Machine Learning for Language Processing
Lecture 6: Vector Space Models of Semantics

Stephen Clark

November 5, 2015

Much of the content in these notes is taken from my book chapter on Vector Space Models of Lexical Meaning. (See Readings for Today's Lecture.)

VSMs in Document Retrieval The document retrieval problem in Information Retrieval (IR) [3] is as follows: given a query — typically represented as a set of query terms — return a ranked list of documents from some set of documents, ordered by relevance to the query. Terms here can be words or lemmas, or multi-word units, depending on the lexical pre-processing being used.

One of the features of most solutions to the document retrieval problem, and indeed Information Retrieval problems in general, is the lack of sophistication of the linguistic modelling employed: both the query and the documents are considered to be "bags of words", i.e. multi-sets in which the frequency of words is accounted for, but the order of words is not. However, this simplifying assumption has worked surprisingly well, and attempts to exploit linguistic structure beyond the word level have not usually improved performance. For the document retrieval problem perhaps this is not too surprising, since queries, particularly on the web, tend to be short (a few words), and so describing the problem as one of simple word matching between query and document is arguably appropriate.

Once the task of document retrieval is described as one of word overlap between query and document, then a vector space model is a natural approach: the basis vectors of the space are words, and both queries and documents are vectors in that space. The coefficient of a document vector for a particular basis vector, in the simplest case, is just the number of times that the word corresponding to the basis appears in the document. Queries are represented in the same way, essentially treating a query as a "pseudo-document".

Term-Frequency Model The figure on the slide gives a simple example containing two short documents. In this example the user is interested in finding

documents describing the England cricketer, Matthew Hoggard, taking wickets against Australia, and so creates the query { *Hoggard, Australia, wickets* }. Here the query is simply the set of these three words. The vectors are formed by assuming the basis vectors given in the term vocabulary list at the top of the figure (in that order); so the coefficient for the basis vector *Hoggard*, for example, for the document vector $\overrightarrow{d1}$, is 2 (since *Hoggard* occurs twice in the corresponding document).

The function for calculating the similarity between query and document is the dot product between the corresponding vectors, which is essentially measuring term overlap between the two.

TF-IDF Model The point of the example is to demonstrate a weakness with using just term-frequency as the vector coefficients: all basis vectors count equally when calculating similarity. In this example, document $d2$ matches the query as well as $d1$, even though $d2$ does not mention Hoggard at all. The solution to this problem is to recognise that some words are more indicative of the meaning of a document (or query) than others. An extreme case is the set of function words: we would not want a query and document to be deemed similar simply because both contain instances of the word "the".

Continuing with the example, let us assume that the document set being searched contains documents describing cricket matches. Since *wicket* is likely to be contained in many such documents, let us assume that this term occurs in 100 documents in total. *Hoggard* is more specific in that it describes a particular England cricketer, so suppose this term occurs in only 5 documents. We would like to down-weight the basis vector for *wicket*, relative to *Hoggard*, since *wicket* is a less discriminating term than *Hoggard*. An obvious way to achieve this is to divide the term-frequency coefficient by the corresponding document frequency (the number of documents in which the term occurs), or equivalently multiply the term frequency by the *inverse document frequency* (IDF). The figure on the slide shows the simple example with IDF applied (assuming that the document frequency for *Australia* is 10 and *collapse* is 3). Document $d1$ is now a better match for the query than $d2$.

Finally, there is one more standard extension to the basic model, needed to counter the fact that the dot product will favour longer documents, since these are likely to have larger word frequencies and hence a greater numerical overlap with the query. The extension is to normalise the document vectors (and the query) by length, which results in the cosine of the angle between the two vectors.

Term-Document Matrix One useful way to think about the document vectors is in terms of a *term-document* matrix. This matrix is formed by treating each term or word vector as a row in the matrix, and each document vector as a

column. The figure on the slide shows the term-document matrix for our simple running example.

The main reason for introducing the term-document matrix is that it provides an alternative perspective on the co-occurrence data, which will lead to vectors for terms themselves. But before considering this perspective, it is important to mention the potential application of *dimensionality reduction* techniques to the matrix, such as singular value decomposition (SVD). The use of SVD, or alternative dimensionality reduction techniques such as non-negative matrix factorisation or random indexing, will not be described here. However, it is important to mention these techniques since they are an important part of the linear algebra toolbox which should be considered by any researchers working in this area. Chapter 18 of [3] provides an excellent textbook treatment of matrix decompositions in the context of Information Retrieval.

The second term-document matrix example (with term frequencies as the values) has more documents and provides a better example for what follows. The first insight is that documents can be clustered based on the terms they contain. So in the example, the c's cluster together, because they tend to contain the same terms; likewise for the m's. But now an alternative suggests itself: rather than looking at the similarity of the columns, consider the similarity of the rows. So we can now say that terms are similar if they tend to occur in the same documents. How similar? The cosine measure can be applied here as well.

A Finer Notion of Context The term-document matrix introduced above gives us the basic structure for determining word similarity. There the intuition was that words or terms are similar if they tend to occur in the same documents. However, this is a very broad notion of word similarity, producing what we might call topical similarity, based on a coarse notion of context. The trick in arriving at a more refined notion of similarity is to think of the term-document matrix as a term-*context* matrix, where, in the IR case, context was thought of as a whole document. But we can narrow the context down to a sentence, or perhaps even a few words either side of the target word.

Once the context has been shortened in this way, then a different perspective is needed on the notion of context. Documents are large enough to consider which words appear in the same documents, but once we reduce the context to a sentence, or only a few words, then similar words will tend not to appear in the same *instance* of a contextual window. The new perspective is to consider single words as contexts, and count the number of times that a context word occurs in the context of the target word.

The example on the slide demonstrates the construction of a *word-word* (or *term-term*) co-occurrence matrix, using a single sentence as the context window. Note that the target words — the words for which context vectors are calculated — do not have to be part of the term vocabulary, which provide the context. Determining which words are similar can be performed using the cosine measure, as before.

Alternative Definitions of Context The previous example uses what is often called a *window* method, where the contextual words for a partiular instance are taken from a sequence of words containing the target word. In the example, the window boundaries are provided by each sentence. When the window is as large as a sentence — or a paragraph or document — the relation that is extracted tends to be one of topical similarity, for example relating *gasoline* and *car*. If the intention is to extract a more fine-grained relation, such as synonymy, then a smaller, and more fine-grained, notion of context is appropriate.

More fine-grained defintions are possible even for the window method. One possibility is to pair a context word with a direction. Now the vector coefficients are weighted counts of the number of times the context word appears to the left, or right, of the target word, respectively. A further possibility is to take position into account, so that a basis vector corresponds to a context word appearing a particular number of words to the left or right of the target word. Whether such modifications improve the quality of the extracted relations is not always clear, and depends on the lexical relations that one hopes to extract.

The next step in refining the context definition is to introduce some linguistic processing. One obvious extension to the window methods is to add part-of-speech tags to the context words. A more sophisticated technique is to only consider context words that are related syntactically to the target word, and to use the syntactic relation as part of the definition of the basis vector. The figure on the slide shows how these various refinements pick out different elements of the target word's linguistic environment. The pipe or bar notation (|) is simply to create pairs, or tuples — for example pairing a word with its part-of-speech tag. The term *contextual element* is used to refer to a basis vector term which is present in the context of a particular instance of the target word.

The intuition for building the word vectors remains the same, but now the basis vectors are more complex. For example, in the grammatical relations case, counts are required for the number of times that *goal*, say, occurs as the direct object of the verb *scored*; and in an adjective modifier relation with *first*; and so on for all word-grammatical relation pairs chosen to constitute the basis vectors. The idea is that these more informative linguistic relations will be more indicative of the meaning of the target word.

Weighting So far we have been assuming that all contextual elements are equally useful as basis vectors. However, this is clearly not the case: the word *the* provides very little information regarding the meaning of the word *goal*. Here we can adopt the idea of weighting from document retrieval. One simple method for weighting a basis vector is to divide the corresponding term frequency by the number of times that the term occurs in a large corpus, or the number of documents that the term occurs in (similar to IDF).

The figure on the slide demonstrates the effect of using IDF in this way for the extreme case of *the* as a basis vector. The effect is a little hard to capture in two

dimensions, but the idea is that, in the vector space to the left of the figure, the vectors for *dog* and *cat* will be pointing much further out of the page — along the *the* basis — than in the vector space on the right. A useful intuition for the effect of IDF is that it effectively "shrinks" those basis vectors corresponding to highly frequent terms, reducing the impact of such bases on the position of the word vectors, and thereby reducing the amount of overlap on those bases when calculating word similarity.

One feature of IDF is that the shrinking effect applies in the same way to all target words (since IDF for a basis vector is independent of any target word). However, we may want to weight basis vectors differently for different target words. For example, the term *wear* may be highly indicative of the meaning of *jacket*, but less indicative of the meaning of *car*. Hence we would want to emphasise the basis vector for *wear* when building the vector for *jacket*, but emphasise other basis vectors — such as *gasoline* — when building the vector for *car*. Curran [1] uses collocation statistics, such as pointwise mutual information, to allow the weighting scheme to have this dependence on the target word. For example, *jacket* and *wear* will be highly correlated according to a collocation statistic because *jacket* co-occurs frequently with *wear* (relative to other basis vector terms).

Evaluation How to evaluate semantic vector space models is a hot topic at the moment. The standard approach is to use a set of human similarity judgements, and calculate a spearman correlation coeffient between the rankings of pairs induced by the human judgements, and those induced by the automatic similarity metric. However, most researchers agree this method is inadequate, not least because the notion of similarity, outside of any context, is not very meaningful. (This is especially true when moving from single words to phrases or even sentences.) The example judgements on the slide are from the frequently used wordsim 353 dataset.

Another problem is that semantic similarity can be interpreted in a variety of ways. Hence a number of researchers have created alternative datasets, being careful to distinguish semantic similarity — e.g. (*car, van*) — and semantic relatedness — e.g. (*car, petrol*). Examples include Simlex-999 and MEN.

One final comment is that the models introduced have a large number of parameters (window size, weighting scheme, and so on). How best to set these paramaters, and whether some settings are better than others for various semantic relations or evaluations, is another hot topic. For a recent paper attempting to investigate this question, see [2].

Readings for Today's Lecture A preprint of my book chapter is available online.

- Vector Space Models of Lexical Meaning. Stephen Clark. *Handbook of*

Contemporary Semantics second edition, edited by Shalom Lappin and Chris Fox. Chapter 16, pp.493-522. Wiley-Blackwell, 2015

- From frequency to meaning: Vector space models of semantics. Peter D. Turney and Patrick Pantel. *Journal of Articial Intelligence Research* 37:141-188. 2010

References

[1] James R. Curran. *From Distributional to Semantic Similarity.* PhD thesis, University of Edinburgh, 2004.

[2] Douwe Kiela and Stephen Clark. A systematic study of semantic vector space model parameters. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC) at EACL 2014*, pages 21–30, 2014.

[3] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.