Combining Symbolic and Distributional Models of Meaning

Stephen Clark and **Stephen Pulman**
Oxford University Computing Laboratory
stephen.[clark|pulman]@comlab.ox.ac.uk

Abstract

There are two main approaches to the representation of meaning in Computational Linguistics: a symbolic approach and a distributional approach. This paper considers the fundamental question of how these approaches might be combined. The proposal is to adapt a method from the Cognitive Science literature, in which symbolic and connectionist representations are combined using tensor products. Possible applications of this method for language processing are described. Finally, a potentially fruitful link between Quantum Mechanics, Computational Linguistics, and other related areas such as Information Retrieval and Machine Learning, is proposed.

Introduction

Representing the meanings of words and sentences in a form suitable for use by a computer is a central problem in Computational Linguistics (Jurafsky & Martin 2000). The problem is of theoretical interest – to linguists, philosophers and computer scientists – but also has practical implications. Finding a suitable meaning representation can greatly improve the effectiveness of a Natural Language Processing (NLP)[1] system, whether it be for automatically translating sentences from one language to another, answering questions, or summarising articles (to give just three examples).

There have been two distinct approaches to the representation of meaning in NLP. The first, the *symbolic* approach, follows the tradition of Montague in using a logic to express the meanings of sentences (Dowty, Wall, & Peters 1981). The logical representation of a sentence is built up compositionally by combining the meanings of its constituent parts. In contrast, the *distributional* approach uses statistics about the contexts in which a word is found, extracted from a large text corpus, to provide a vector-based representation of the meaning of an individual word. Thus the symbolic approach is largely concerned with how individual meanings are combined, but leaves the meanings of the elementary units (the words) unanalysed; whereas the distributional approach is concerned with the meanings of words, but has little to say about how these meanings combine. Given the importance in NLP of both these approaches to natural language semantics, a fundamental question is whether they can be integrated in a useful and meaningful fashion. This is an interesting theoretical question, but also has practial consequences, some of which will be described later in the paper.

[1]The terms *Computational Linguistics* and *Natural Language Processing* will be used interchangeably in this paper.

Representations in Cognitive Science

A similar question has been addressed by the Cognitive Science community, where two distinct approaches to the computational modelling of the mind have been developed.[2] The first, the *connectionist* approach, models the mind/brain as a massively interconnected network of simple processing units, all operating in parallel. Each unit has an activation value and is connected to other units through connections, each with a corresponding weight. The activation value is calculated according to some simple formula involving the activation values of neighboring units and the connection weights. The framework used to manipulate these objects is the mathematics of vector and matrix algebra.

The second approach is to treat the mind as a serial device which manipulates discrete structures consisting of abstract symbols. The remarkable productivity of the mind, which enables it to process an unbounded number of distinct inputs, is explained by allowing the device to combine the symbols in an infinite number of ways. A primary example of this *combinatorial* strategy comes from linguistics, in which combinatorial theories of syntax, based on the seminal work of (Chomsky 1957), explain how humans can produce and comprehend an unbounded number of sentences. Grammars consisting of a finite number of rewrite rules, some of which are recursive, can model the hierarchical structures of these sentences. The meaning of a sentence can then be computed so as to satisfy the principle of *compositionality* – the meaning of a sentence is a function of the meanings of its components, the semantic analogue of syntactic creativity. Here the mathematics of formal language theory and logic provide the theoretical framework.

(Smolensky & Legendre 2006) integrate the connectionist and symbolic models in the following way. First,

> a symbolic structure **s** is defined by a collection of **structural roles** $\{r_i\}$ each of which may be occupied

[2]The following description of the two approaches is based on (Smolensky & Legendre 2006).

by a **filler** $\mathbf{f}_i$; **s** is a set of **constituents**, each a **filler/role binding** $\mathbf{f}_i/r_i$. (p.66)

Linguistics again provides primary examples of such structures: many of the syntactic representations used in linguistics – whether they be parse trees, dependency trees or functional structures more closely associated with predicate argument structure – can be thought of as sets of **constituents**.

The key question is how to take connectionist representations of the roles and fillers, which are vectors of activation values, and produce a connectionist representation of the symbolic structure. Smolensky and Legendre's solution is to use a **tensor product** representation:

$$\mathbf{s} = \sum_i \mathbf{f}_i \otimes \mathbf{r}_i \quad (1)$$

The activity vector for the complete structure is the superposition (vector sum) of the constituent vectors, each of which is a tensor product of the activity vector for the filler and the activity vector for the structural role occupied by the filler. Smolensky and Legendre note that, in cognitive domains such as language, the representations are recursive: the fillers and roles of **s** can themselves be tensor product representations made up of other fillers and roles.

(Smolensky & Legendre 2006) argue at length for why the tensor product is appropriate for combining connectionist and symbolic structures. A number of other proposals for realizing symbolic structures in vector representations are described. Smolensky's claim is that, despite appearances to the contrary, all are special cases of a generalized tensor product representation.

(Aerts & Gabora 2005) also propose use of the tensor product, in order to combine vector-based representations of concepts. More specifically they are interested in the "pet fish" problem, a well-known phenomenon in psychology where subjects typically rate a guppy as a poor example of a fish, a poor example of a pet, but a good example of a pet fish. This observation has been used to attack the claim that *prototypes* (Rosch & Mervis 1975) can serve as word senses. While the denotation of *pet fish* is arguably the intersection of the set of pets and the set of fish, the prototypes do not seem to combine in a straightforwardly compositional way. Aerts and Gabora, however, claim that the procedure used for combining two quantum entities (the tensor product) can be used to combine pet and fish to yield the desired properties of pet fish.

Symbolic Representations of Meaning

Symbolic meaning representations are usually expressed in some form of logic, or knowledge representation language. Word senses are usually, though not invariably, treated as primitive predicates of the logic with fixed denotations, and any purely word-related inferences captured by axioms. In some implemented NLP systems (Bos 2005) these axioms are derived automatically from community resources like WordNet. Parsed sentences are translated into logical forms via some kind of syntax-directed translation technique; this is a fully compositional process, combining the meanings of words into meanings for phrases and then into meanings for whole sentences. Typically, only fully parsed sentences can be interpreted, since the translation process is guided by the syntactic structure of the sentence. In principle, the resulting logical representations can then be used in conjunction with a theorem prover (often this requires a further translation step into first order logic, where possible) or directly transformed to executable code like Prolog or SQL.

This is in effect the "classical AI" approach to meaning representation and, as such, symbolic approaches have often been criticised for their lack of robustness and scalability: implemented systems tend to be small scale and domain specific. However, the recent improvements in wide-coverage parsing (Clark & Curran 2004) have led to attempts to scale up symbolic approaches, with some success (Bos 2005).

Distributional Representations of Meaning

Document Content

Vector space models of document content originated in information retrieval (Sparck Jones & Willett 1997). A document is represented as a vector in a high-dimensional "information space", with the words from a vocabulary forming a set of orthogonal basis vectors. The vocabulary is typically the set of words found in a large document collection. The value of each component of the document vector is intended to represent the degree to which the corresponding vocabulary word is indicative of the document's content or meaning. This is typically implemented using a term frequency-inverse document frequency (TF-IDF) statistic. Term frequency is the number of times the vocabulary word appears in the document, and inverse document frequency is the reciprocal of the total number of documents in which the word appears (given some document collection). IDF is intended to measure the importance of a word in determining the content of a document, or distinguishing one document from another.

One advantage in representing documents as vectors is that semantic similarity between documents is easily modelled as an inner product in the vector space. The document vectors are typically normalised so that documents of different lengths can be meaningfully compared, in which case the inner product is the cosine of the angle between the vectors. This geometric interpretation of document content is appealing, and provides solutions to a number of IR problems. For example, the document retrieval problem (the main problem addressed by Google and other search engines) can be effectively solved by representing a query as a "mini-document" in the document space, and measuring the relevance of each document to the query using the cosine measure.

This simple model of a document's content is known as a *bag-of-words* model, because any higher level relations between words – even linear order – are ignored. The bag-of-words model is surprisingly effective for the document retrieval problem and, despite many years of research attempting to improve on it, still provides state-of-the-art performance. However, there are now IR tasks such as Question Answering (QA) in which more sophisticated linguistic representations have proven useful.

Thesaurus Extraction

It seems obligatory in a paper on distributional similarity to quote Firth's famous dictum that "you shall know a word by the company it keeps". This neatly expresses the idea behind the *distributional hypothesis*, namely that words with similar meanings appear in similar contexts. Intuitively, *car* and *motorbike* have similar meanings because cars and motorbikes can be driven, bought, sold, crashed, and so on.

Such a model can be implemented by taking a large body of text and creating context vectors for each *headword* of interest. The context can be a simple window surrounding the headword, or a more linguistically-motivated context consisting of grammatical relations (Grefenstette 1992). To rephrase the previous example, *car* and *motorbike* have similar meanings because *car* and *motorbike* both appear as the direct object of *drove*, *bought*, *sold*, *cleaned*; as the subject of *sped*, *crashed*; as the modifiee of *red*, *new*, *old*; and so on.

Each component of the context vector represents a particular context, for example the direct object of the verb *buy*. The value for each component is the number of times the headword is seen in that context. This value can be further weighted to take into account how indicative the context is of the headword's meaning, in a similar way to the use of IDF in document retrieval. Headwords can then be compared using the cosine, or some other, measure. (Curran 2004) applies these methods to a large corpus (two billion words) to create a noun thesaurus: an entry is created for each noun headword consisting of the top-N most similar headwords, for some suitable value of N.

(Schütze 1998) extends these methods to represent the content of words, i.e. word senses. He proposes a method in which word senses are represented by second order vectors. These are derived in two steps. First, each target word is represented by a co-occurrence vector, as above; however, these first order vectors will be noisy where the word has more than one sense, since these will typically occur in different contexts: for example 'bond' will occur in a financial context, as well as an "imprisonment" context. To counter this, for each headword occurrence Schütze computes the centroid of the vectors for each of the words within the context window. This effectively averages the direction of the words in the context. The resulting context vectors are then clustered, with each cluster hopefully representing a different sense of the target word. This model of word senses is shown to perform well in word sense disambiguation tasks.

It is interesting to consider the vector space model of word meaning by the usual criteria for theories of word sense, since it is a direct implemention of the "distributional" theory, advocated most famously by linguists like Firth, quoted above, Zellig Harris and Maurice Gross, or as a version of the "holistic" theory put forward by various philosophers: Quine, Davidson, and Churchland. As (Fodor & Lepore 1999) argue in a recent attack on such theories, it is not at all obvious that they assign to word senses the most basic requirement: that they should support a compositional model of semantic interpretation, or be preserved in translation.

Some limited kind of compositionality has perhaps been demonstrated by (Widdows 2004) who showed that a kind of negation on word senses could be defined in vector space

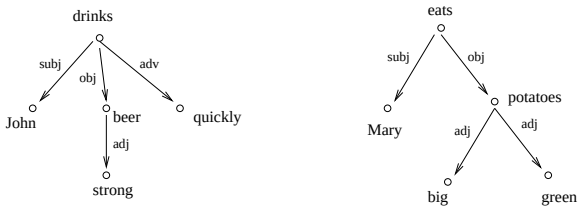

Figure 1: Example dependency trees

models. A concept like *suit* NOT *lawsuit* (i.e. the clothing sense) can be defined by projecting the first order vector for *suit* (which will smear all the senses together) onto the orthogonal subspace for *lawsuit*. Under this model the most similar concepts to *suit* included terms like *plaintiff* and *sued* whereas the most similar to *suit* NOT *lawsuit* were concepts like *pants*, *shirt*, and *jacket*.

But this is not enough to satisfy real compositionality. To put the problem simply, can we find a way of combining the vectors for individual words (or word senses) in such a way that the representation for a sentence like *the dog bit the man* will be different from that for *the man bit the dog*? Or, in more detail, that the representation for the first sentence should bear some revealing relationship to that for *the animal bit the human*? A solution to this problem has practical as well as theoretical interest. Calculating semantic similarity between sentences, or larger discourse units, is currently a hot topic in NLP, and has many applications, including Question Answering, textual entailment, and multi-document summarisation, to name a few.

The Combination

The proposal here is to adopt the tensor product representations used by Smolensky for combining the symbolic and distributional meaning representations. The symbolic representation, at the sentence level, could be a parse tree, a dependency graph, a set of predicate argument relations: essentially any structure that can be represented in terms of filler-role bindings. The distributional representation, at the word level, is assumed to be some context vector.

Figure 1 contains dependency trees for the sentences *John drinks strong beer quickly* and *Mary eats big green potatoes*. The kind of tensor product representation we have in mind for the first sentence is as follows:

$$\textit{drinks} \otimes \text{subj} \otimes \textit{John} \otimes \text{obj} \otimes (\textit{beer} \otimes \text{adj} \otimes \textit{strong}) \otimes \text{adv} \otimes \textit{quickly}$$

where the order of the products is determined by some canonical traversal of the tree. How to obtain vectors for the dependency relations – subj, obj, etc. – is an open question. For now we will simply assume that the dependency relations form an orthonormal basis in a "relation space", so that each relation has length one and the inner product between two different relations is zero.

What have we achieved with this representation? First, the vector for *dog bites man* is not the same as the vector for *man bites dog*. Second, the representation allows us to compare sentences such as *man reads magazine* and *woman browses newspaper* in a straightforward way using the fol-

lowing property of the tensor product:

$$(w_1 \otimes w_2).(w_3 \otimes w_4) = (w_1.w_3) \times (w_2.w_4) \qquad (2)$$

The similarity can be calculated by simply comparing the respective pairs in each argument slot and multiplying the inner products together:

$$\textit{man.woman} \times \textit{reads.browses} \times \textit{magazine.newspaper}$$

The inner products between the respective dependency relations will be one, because the corresponding relations are the same. One consequence of equation 2 is that similarity between these vectors can be calculated without building the tensor product representations. This is a significant feature for any implementation because the number of basis vectors in the space $V \otimes W$ is the size of V times the size of W; and since the context vectors representing the words could have tens of thousands of components, the vector product representations quickly become too large to build explicitly.

One weakness of the representation is that it is only possible to compare dependency trees of the same type, where type here means the tuple of dependency relations obtained by traversing the tree in some canonical order. This means, for example, that *the dog bit the man* cannot be compared with *the man was bit by the dog*. This problem can be overcome by using a more abstract representation in which the passive and active forms of verbs are identified. However, this approach will only work for syntactic alternations, and will not allow the sentences in Figure 1 to be compared.

A possible solution to this problem is to use a *convolution kernel* (Haussler 1999). A convolution kernel between two structures considers all possible ways of decomposing those structures, and sums up the similarities between all the pairwise decompositions. In the case of the dependency trees, this would involve decomposing each tree into all its subtrees (this being a natural decomposition of a tree) and summing the similarities between all pairs of subtrees. For this particular example, this would allow *John drinks* to be compared with *Mary eats*; *drinks beer* to be compared with *eats potatoes*; *John drinks beer* to be compared with *Mary eats potatoes*; *strong beer* to be compared with *green potatoes*; and so on for all the subtrees. Since the number of subtrees grows exponentially with the size of the tree, the subtrees cannot be listed explicitly. A possible solution to this problem is to extend the tree kernel of (Collins & Duffy 2002), which is a dynamic programming method for calculating the number of common subtrees between two trees.

Conclusions

This paper has been submitted to a symposium on Quantum Mechanics (QM) and AI, so what is the connection to QM? The mathematical theory of QM is based on Hilbert spaces; the objects in which we have situated word meanings are Hilbert spaces. The operator we have proposed for combining word meanings is the tensor product; composite systems in QM, formed by interacting quantum-mechanical systems, are represented using tensor products (Hughes 1989). This link suggests that work in NLP which uses vector spaces may benefit from borrowing more from the well-developed mathematical theory of QM.

Other possible links exist. NLP is currently dominated by probabilistic models. How to integrate probabilistic models with the distributional models described here is another interesting question. The mathematical theory of QM, as well as being based on Hilbert spaces, is a probabilistic theory. (Widdows 2004) suggests a link between quantum logic and the lattices he uses to model negation. (van Rijsbergen 2004) has proposed a quantum mechanical theory for modelling relevance in IR. Our proposal is that the interaction of QM and language is a fruitful area of research for AI.

References

Aerts, D., and Gabora, L. 2005. A state-context-property model of concepts and their combinations II: A Hilbert space representation. *Kybernetes* 34(1&2):192–221.

Bos, J. 2005. Towards wide-coverage semantic interpretation. In *Proceedings of the Sixth International Workshop on Computational Semantics (IWCS-6)*, 42–53.

Chomsky, N. 1957. *Syntactic Structures*. The Hague: Mouton.

Clark, S., and Curran, J. R. 2004. Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Meeting of the ACL*, 104–111.

Collins, M., and Duffy, N. 2002. Convolution kernels for natural language. In Dietterich, T. G.; Becker, S.; and Ghahramani, Z., eds., *Advances in Neural Information Processing Systems 14*. MIT Press. 625–632.

Curran, J. R. 2004. *From Distributional to Semantic Similarity*. Ph.D. Dissertation, University of Edinburgh.

Dowty, D. R.; Wall, R. E.; and Peters, S. 1981. *Introduction to Montague Semantics*. Dordrecht: D. Reidel.

Fodor, J., and Lepore, E. 1999. All at sea in semantic space: Churchland on meaning similarity. *Journal of Philosophy* 96(8):381–403.

Grefenstette, G. 1992. Use of syntactic context to produce term association lists for text retrieval. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, 89–97.

Haussler, D. 1999. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, Computer Science Department, University of California in Santa Cruz.

Hughes, R. I. G. 1989. *The Structure and Interpretation of Quantum Mechanics*. Cambridge, MA: Harvard University Press.

Jurafsky, D., and Martin, J. H. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall.

Rosch, E., and Mervis, C. B. 1975. Family resemblances: Studies in the internal structures of categories. *Cognitive Psychology* 7:573–605.

Schütze, H. 1998. Automatic word sense discrimination. *Computational Linguistics* 24(1):97–123.

Smolensky, P., and Legendre, G. 2006. *The Harmonic Mind: from neural computation to optimality-theoretic grammar*. Cambridge, MA: MIT Press.

Sparck Jones, K., and Willett, P., eds. 1997. *Readings in Information Retrieval*. San Franscisco, CA: Morgan Kaufmann.

van Rijsbergen, C. J. 2004. *The Geometry of Information Retrieval*. Cambridge University Press.

Widdows, D. 2004. *Geometry and Meaning*. Stanford University: CSLI Publications.