Scalable Techniques for Creating Semantic Vector Representations

Michael N. Jones
Departments of Psychology and Cognitive Science
Indiana University
1101 E. 10th St.
Bloomington, IN, 47405
jonesmn@indiana.edu

Gabriel Recchia
Cognitive Science Program
Indiana University
1910 E. 10th St.
Bloomington, IN, 47406
grecchia@indiana.edu

Abstract

Current vector space models for representing lexical semantics rely on sophisticated dimensional reduction operations. The complex data reduction step introduces a limitation for these models to scale up to take advantage of much larger text corpora. We examine previous work with scalable metrics, and explore the ability of scalable and incremental random vector accumulation (RVA) techniques to learn semantic representations gracefully from massive corpora. We compare RVAs to scalable metrics (PMI) as well as LSA on standard semantic evaluation tasks.

1 Introduction

Recent advances in semantic space modeling have led to increasingly sophisticated techniques for inferring reduced semantic vector representations for lexical semantics. These techniques typically begin with a term-by-document representation of a corpus in which each document (column vector) is represented as a frequency distribution over words (Salton & McGill, 1983). Next, a decomposition algorithm is commonly applied to reduce the dimensionality of the original matrix, revealing “latent” semantic components that explain the maximum amount of variance in the original matrix (i.e., explain how words are used across contexts). These models have seen remarkable success at accounting for human data in a variety of tasks, and have been used widely in practical systems that require a realistic representation of lexical semantics.

The data reduction step in these models is key to their discovery of semantic structure. The best-known example is Latent Semantic Analysis (LSA; Landauer & Dumais, 1997) use of singular-value decomposition (SVD) for dimensional reduction. More recent successful models use techniques from probabilistic inference to determine a reduced representation for word meaning from the original matrix (e.g., Griffiths, Steyvers, & Tenenbaum, 2007; Hoffman, 1999). Typically, similarity between two words is computed as the cosine of the angle (or Euclidean distance) between their reduced vectors.

These models are commonly trained on corpora with less than 20 million tokens and approximately 100k word types. However, since the emergence of these semantic space models, much larger word corpora have become available yielding more comprehensive data on which to train them.

A major limitation of popular decomposition techniques is that they are too complex to scale up to larger word corpora. Because standard algorithms for computing SVD require the term-by-document matrix to be stored in memory, training LSA on a corpus of hundreds of millions of tokens is infeasible even with high-end supercomputing resources. The problem is exacerbated by the fact that as the size of the corpus increases, the number of rows and columns in the matrix both increase significantly, the number of columns growing linearly with the number of documents and the number of rows growing approximately in proportion to the square root of the number of tokens.

A second limitation of standard decomposition techniques is the lack of incrementality: They do not update representations incrementally in response to a continual accumulation of language input but, rather, must recompute the reduced representation each time a new context (e.g. document) is added to the corpus. This makes decomposition techniques very inflexible to dynamically growing textbases. As larger text cor-
When corpora become available, it is an open question whether a better solution to semantic representation would be using a simpler algorithm that is capable of scaling up to take advantage of the larger data sample, or whether time would better be spent optimizing decomposition techniques such as SVD (cf. Banko & Brill, 2001).

2 Comparing LSA and PMI

Recently, Recchia and Jones (2009) compared the performance of LSA to the much simpler technique of pointwise mutual information (PMI; Church & Hanks, 1990) as a function of corpus size, while controlling for a host of potential confounds (see also Bullinaria & Levy, 2006). PMI can be formally defined as the joint probability of co-occurrence of two words relative to the product of their marginal probabilities in a text corpus:

\[
PMI(x, y) = \log \frac{P(x, y)}{P(x)P(y)}
\]

Recchia and Jones found that LSA yielded better predictions of human semantic data on a variety of tasks compared to PMI when both were trained on a modest-sized sample of Wikipedia. However, when PMI was scaled up to the full Wikipedia corpus, an infeasible task for LSA, it benefited from the additional data and greatly exceeded LSA’s performance from the smaller corpus. The authors conclude that simple algorithms (such as PMI) that can scale up to large amounts of data will yield a better return on invested research effort than designing better decomposition techniques that are necessarily bottlenecked by practical limitations.

Although PMI outperforms LSA cosines when it is trained on a sufficiently large corpus, it still has many shortcomings due to the fact that it is a scalar metric. Firstly, PMI cannot detect indirect similarity—that is, similarity between words that do not directly co-occur but is revealed by higher-order statistics. As such, PMI has considerably more difficulty with synonyms, which functionally should not occur in the same context, unless it is augmented with more complex methods that explicitly incorporate higher-order co-occurrences (Lemaire and Denhière, 2006). One of LSA’s strengths is learning indirect relationships that emerge as a byproduct of dimensional reduction. Secondly, PMI has been criticized because it places too much emphasis on infrequent words. Hence, the most similar word to car is not usually automobile or honda, but rather flywheel or camshaft. More recent log-likelihood adjustments to PMI compensate for this somewhat, but still suffer from the problem of infrequent co-occurrences to some extent. Most importantly, PMI is a scalar metric as is the cosine, but one cannot recover a structural representation (a multidimensional vector) from a PMI value. Structured semantic vectors are a strength of models such as LSA, and are very desirable for a variety of reasons. Firstly, vector representations allow for data compression. To store the similarity of 100k words, a \((100k)^2\) similarity matrix is necessary\(^1\). A reduced vector representation of \(D\) dimensions, however, only requires a \(100k \times D\) matrix to store the same information, and similarity can always be computed on the fly for target pairs. Secondly, vector representations are flexible: many applications attempt to estimate the meaning of a sentence or larger discourse unit as the centroid of the vectors for the words in the context. Such computational flexibility requires the vectors for the individual words and cannot be determined by pairwise scalars. Finally, structural representations are used in a variety of successful applications that use the vectors to estimate meaning in essay scoring (Foltz, et al., 1998), analogy and metaphor (Kintsch, 2001), semantic role disambiguation (Levy, 2007), and automated tutoring systems (Wade-Stein & Kintsch, 2004). All of these successful systems currently use LSA vectors. As noted earlier, however, the fidelity of LSA representations is limited by processing constraints.

Given that PMI produces greater correspondence to human semantic ratings than LSA when scaled up to larger amounts of data, the current paper explores scalable methods of creating vector representations that would display similar performance to PMI. These techniques must be computationally simple enough to efficiently scale to massive text sources, and also must adjust their representations continuously as text is added.

3 Random Vector Algorithms

Promising techniques for creating scalable vector representations from simple sums or products of random vectors have recently emerged. These techniques scale impressively to large datasets and

---

\(^1\) Granted, this is often a sparse matrix that can be efficiently searched.
are efficient to compute on a common laptop. The techniques also satisfy our previous desiderata of incrementality: they are able to gracefully update structure continuously as new data are added. In addition, random accumulation techniques are consistent with the global properties of human neural codes and change in brain structure as a function of learning (for an excellent review, see Kanerva, 2009).

3.1 Random Indexing

Rather than creating an initial term-by-document co-occurrence matrix as decomposition techniques do, recent random indexing models develop incremental random representations (for an introduction, see Sahlgren, 2005). Indexing models are heavily based on cognitive work with sparse distributed memory codes and hyperdimensional computing (Kanerva, 1988, 2009).

In these models, each word and each context is first assigned a random high-dimensional sparse vector: they contain a small proportion of +/- elements with all other elements set to zero. Each of these “index” vectors has a high probability of being orthogonal to the others, and the model takes advantage of the fact that there are many more nearly orthogonal vectors to a target than a single truly orthogonal vector. Assigning vectors in this way approximates the characteristics of orthogonality.

Once the sparse binary index vectors are constructed, a word’s lexical vector becomes the sum of the index vectors for the contexts in which it appears throughout the text corpus. Document space can also be constructed as the sum of the index vectors for words appearing in each document. Hence, random indexing does not require a dimensional reduction operation (or it can be thought of as implicit), and the model can efficiently create its representation in an incremental fashion.

Random indexing models have been demonstrated to account for a variety of results that LSA has been successfully applied to, as well as the acquisition of bilingual lexicons from parallel corpora (Sahlgren, 2006). It can be shown analytically that random indexing models are equivalent to other techniques that rely on near-orthogonal approximations, such as Random Projection (Padimtiriou et al., 1998) and Random Mapping (Kaski, 1999).

3.2 Random Vector Accumulators

Very similar to random indexing and projection are recent random vector accumulators (RVAs). The principled difference is that RVAs use dense Gaussian vectors rather than sparse binary codes. In this sense, they use far fewer elements to represent a word’s meaning and, in some cases, can be faster to compute and may have a higher storage capacity compared to random indexing.

An example of an RVA can be found in Jones’ BEAGLE model (Jones, Kintsch, & Mewhort, 2006; Jones & Mewhort, 2007). The model assigns each word an initial random signal vector, \( e_i \). Initial vector elements are sampled randomly from a Gaussian distribution with \( \mu = 0 \) and \( \sigma = 1/\sqrt{D} \), where \( D \) is the dimensionality of the vector. As a text corpus is processed, a word’s lexical memory vector, \( m_i \), is updated each time the word is encountered as the sum of the initial vectors for the other words appearing in a context with it. Hence, a word’s context in this model is defined as the other words appearing with it. Typically, this type of model considers a word’s context region to be a sentence. After the corpus is processed, a word’s lexical representation is then the sum of the random initial vectors for all words that have co-occurred with it, weighted by their frequency. Function words are typically ignored in the model by employing a standard stopword list.

The summing of random vectors is extremely efficient and allows the reduced dimensionality to be specified a priori. As with nodes in a connectionist network, a smaller dimensionality to the vector representations simply means the model will distribute the semantic information differently. Unlike LSA, each element plays an equal role in the distributed semantic representation of a word. Similar to LSA, the cosine between the \( m \) vectors of two words can be used as a similarity metric indicating their relatedness.

An RVA model is able to learn both from direct co-occurrences (as does PMI) but can also learn indirect information (as does LSA). For example, if hammer and nail frequently co-occur in sentences, then they will have common random initial vectors summed into their lexical representations, bringing them closer together in semantic space.
formally encoded as: \( \text{make excellent pets,} \) the word positional vector. Hence, in the sentence “dingoes and order information represented in a single co-occurrence mixture of context words and, hence, have the same random vectors summed into their lexical representations. Note that no syntactic or role information is employed here—similarity is simply based on the distributional hypothesis (Harris, 1951).

RVAs are very scalable and compress the semantic information for a word into a very compact distributed representation. They also have the desired incremental property: If an additional context is added to a corpus, this additional random information is simply summed into the lexical vectors. The continuous update nature of RVAs have made them particularly appropriate for modeling lexical acquisition trends in humans (Jones & Mewhort, 2007).

In addition, RVAs are suitable for representing word transition information. Jones and Mewhort (2007) refer to the above-described summation of random vectors appearing in contexts as a word’s context information. However, they note that the model can also represent a word’s order information in the same lexical vector. Order information is, loosely, information about a word’s position relative to other words in sentences (they explicitly avoid the term “syntax” since order is simply a distributed representation of n-gram co-occurrences). Jones and Mewhort accomplish this by summing circular convolutions of a word with other random vectors in context (convolution is borrowed here from signal processing). The result is a distributed vector representation reflecting the word’s n-gram history. An interesting component of the model is that the convolution coding process may be inverted to retrieve candidate words that would fit into a particular transition (the boy kicked the ____ ) from the distributed vectors.

Jones and Mewhort (2007) define a word’s lexical representation as being a mixture of context and order information represented in a single composite vector. Hence, in the sentence “dingoes make excellent pets,” the word excellent would be formally encoded as:

\[
m_i = m_i + c_i + o_i = m_i + \sum_{j=1}^{N} e_j + \sum_{j=1}^{p-1} h_{i,j} \]

\[
= m_i + \begin{bmatrix} e_{\text{dogs}} + \\ e_{\text{make}} + \\ e_{\text{pets}} \end{bmatrix} + \begin{bmatrix} (\oplus e_{\text{pets}}) + \\ (e_{\text{make}} \oplus \Phi) + \\ (e_{\text{make}} \oplus \Phi \oplus e_{\text{pets}}) \end{bmatrix} + \begin{bmatrix} (e_{\text{dogs}} \oplus e_{\text{make}} \oplus \Phi) + \\ (e_{\text{dogs}} \oplus e_{\text{make}} \oplus \Phi \oplus e_{\text{pets}}) \end{bmatrix}
\]

where \( \lambda \) represents the number of neighbors that the word can be bound with, \( p \) the position of the word in the sentence, and the asterisk-circle operator denotes directed circular convolution. Table 1 demonstrates the structure learned by context, order, and by the composite representations by listing the nearest neighbors to eat and car in the lexical space after training on the TASA corpus.

**Table 1.** Near neighbors in various RVA spaces.

**Target: EAT**

<table>
<thead>
<tr>
<th>Context</th>
<th>Order</th>
<th>Composite</th>
</tr>
</thead>
<tbody>
<tr>
<td>eaten</td>
<td>.85</td>
<td>buy .91</td>
</tr>
<tr>
<td>eating</td>
<td>.81</td>
<td>get .90</td>
</tr>
<tr>
<td>food</td>
<td>.70</td>
<td>sell .89</td>
</tr>
<tr>
<td>hunt</td>
<td>.67</td>
<td>move .89</td>
</tr>
<tr>
<td>digest</td>
<td>.65</td>
<td>save .89</td>
</tr>
<tr>
<td>grow</td>
<td>.64</td>
<td>sleep .88</td>
</tr>
<tr>
<td>need</td>
<td>.64</td>
<td>keep .88</td>
</tr>
<tr>
<td>foods</td>
<td>.63</td>
<td>swallow .88</td>
</tr>
<tr>
<td>plants</td>
<td>.62</td>
<td>win .87</td>
</tr>
<tr>
<td>insects</td>
<td>.60</td>
<td>catch .87</td>
</tr>
<tr>
<td>nutritious</td>
<td>.60</td>
<td>produce .87</td>
</tr>
</tbody>
</table>

**Target: CAR**

<table>
<thead>
<tr>
<th>Context</th>
<th>Order</th>
<th>Composite</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>.72</td>
<td>boat .95</td>
</tr>
<tr>
<td>drive</td>
<td>.71</td>
<td>ship .94</td>
</tr>
<tr>
<td>driving</td>
<td>.68</td>
<td>truck .94</td>
</tr>
<tr>
<td>road</td>
<td>.67</td>
<td>house .93</td>
</tr>
<tr>
<td>drove</td>
<td>.67</td>
<td>bus .93</td>
</tr>
<tr>
<td>wheels</td>
<td>.67</td>
<td>computer .93</td>
</tr>
<tr>
<td>truck</td>
<td>.67</td>
<td>fire .93</td>
</tr>
<tr>
<td>seat</td>
<td>.63</td>
<td>train .93</td>
</tr>
<tr>
<td>drivers</td>
<td>.62</td>
<td>bank .92</td>
</tr>
<tr>
<td>parked</td>
<td>.60</td>
<td>camera .92</td>
</tr>
<tr>
<td>cars</td>
<td>.59</td>
<td>ball .92</td>
</tr>
<tr>
<td>street</td>
<td>.59</td>
<td>dog .91</td>
</tr>
</tbody>
</table>
Hence, RVAs have a potential benefit over PMI in that they can learn indirect information and build structured vector representations. Further, they have a benefit compared to LSA in that they are scalable, incremental, and can be used to represent word order information in addition to contextual information. Several studies with humans have shown the importance of positional information in lexical processing (Jones et al., 2006; Sahlgren, Holst, & Kanerva, 2008).

In the next section, we compare RVAs to LSA and PMI on a host of commonly used human lexical semantic ratings, and explore the effect of corpus scaling on RVA performance with the goal of creating structured vector representations that have the strengths of PMI, LSA, and potential order information in a single mixture vector.

4 Evaluating RVAs

4.1 RVAs To Be Evaluated

We compared the performance of PMI, LSA, and several RVA variants on six tasks previously used in the literature to evaluate measures of semantic similarity. The concept of a word’s optimal “context,” has proven difficult to define: previous work has argued not only in favor of small windows of text adjacent to the target word (Bullinaria & Levy, 2006) but also in favor of larger “discrete meaningful units” such as sentences or coherent documents (Landauer et al., 2007). In this experiment, we take both approaches. 1024-dimensional vectors were used for each RVA, and vector cosines were used as the similarity metric in all cases. The RVAs that we evaluated were implemented as follows:

**Document-based context representation.** As described in section 3.2, the representation of a word \( w \) is simply the summation of all initial random signal vectors that correspond to the words that \( w \) has co-occurred in the same document with. In order fairly compare with Recchia and Jones (2009), who defined a document as any contiguous sequence of 10 sentences that were all part of the same article when calculating PMI estimates, this same definition of a document is used here.

**Window-based context representation.** The window-based RVA was equivalent to the context representation described above. The principal difference between this implementation and that of Jones and Mewhort (2007) is that the context regions for the window-based RVA are not meaningful sentences but are rather windows of text consisting of the \( n \) words to the right and to the left of the target word. The value of \( n \) was considered a free parameter and was optimized separately for each task, much as the number of factors \( k \) used in SVD factorization is commonly optimized separately by task in evaluations of LSA.

**Order representation.** As described in section 3.2, the representation of a word \( w \) is the summation of the convolutions of all of the initial random signal vectors \( e_1, \ldots, e_m \) that correspond to the words taking part in every \( n \)-gram surrounding \( w \). Per the summation in section 3.2, the maximum length of \( n \)-gram that \( w \) incorporates into its representation is determined by the \( \lambda \) parameter, which restricts the number of neighbors that \( w \) can be bound with. Following Jones et al. (2006), a \( \lambda \) of 5 was selected. Because the order representation is inherently window-based, we did not distinguish between document-based and window-based order representations as we did for the context representation RVAs.

**Composite representation.** As described in section 3.2, this RVA sums the context and order representations into a single vector representation for each word. The two sources are normalized to unit length prior to computing each sum.

4.2 Evaluation Materials

Accuracy was evaluated on two synonymy tests: the English as a Second Language (ESL; Turney, 2001) and the Test of English as a Foreign Language (TOEFL; Landauer & Dumais, 1997) synonymy assessments. Additionally, rank correlations to human judgments of the semantic similarity of word pairs were calculated using the similarity judgments obtained from Rubenstein and Goodenough (1965), Miller and Charles (1991), Resnik (1995), and Finkelstein et al. (2002). We abbreviate these tasks as ESL, TOEFL, RG, MC, R, and WS353, respectively.

Each of the synonymy tests consist of a collection of multiple-choice questions originally designed for non-native speakers of English. In each case, a single cue word is paired with a list of four
possible synonyms, and the test taker is required to select the word from the list that is most semantically similar to the cue word. ESL includes 50 items compiled from synonymy tests for students of English as a Second Language, while TOEFL consists of 80 retired items from the Educational Testing Service’s test of the same name. ESL has been used to evaluate metrics of semantic similarity by Turney (2001), Islam and Inkpen (2006), and Rohde, Gonnerman, and Plaut (2006). TOEFL is also a well-accepted evaluation task for metrics of semantic similarity (Bullinaria & Levy, 2006; Pado & Lapata, 2007; Rapp, 2003). Accuracy is calculated as the percentage of all questions answered correctly. In cases in which a metric of semantic similarity rates all choices as equally semantically similar to the cue, it is awarded one-fourth of a point for that question, the expected value of a correct guess were the algorithm to select one of the options randomly.

In the four remaining tasks, we generated cosine similarity values from each model for each pair of words. Following Budiu et al. (2007), we report Spearman rank correlations to human judgments for each dataset.

### 4.3 Training Materials

PMI, LSA, and all RVAs were evaluated on a 2006 version of the Wikipedia corpus segmented into sentences and articles but otherwise stripped of formatting, markup, and nonalphabetic characters by Willits, D’Mello, Duran, and Olney (2007). To explore the effect of corpus size while controlling for corpus type, we also evaluated the models on a much smaller subset of the Wikipedia corpus. 37,600 documents from Wikipedia were sampled randomly without replacement from the full corpus. To ensure a fair comparison with the PMI results of Recchia & Jones (2009), a document was again defined as a contiguous sequence of 10 sentences from the same article. This yielded a subset of Wikipedia representing a wide variety of topics and articles, with 37,600 documents that averaged 162 words each. The subset was created to reflect the characteristics of standard corpora (e.g., TASA) on which LSA is commonly trained. The entire subset contained 6,102,845 tokens and 251,703 types, in contrast to 417,775,181 tokens and 3,404,652 types in the full Wikipedia.

### 4.4 Design and Procedure

We first trained our four RVAs on the small Wikipedia subset of 37,600 documents. Next, we scaled up to the full Wikipedia corpus. As previously discussed, LSA could be trained on the subset only due to the complexity of SVD and the fact that LSA’s memory requirements increase exponentially as the size of the corpus is increased. Although RVAs do not have this issue with memory requirements, the convolution operation used in the order and composite representations proved too time-consuming for us to train them on the entire Wikipedia corpus. Thus, for the full Wikipedia, results are only reported for the document-based context representation, the window-based context representation, and PMI.

### 5 Results

Table 2 displays the results on all six tasks for each model trained on the subset of Wikipedia. LSA outperforms all of the other methods on the smaller corpus, presumably because it is able to induce more information from the smaller dataset with SVD. PMI performed only slightly worse than LSA on the smaller corpus. Both LSA and PMI heavily outperformed all of the RVAs on the Wikipedia subset. One interesting pattern to note, however, is that the addition of order information to an RVA allows it to outperform all other models on synonymy tests such as the TOEFL. The order information constrains the RVA to favor paradigmatic similarity useful to synonymy tasks—this constraint is not possible in either LSA or PMI.

<table>
<thead>
<tr>
<th></th>
<th>PMI</th>
<th>LSA</th>
<th>Context (Docs)</th>
<th>Context (Window)</th>
<th>Order</th>
<th>Composite</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESL</td>
<td>.35</td>
<td>.36</td>
<td>.28</td>
<td>.36</td>
<td>.20</td>
<td>.24</td>
</tr>
<tr>
<td>TOEFL</td>
<td>.41</td>
<td>.44</td>
<td>.33</td>
<td>.40</td>
<td>.46</td>
<td>.47</td>
</tr>
<tr>
<td>RG</td>
<td>.46</td>
<td>.46</td>
<td>.14</td>
<td>.33</td>
<td>.07</td>
<td>.10</td>
</tr>
<tr>
<td>MC</td>
<td>.47</td>
<td>.62</td>
<td>.12</td>
<td>.39</td>
<td>.08</td>
<td>.09</td>
</tr>
<tr>
<td>R</td>
<td>.46</td>
<td>.60</td>
<td>.17</td>
<td>.44</td>
<td>.06</td>
<td>.09</td>
</tr>
<tr>
<td>WS353</td>
<td>.54</td>
<td>.57</td>
<td>.20</td>
<td>.31</td>
<td>.13</td>
<td>.23</td>
</tr>
</tbody>
</table>

Table 3 reports the results of PMI, the document-based RVA, and the window-based RVA when trained on the full Wikipedia corpus. Numbers in parentheses represent the window size that
yielded the optimum score for the window-based context representation.

Table 3. Model performance on the full corpus.

<table>
<thead>
<tr>
<th></th>
<th>PMI</th>
<th>Context (Docs)</th>
<th>Context (Window)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESL</td>
<td>.62</td>
<td>.44</td>
<td>.46 (n=20)</td>
</tr>
<tr>
<td>TOEFL</td>
<td>.64</td>
<td>.54</td>
<td>.71 (n=10)</td>
</tr>
<tr>
<td>RG</td>
<td>.78</td>
<td>.62</td>
<td>.64 (n=10)</td>
</tr>
<tr>
<td>MC</td>
<td>.86</td>
<td>.68</td>
<td>.60 (n=10)</td>
</tr>
<tr>
<td>R</td>
<td>.76</td>
<td>.74</td>
<td>.76 (n=50)</td>
</tr>
<tr>
<td>WS353</td>
<td>.73</td>
<td>.45</td>
<td>.46 (n=14)</td>
</tr>
</tbody>
</table>

When scaled up to the full corpus (an impossible feat for LSA due to the complexity of SVD), the RVAs are able to capitalize on the additional data and now outperform LSA on every task except WS353. However, the RVAs are still outperformed by PMI, which also scales gracefully to take advantage of the additional data. Interestingly, the optimized window version RVA outperformed PMI on the TOEFL task. Presumably, this result is partially due to the restricted context (causing a focus on paradigmatic information), and partially due to the fact that RVAs can capitalize on indirect information necessary for synonymy tasks.

Generally, the data suggested three major differences between the models. Firstly, the RVAs are able to capitalize on indirect similarity, and this information gives them useful variance for synonymy tests. Secondly, RVAs do not take into consideration the base frequency (marginal probability) of words. Hence, words may become similar to one another simply due to chance co-occurrences with high-frequency words. RVAs generate more plausible lists of near neighbors than PMI (see car example in Section 2), but they clearly need a way of filtering out the effects of chance co-occurrence due to frequency as PMI does. Thirdly, it is desirable to incorporate order information into a vector representation for a variety of tasks, particularly synonymy. Order information gave the largest boost to synonym tests in the Wikipedia subset corpus. However, convolution was too computationally complex to apply to the full Wikipedia corpus; in the discussion we comment on some potential techniques to estimate order information with new scalable techniques.

6 Discussion

In this paper we take a small step towards exploring RVAs as scalable techniques to create reduced semantic vector representations from distributional information in a text corpus. Compared to scalar metrics, RVAs offer the advantage of creating structural representations, and the ability to take advantage of indirect statistical associations. However, the simple technique for building RVAs needs modification to capture useful variance contained in scalar metrics such as PMI. Compared to decomposition techniques such as LSA, RVAs offer the advantage of being able to scale up to take advantage of much larger datasets when constructing their representations. In addition, they have the ability to encode a word’s paradigmatic order information together with its contextual information in a composite mixture vector.

RVAs and PMI are clearly capitalizing on different types of information. For all of the tasks presented here, RVA cosines and PMI values correlated more highly with the human ratings than they did with each other. Future attempts to build RVAs can benefit from trying to capitalize on the unique variance that PMI is using to find word similarities.

In our study, RVAs were dramatically outperformed by the LSA cosines when trained on the smaller corpus. When scaled up to the larger corpus, however, they exceeded LSA’s performance on all tasks (with the exception of WS353). Comparison to PMI trained on the larger corpus suggests that RVAs could be enhanced by weighting marginal probabilities of occurrence when summing the random initial vectors in learning. However, it should be noted that RVAs will often produce a much more realistic pattern of nearest neighbors because they are not as easily swayed by co-occurrences with infrequent words as PMI is.

In addition, RVAs can outperform PMI particularly when the task requires indirect statistical relationships, such as the TOEFL task. The addition of order information further enhances the performance of RVAs on synonym tests on the small corpus. This pattern implies that performance on synonymy tests could be significantly boosted if an RVA with order information was scaled up to a larger corpus. Unfortunately, the circular convolution operation used to encode distributed n-grams
is too computationally complex to make it appealing as a candidate to scale up to large corpora.

6.1 Alternative Techniques

The criticism that PMI fails to take into account indirect statistical relationships has already been addressed by a number of authors. For example, PMI has been specifically augmented to take advantage of higher-order statistical information (Islam & Inkpen, 2006; Lemaire & Denhière, 2006), however, PMI remains a scalar metric and lacks the flexibility of structured vector representations. Gorman and Curran (2006) provide a review of scalable algorithms for encoding distributional similarity including heuristic, vector-based, and hierarchical systems.

A simple alternative is to reverse-engineer the vector representation from the scalar metric. Since RVAs distribute the semantic representation evenly across random elements, the elements themselves have no particular meaning—only the pattern does. One could simply use an algorithm to generate two vectors with the desired structure directly from the PMI value. However, the construction of these vectors rapidly increases in complexity as desired similarity structure increases beyond pairs. In addition, the time course of semantic organization as a function of training is lost. Most importantly, pairwise measures cannot be used to estimate the similarity of larger discourse contexts as the centroid of component vectors (as is commonly used in many current intelligent systems).

A major drawback of RVAs, however, is that they do not make use of the full vector space. The random initial vectors have an expected cosine of zero with each other. Since the lexical vectors are sums of these initial vectors, and all lexical vectors represent English word types, final lexical vectors are unlikely to have zero similarity to each other and will not be negatively related. Hence, the range of cosines is usually between 0.1 and 1.0 in an RVA. LSA has similar range restriction issues with few word pairs ever dipping below a cosine of zero to each other. An optimal solution would take advantage of the full similarity space.

6.2 Efficient Integration of Order

When trained on the smaller corpus, the RVA that incorporated contextual similarity and order information (via vector convolutions) greatly outperformed the other models on the TOEFL task. This pattern is similar to the finding that order information greatly improves the fit to human semantic priming data (Jones et al., 2006), and suggests that scaling order up to larger corpora could benefit synonymy relationships. However, the circular convolution operation used to encode distributed n-grams is relatively inefficient, scaling in $O(n^2)$ time, where $O$ also increases as a nonlinear function of window length. This can be sped up to $O(n \log(n))$ time using a fast-Fourier transform and operating in the frequency domain, but it is still intractable as an order encoding operation.

A potentially promising solution to this complexity issue is the use of random permutations to integrate order information into vector representations (Kanerva, 2009; Sahlgren et al., 2008). This technique approximates the structure learned by convolution with efficient permutations of initial vector elements to represent the other words that appear in positions around a target word in a distributed fashion. Preliminary work suggests that random permutations encode very similar information to convolutions (Sahlgren et al.), and scale gracefully to large datasets. Random permutations work equally well with sparse binary vectors (from Random Indexing) or dense Gaussian vectors (from RVAs).

6.3 Conclusions

Structured semantic vector representations are useful in a variety of applications, from input representations for computational models of discourse comprehension to applied systems for search and flexible data compression. Our work here mirrors other areas in computational linguistics (cf. Banko & Brill, 2001), suggesting that more progress can be made by exploring simple systems that can scale up to large text corpora when constructing vector representations as opposed to focusing effort on better dimensional reduction techniques. RVAs are an appealing candidate. However, much more work is needed to allow RVAs to represent variance inherent in currently popular scalar metrics such as PMI.
References


