score differs significantly from chance, $p < .05$, one-tailed.

General Discussion

Experiment 1 demonstrates that RPs are capable of high retrieval accuracy even when many paired associates are stored in a single trace, and their storage capacity appears to be slightly better than that of circular convolution for low dimensionalities. Experiments 2 and 3 reveal that both methods achieve approximately equal performance on a battery of semantic tasks when trained on a small corpus, but that RPs are ultimately capable of achieving superior performance due to their higher scalability. In all, these results suggest that RPs are worthy of further study both as encoders of sequential information in word space models and as operators capable of storing associative information more generally. It should be noted that Sahlgren et al. (2008) found better synonymy performance when RPs were trained on "direction" vectors rather than order vectors; direction vectors simply encode whether words appear before or after a word in the temporal stream, but ignore the order chain. Given the computational efficiency of this approach, future work should explore the effects of scaling to large-scale data on RP direction vectors.

Both convolutions and RPs are naturally derived from properties of the human cognitive system, namely groups of neurons connected with a certain degree of randomness (see Plate, 2003 for convolution and Kanerva, 2009 for RPs; also see Howard et al. [in press] for a related model using neural properties of temporal context encoding). The current work demonstrates that when a model is able to apply these associative learning mechanisms across a large amount of episodic experience with linguistic structure, it produces much better approximations of human semantic structure. As Elman (2009) has argued, the encoding of large-scale order information is a core component of a word's lexical representation that is often overlooked. Future work needs to explore application of large-scale RP encoding to more complex semantic and linguistic tasks.

Acknowledgements

This research was supported in part by a grant from Google Inc. to MNJ and a Shared University Research Grant from IBM to Indiana University.

References

Banko, M. & Brill, E. (2001). Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Conference of the Association for Computational Linguistics* (pp. 26-33). Toulouse, France: Association for Computational Linguistics.

Burgess, C., & Lund, K. (2000). The dynamics of meaning in memory. In E. Dietrich & A. Markman (Eds.), *Cognitive dynamics: Conceptual and representational change in humans and machines* (pp. 117-156).

Elman, J. L. (2009). On the meaning of words and dinosaur bones: Lexical knowledge without a lexicon. *Cognitive Science, 33,* 547-582.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Z., S., Wolfman, G., & Ruppin, E. (2002). Placing search in context: The concept revisited. *Association for Computing Machinery Transactions on Information Systems, 20*(1), 116-131.

Hare, M., Jones, M. N., Thomson, C., Kelly, S., & McRae, K. (2009). Activating event knowledge. *Cognition, 111*(2), 151-167.

Howard, M. W., Shankar, K. H., and Jagadisan, U. K. K. (In press). Constructing semantic representations from a gradually-changing representation of temporal context. *Topics in Cognitive Science.*

Jones, M. N. (2007). Holographic neural networks. Poster presented at the 48th Meeting of the Psychonomic Society. Long Beach, CA.

Jones, M. N., Kintsch, W., & Mewhort, D. J. K. (2006). High-dimensional semantic space accounts of priming. *Journal of Memory and Language, 55,* 534-552.

Jones, M. N. & Mewhort, D. J. K. (2007). Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, 114, 1-37.

Miller, G. A. and Charles, W. G. (1991). Contextual correlates of semantic similarity. *Language and Cognitive Processes,* 6(1), 1-28.

McKoon, G. & Ratcliff, R. (2003). Meaning through syntax: Language comprehension and the reduced relative clause construction. *Psychological Review, 110*, 490-525.

Kanerva, P. (2009). Hyperdimensional computing: An introduction to computing in distributed representations with high-dimensional random vectors. *Cognitive Computation, 1,* 139-159.

Kanerva, P., Kristoferson, J., & Holst, A. (2000). Random indexing of text samples for latent semantic analysis. In *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, (p. 1036). Hillsdale, NJ: Erlbaum.

Landauer, T. K. & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review, 104,* 211-240.

Landauer, T., McNamara, D. S., Dennis, S., & Kintsch, W. (Eds.) (2007). *Handbook of latent semantic analysis.* Mahwah, NJ: Erlbaum.

Levy, S. D. (2007). Changing semantic role representations with holographic memory. (Report No. FS-07-04). In *Computational approaches to representation change during learning and development: Papers from the 2007 AAAI Symposium*. Menlo Park, CA: AAAI Press.

Pike, R. (1986). Comparison of convolution and matrix distributed memory systems for associative recall and recognition. *Psychological Review, 91*, 281-293.

Plate, T. (2003). *Holographic reduced representation: Distributed representation for cognitive structures*. CSLI Publications.

Recchia, G. L., & Jones, M. N. (2009). More data trumps smarter algorithms: Comparing pointwise mutual information to latent semantic analysis. *Behavior Research Methods*, 41(3), 647-56.

Resnik, P. (1995). Using information content to evaluate semantic similarity. In C. S. Mellish (Ed.), *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 448-453). Montréal, Canada: Morgan Kaufmann.

Risley, T. R. & Hart, B. (2006). Promoting early language development. In N. F. Watt, C. Ayoub, R. H. Bradley, J. E. Puma & W. A. LeBoeuf (Eds.), *The crisis in youth mental health: Critical issues and effective programs, Volume 4, Early intervention programs and policies*, 83-88. Westport, CT: Praeger.

Riordan, B., & Jones, M. N. (2007). Comparing semantic space models using child-directed speech. In D. S. MacNamara & J. G. Trafton (Eds.), *Proceedings of the 29th Annual Conference of the Cognitive Science Society,* 599-604.

Rubenstein, H., & Goodenough, J. (1965). Contextual correlates of synonymy. *Communications of the Association for Computing Machinery, 8*(10), 627-633.

Sahlgren, M., Holst, A., & Kanerva, P. (2008). Permutations as a means to encode order in word space. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society,* p. 1300-1305.