A Context-Theoretic Framework for Compositionality in Distributional Semantics

Daoud Clarke[*]
University of Hertfordshire

Formalizing "meaning as context" mathematically leads to a new, algebraic theory of meaning, in which composition is bilinear and associative. These properties are shared by other methods that have been proposed in the literature, including the tensor product, vector addition, point-wise multiplication, and matrix multiplication.

Entailment can be represented by a vector lattice ordering, inspired by a strengthened form of the distributional hypothesis, and a degree of entailment is defined in the form of a conditional probability. Approaches to the task of recognizing textual entailment, including the use of subsequence matching, lexical entailment probability, and latent Dirichlet allocation, can be described within our framework.

1. Introduction

This article presents the thesis that defining meaning as context leads naturally to a model in which meanings of strings are represented as elements of an associative algebra over the real numbers, and entailment is described by a vector lattice ordering. This model is general enough to encompass several proposed methods of composition in vector-based representations of meaning.

In recent years, the abundance of text corpora and computing power has allowed the development of techniques to analyze statistical properties of words. For example techniques such as latent semantic analysis (Deerwester et al. 1990) and its variants, and measures of distributional similarity (Lin 1998; Lee 1999), attempt to derive aspects of the meanings of words by statistical analysis, and statistical information is often used when parsing to determine sentence structure (Collins 1997). These techniques have proved useful in many applications within computational linguistics and natural language processing (Grefenstette 1994; Schütze 1998; Bellegarda 2000; Choi, Wiemer-Hastings, and Moore 2001; Lin 2003; McCarthy et al. 2004), arguably providing evidence that they capture something about the nature of words that should be included in representations of their meaning. However, it is very difficult to reconcile these techniques with existing theories of meaning in language, which revolve around logical and ontological representations. The new techniques, almost without exception, can be viewed as dealing with vector-based representations of meaning, placing meaning (at least at the word level) within the realm of mathematics and algebra; conversely the older theories of meaning dwell in the realm of logic and ontology. It seems there is

[*] Gorkana Group, Discovery House, 28–48 Banner Street, London EC1Y8QE.
E-mail: `daoud.clarke@gorkana.com`.

Submission received: 29 October 2008; revised submission received: 28 April 2011; accepted for publication: 29 May 2011.

no unifying theory of meaning to provide guidance to those making use of the new techniques.

The problem appears to be a fundamental one in computational linguistics because the whole foundation of meaning seems to be in question. The older, logical theories often subscribe to a model-theoretic philosophy of meaning (Kamp and Reyle 1993; Blackburn and Bos 2005). According to this approach, sentences should be translated to a logical form that can be interpreted as a description of the state of the world. The new vector-based techniques, on the other hand, are often closer in spirit to the philosophy of *meaning as context*, the idea that the meaning of an expression is determined by how it is used. This is an old idea with origins in the philosophy of Wittgenstein (1953), who said that "meaning just is use," Firth's (1968) "You shall know a word by the company it keeps," and the distributional hypothesis of Harris (1968), that words will occur in similar contexts if and only if they have similar meanings. This hypothesis is justified by the success of techniques such as latent semantic analysis as well as experimental evidence (Miller and Charles 1991). Although the two philosophies are not obviously incompatible—especially because the former applies mainly at the sentence level and the latter mainly at the word level—it is not clear how they relate to each other.

The problem of how to compose vector representations of meanings of words has recently received increased attention (Clark, Coecke, and Sadrzadeh 2008; Mitchell and Lapata 2008; Widdows 2008; Erk and Padó 2009; Baroni and Zamparelli 2010; Guevara 2011; Preller and Sadrzadeh 2011) although the problem has been considered in earlier work (Smolensky 1990; Landauer and Dumais 1997; Foltz, Kintsch, and Landauer 1998; Kintsch 2001). A solution to this problem would have practical as well as philosophical benefits. Current techniques such as latent semantic analysis work well at the word level, but we cannot extend them much beyond this, to the phrase or sentence level, without quickly encountering the data-sparseness problem: There are not enough occurrences of strings of words to determine what their vectors should be merely by looking in corpora. If we knew how such vectors should compose then we would be able to extend the benefits of the vector based techniques to the many applications that require reasoning about the meaning of phrases and sentences.

This article describes the results of our own efforts to identify a theory that can unite these two paradigms, introduced in the author's DPhil thesis (Clarke 2007). In addition, we also discuss the relationship between this theory and methods of composition that have recently been proposed in the literature, showing that many of them can be considered as falling within our framework.

Our approach in identifying the framework is summarized in Figure 1:

- Inspired by the philosophy of meaning as context and vector-based techniques we developed a mathematical model of meaning as context, in which the meaning of a string is a vector representing contexts in which that string occurs in a hypothetical infinite corpus.
- The theory on its own is not useful when applied to real-world corpora because of the problem of data sparseness. Instead we examine the mathematical properties of the model, and abstract them to form a framework which contains many of the properties of the model. Implementations of the framework are called context theories because they can be viewed as theories about the contexts in which strings occur. By analogy with the term "model-theoretic" we use the term "context-theoretic" for concepts relating to context theories, thus we call our framework the context-theoretic framework.

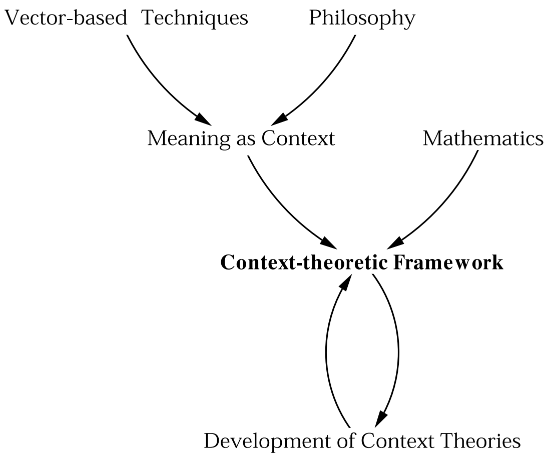

Figure 1
Our approach in developing the context-theoretic framework.

- In order to ensure that the framework was practically useful, context theories were developed in parallel with the framework itself. The aim was to be able to describe existing approaches to representing meaning within the framework as fully as possible.

In developing the framework we were looking for specific properties; namely, we wanted it to:

- provide some guidelines describing in what way the representation of a phrase or sentence should relate to the representations of the individual words as vectors;
- require information about the probability of a string of words to be incorporated into the representation;
- provide a way to measure the degree of entailment between strings based on the particular meaning representation;
- be general enough to encompass logical representations of meaning; and
- be able to incorporate the representation of ambiguity and uncertainty, including statistical information such as the probability of a parse or the probability that a word takes a particular sense.

The framework we present is abstract, and hence does not subscribe to a particular method for obtaining word vectors: They may be raw frequency counts, or vectors obtained by a method such as latent semantic analysis. Nor does the framework provide a recipe for how to represent meaning in natural language; instead it provides restrictions on the set of possibilities. The advantage of the framework is in ensuring that techniques are used in a way that is well-founded in a theory of meaning. For example, given vector representations of words, there is not one single way of combining these to give vector representations of phrases and sentences, but in order to fit within the framework there are certain properties of the representation that need to hold. Any method of combining these vectors in which these properties hold can be considered within the framework and is thus justified according to the underlying theory; in addition the framework instructs us as to how to measure the degree of entailment between strings according to that particular method.

The contribution of this article is as follows:

- We define the context-theoretic framework and introduce the mathematics necessary to understand it. The description presented here is cleaner than that of Clarke (2007), and in addition we provide examples that should provide intuition for the concepts we describe.
- We relate the framework to methods of composition that have been proposed in the literature, namely:
  - vector addition (Landauer and Dumais 1997; Foltz, Kintsch, and Landauer 1998);
  - the tensor product (Smolensky 1990; Clark and Pulman 2007; Widdows 2008);
  - the multiplicative models of Mitchell and Lapata (2008);
  - matrix multiplication (Baroni and Zamparelli 2010; Rudolph and Giesbrecht 2010); and
  - the approach of Clark, Coecke, and Sadrzadeh (2008).

It is important to note that the purpose of describing related work in terms of our framework is not merely to demonstrate the generality of our framework: In doing so, we identify previously ignored features of this work such as the lattice structure within the vector space. This allows any one of these approaches to be endowed with an entailment property defined by this lattice structure, based on a philosophy of meaning as context.

Although the examples described here show that existing approaches can be described within the framework and show some of its potential, they cannot demonstrate its full power. The mathematical structures we make use of are extremely general, and we hope that in the future many interesting discoveries will be made by exploring the realm we identify here.

Our approach in defining the framework may be perceived as overly abstract; however, we believe this approach has many potential benefits, because approaches to composition which may have been considered unrelated (such as the tensor product and vector addition) are now shown to be related. This means that when studying such constructions, work can be avoided by considering the general case, for the same reason that class inheritance aids code reuse. For example, definitions given in terms of the framework can be applied to all instances, such as our definition of a degree of entailment. We also hope to motivate people to prove theorems in terms of the framework, having demonstrated its wide applicability.

The remainder of the article is as follows: In Section 2 we define our framework, introducing the necessary definitions, and showing how related work fits into the framework. In Section 3 we introduce our motivating example, showing that a simple mathematical definition of the notions of "corpus" and "context" leads to an instance of our framework. In Section 4, we describe specific instances of our framework in application to the task of recognizing textual entailment. In Section 5 we show how the sophisticated approach of Clark, Coecke, and Sadrzadeh (2008) can be described within our framework. Finally, in Section 6 we present our conclusions and plans for further work.

2. Context Theory

In this section, we define the fundamental concept of our concern, a context theory, and discuss its properties. The definition is an abstraction of both the more commonly used methods of defining composition in vector-based semantics and our motivating example of meaning as context, described in the next section. Because of its relation to this motivating example, a context theory can be thought of as a hypothesis describing in what contexts all strings occur.

Definition 1 (Context Theory)
A **context theory** is a tuple $\langle A, \mathcal{A}, \xi, V, \psi \rangle$, where A is a set (the alphabet), $\mathcal{A}$ is a unital algebra over the real numbers, ξ is a function from A to $\mathcal{A}$, V is an abstract Lebesgue space, and ψ is an injective linear map from $\mathcal{A}$ to V.

We will explain each part of this definition, introducing the necessary mathematics as we proceed. We assume the reader is familiar with linear algebra; see Halmos (1974) for definitions that are not included here.

2.1 Algebra over a Field

We have identified an algebra over a field (or simply **algebra** when there is no ambiguity) as an important construction because it generalizes nearly all the methods of vector-based composition that have been proposed. An algebra adds a multiplication operation to a vector space; the vector space is intended to describe meaning, and it is this multiplication operation that defines the composition of meaning in context-theoretic semantics.

Definition 2 (Algebra over a Field)
An **algebra over a field** is a vector space $\mathcal{A}$ over a field K together with a binary operation $(a, b) \mapsto ab$ on $\mathcal{A}$ that is bilinear,

$$a(\alpha b + \beta c) = \alpha ab + \beta ac \tag{1}$$

$$(\alpha a + \beta b)c = \alpha ac + \beta bc \tag{2}$$

and associative, $(ab)c = a(bc)$ for all $a, b, c \in \mathcal{A}$ and all $\alpha, \beta \in K$. Some authors do not place the requirement that an algebra is associative, in which case our definition would refer to an **associative algebra**. An algebra is called **unital** if it has a distinguished **unity** element 1 satisfying $1x = x1 = x$ for all $x \in \mathcal{A}$. We are generally only interested in **real** algebras, where K is the field of real numbers, $\mathbb{R}$.

Example 1
The square real-valued matrices of order n form a real unital associative algebra under standard matrix multiplication. The vector operations are defined entry-wise. The unity element of the algebra is the identity matrix.

This means that our proposal is more general than that of Rudolph and Giesbrecht (2010), who suggest using matrix multiplication as a framework for distributional semantic composition. The main differences in our proposal are as follows.

- We allow dimensionality to be infinite, instead of restricting ourselves to finite-dimensional matrices.
- Matrix algebras form a $*$-algebra, whereas we do not currently impose this requirement.
- Many of the vector spaces used in computational linguistics have an implicit lattice structure; we emphasize the importance of this structure and use the associated partial ordering to define entailment.

The purpose of ξ in the context theory is to associate elements of the algebra with strings of words. Considering only the multiplication of $\mathcal{A}$ (and ignoring the vector operations), $\mathcal{A}$ is a monoid, because we assumed that the multiplication on $\mathcal{A}$ is associative. Then ξ induces a monoid homomorphism $a \mapsto \hat{a}$ from A^* to $\mathcal{A}$. We denote the mapped value of $a \in A^*$ by $\hat{a} \in \mathcal{A}$, which is defined as follows:

$$\hat{a} = \xi(a_1)\xi(a_2)\dots\xi(a_n) \tag{3}$$

where $a = a_1a_2\dots a_n$ for $a_i \in A$, and we define $\hat{\epsilon} = 1$, where ϵ is the empty string. Thus, the mapping defined by ^ allows us to associate an element of the algebra with every string of words.

The algebra is what tells us how meanings compose. A crucial part of our thesis is that meanings can be represented by elements of an algebra, and that the type of composition that can be defined using an algebra is general enough to describe the composition of meaning in natural language. To go some way towards justifying this, we give several examples of algebras that describe methods of composition that have been proposed in the literature: namely, point-wise multiplication (Mitchell and Lapata 2008), vector addition (Landauer and Dumais 1997; Foltz, Kintsch, and Landauer 1998), and the tensor product (Smolensky 1990; Clark and Pulman 2007; Widdows 2008).

Example 2 (Point-wise Multiplication)
Consider the n-dimensional real vector space $\mathbb{R}^n$. We describe a vector $u \in \mathbb{R}^n$ in terms of its components as $(u_1, u_2, \dots u_n)$ with each $u_i \in \mathbb{R}$. We can define a multiplication $\cdot$ on this space by

$$(u_1, u_2, \dots, u_n) \cdot (v_1, v_2, \dots, v_n) = (u_1v_1, u_2v_2, \dots u_nv_n) \tag{4}$$

It is easy to see that this satisfies the requirements for an algebra as specified earlier. Table 1 shows a simple example of possible occurrences for three terms in three different contexts, d_1, d_2, and d_3, which may, for example, represent documents. We use this to define the mapping ξ from terms to vectors. Thus, in this example, we have $\xi(cat) = (0, 2, 3)$ and $\xi(big) = (1, 3, 0)$. Under point-wise multiplication, we would have

$$\widehat{big\ cat} = \xi(big) \cdot \xi(cat) = (1, 3, 0) \cdot (0, 2, 3) = (0, 6, 0) \tag{5}$$

Table 1
Example of possible occurrences for three terms in three different contexts.

	d_1	d_2	d_3
cat	0	2	3
animal	2	1	2
big	1	3	0

One commonly used operation for composing vector-based representations of meaning is vector addition. As noted by Rudolph and Giesbrecht (2010), this can be described using matrix multiplication, by embedding an n-dimensional vector u into a matrix of order $n+1$:

$$\begin{pmatrix} \alpha & u_1 & u_2 & \cdots & u_n \\ 0 & \alpha & 0 & \cdots & 0 \\ 0 & 0 & \alpha & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \alpha \end{pmatrix} \tag{6}$$

where $\alpha = 1$. The set of all such matrices, for all real values of α, forms a **subalgebra** of the algebra of matrices of order $n+1$. A subalgebra of an algebra $\mathcal{A}$ is a sub-vector space of $\mathcal{A}$ which is closed under the multiplication of $\mathcal{A}$. This subalgebra can be equivalently described as follows:

Example 3 (Additive Algebra)
For two vectors $u = (\alpha, u_1, u_2, \ldots u_n)$ and $v = (\beta, v_1, v_2 \ldots v_n)$ in $\mathbb{R}^{n+1}$, we define the additive product $\boxplus$ by

$$u \boxplus v = (\alpha\beta,\ \alpha v_1 + \beta u_1,\ \alpha v_2 + \beta u_2,\ \ldots \alpha v_n + \beta u_n) \tag{7}$$

To verify that this multiplication makes $\mathbb{R}^{n+1}$ an algebra, we can directly verify the bilinear and associativity requirements, or check that it is isomorphic to the subalgebra of matrices discussed previously.

Using Table 1, we define ξ_+ so that it maps n-dimensional context vectors to $\mathbb{R}^{n+1}$, where the first component is 1, so $\xi_+(\mathit{big}) = (1,1,3,0)$ and $\xi_+(\mathit{cat}) = (1,0,2,3)$ and

$$\widehat{\mathit{big\ cat}} = \xi_+(\mathit{big}) \boxplus \xi_+(\mathit{cat}) = (1,1,5,3) \tag{8}$$

Point-wise multiplication and addition are not ideal as methods for composing meaning in natural language because they are commutative; although it is often useful to consider the simpler, commutative case, natural language itself is inherently non-commutative. One obvious method of composing vectors that is not commutative is the **tensor product**. This method of composition can be viewed as a product in an algebra by considering the **tensor algebra**, which is formed from direct sums of all tensor powers of a base vector space.

We assume the reader is familiar with the tensor product and direct sum (see Halmos [1974] for definitions); we recall their basic properties here. Let V_n denote a vector space of dimensionality n (note that all vector spaces of a fixed dimensionality are isomorphic). Then the tensor product space $V_n \otimes V_m$ is isomorphic to a space V_{nm} of dimensionality nm; moreover, given orthonormal bases $B = \{b_1, b_2, \ldots, b_n\}$ for V_n and $C = \{c_1, c_2, \ldots, c_m\}$ for V_m there is an orthonormal basis for V_{nm} defined by

$$\{b_i \otimes c_j : 1 \leq i \leq n \text{ and } 1 \leq j \leq m\} \tag{9}$$

Example 4
The multiplicative models of Mitchell and Lapata (2008) correspond to the class of finite dimensional algebras. Let $\mathcal{A}$ be a finite-dimensional vector space. Then every associative bilinear product on $\mathcal{A}$ can be described by a linear function T from $\mathcal{A} \otimes \mathcal{A}$ to $\mathcal{A}$, as required in Mitchell and Lapata's model. To see this, consider the action of the product $\cdot$ on two orthonormal basis vectors a and b of $\mathcal{A}$. This is a vector in $\mathcal{A}$, thus we can define $T(a \otimes b) = a \cdot b$. By considering all basis vectors, we can define the linear function T.

If the tensor product can loosely be viewed as "multiplying" vector spaces, then the direct sum is like adding them; the space $V_n \oplus V_m$ has dimensionality $n + m$ and has basis vectors

$$\{b_i \oplus 0 : 1 \leq i \leq n\} \cup \{0 \oplus c_j : 1 \leq j \leq m\}; \tag{10}$$

it is usual to write $b \oplus 0$ as b and $0 \oplus c$ as c.

Example 5 (Tensor Algebra)
If V is a vector space, then we define $T(V)$, the **free algebra** of **tensor algebra** generated by V, as:

$$T(V) = \mathbb{R} \oplus V \oplus (V \otimes V) \oplus (V \otimes V \otimes V) \oplus \cdots \tag{11}$$

where we assume that the direct sum is commutative. We can think of it as the direct sum of all tensor powers of V, with $\mathbb{R}$ representing the zeroth power. In order to make this space an algebra, we define the product on elements of these tensor powers, viewed as subspaces of the tensor algebra, as their tensor product. This is enough to define the product on the whole space, because every element can be written as a sum of tensor powers of elements of V. There is a natural embedding from V to $T(V)$, where each element maps to an element in the first tensor power. Thus for example we can think of u, $u \otimes v$, and $u \otimes v + w$ as elements of $T(V)$, for all $u, v, w \in V$.

This product defines an algebra because the tensor product is a bilinear operation. Taking $V = \mathbb{R}^3$ and using ξ as the natural embedding from the context vector of a string $T(V)$, our previous example becomes

$$\begin{align}
\widehat{big\ cat} &= \xi(big) \otimes \xi(cat) \tag{12}\\
&= (1,3,0) \otimes (0,2,3) \tag{13}\\
&\cong (1(0,2,3), 3(0,2,3), 0(0,2,3)) \tag{14}\\
&\cong (0,2,3,0,6,9,0,0,0) \tag{15}
\end{align}$$

where the last two lines demonstrate how a vector in $\mathbb{R}^3 \otimes \mathbb{R}^3$ can be described in the isomorphic space $\mathbb{R}^9$.

2.2 Vector Lattices

The next part of the definition specifies an abstract Lebesgue space. This is a special kind of vector lattice, or even more generally, a partially ordered vector space. This lattice structure is implicit in most vector spaces used in computational linguistics, and an important part of our thesis is that the partial ordering can be interpreted as an entailment relation.

Definition 3 (Partially Ordered Vector Space)
A partially ordered vector space V is a real vector space together with a partial ordering $\leq$ such that:

if $x \leq y$ then $x + z \leq y + z$
if $x \leq y$ then $\alpha x \leq \alpha y$

for all $x, y, z \in V$, and for all $\alpha \geq 0$. Such a partial ordering is called a **vector space order** on V. An element u of V satisfying $u \geq 0$ is called a **positive element**; the set of all positive elements of V is denoted V^+. If $\leq$ defines a lattice on V then the space is called a **vector lattice** or **Riesz space**.

Example 6 (Lattice Operations on $\mathbb{R}^n$)
A vector lattice captures many properties that are inherent in real vector spaces when there is a *distinguished basis*. In $\mathbb{R}^n$, given a specific basis, we can write two vectors u and v as sequences of numbers: $u = (u_1, u_2, \ldots u_n)$ and $v = (v_1, v_2, \ldots v_n)$. This allows us to define the lattice operations of meet $\wedge$ and join $\vee$ as

$$u \wedge v = (\min(u_1, v_1), \min(u_2, v_2), \ldots \min(u_n, v_n)) \tag{16}$$

$$u \vee v = (\max(u_1, v_1), \max(u_2, v_2), \ldots \max(u_n, v_n)) \tag{17}$$

These are the component-wise minimum and maximum, respectively. The partial ordering is then given by $u \leq v$ if and only if $u \wedge v = u$, or equivalently $u_n \leq v_n$ for all n. A graphical depiction of the meet operation is shown in Figure 2.

The vector operations of addition and multiplication by scalar, which can be defined in a similar component-wise fashion, are nevertheless independent of the particular

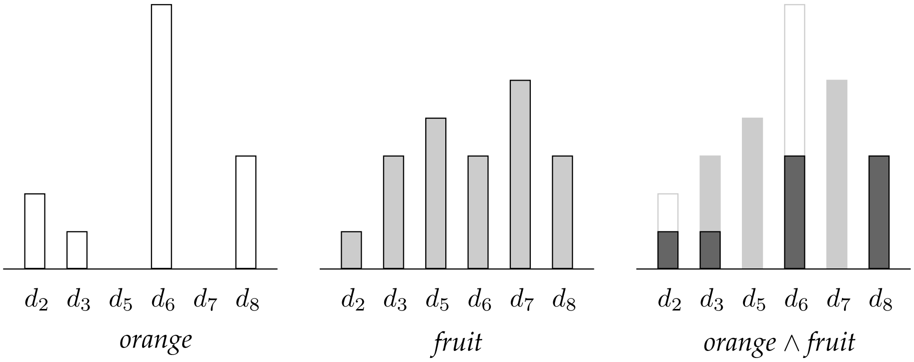

Figure 2
Vector representations of the terms *orange* and *fruit* based on hypothetical occurrences in six documents and their vector lattice meet (the darker shaded area).

basis chosen. Conversely, the lattice operations depend on the choice of basis, so the operations as defined herein would behave differently if the components were written using a different basis. We argue that it makes sense for us to consider these properties of vectors in the context of computational linguistics because we can often have a distinguished basis: namely, the one defined by the contexts in which terms occur. Of course it is true that techniques such as latent semantic analysis introduce a new basis which does not have a clear interpretation in relation to contexts; nevertheless they nearly always identify a distinguished basis which we can use to define the lattice operations. Because our aim is a theory of meaning as context, we should include in our theory a description of the lattice structure which arises out of consideration of these contexts.

We argue that the mere association of words with vectors is not enough to constitute a theory of meaning—a theory of meaning must allow us to interpret these vectors. In particular it should be able to tell us whether one meaning entails or implies another; indeed this is one meaning of the verb *to mean*. Entailment is an asymmetric relation: "x entails y" does not have the same meaning as "y entails x". Vector representations allow the measurement of similarity or distance, through an inner product or metric; this is a symmetric relation, however, and so cannot be suitable for describing entailment.

In propositional and first order logic, the entailment relation is a partial ordering; in fact it is a Boolean algebra, which is a special kind of lattice. It seems natural to consider whether the lattice structure that is inherent in the vector representations used in computational linguistics can be used to model entailment.

We believe our framework is suited to all vector-based representations of natural language meaning, however the vectors are obtained. Given this assumption, we can only justify our assumption that the partial order structure of the vector space is suitable to represent the entailment relation by observing that it has the right kind of properties we would expect from this relation.

There may be more justification for this assumption, however, based on the case where the vectors for terms are simply their frequencies of occurrences in n different contexts, so that they are vectors in $\mathbb{R}^n$. In this case, the relation $\xi(x) \leq \xi(y)$ means that y occurs at least as frequently as x in every context. This means that y occurs in at least as wide a range of contexts as x, and occurs as least as frequently as x. Thus the statement "x entails y if and only if $\xi(x) \leq \xi(y)$" can be viewed as a stronger form of the distributional hypothesis of Harris (1968).

In fact, this idea can be related to the notion of **distributional generality**, introduced by Weeds, Weir, and McCarthy (2004) and developed by Geffet and Dagan (2005). A term x is distributionally more general than another term y if x occurs in a subset of the contexts that y occurs in. The idea is that distributional generality may be connected to **semantic generality**. An example of this is the **hypernymy** or **is-a** relation that is used to express generality of concepts in ontologies; for example, the term *animal* is a hypernym of *dog* because a dog is an animal. Weeds, Weir, and McCarthy (2004, p. 1019) explain the connection to distributional generality as follows:

> Although one can obviously think of counter-examples, we would generally expect that the more specific term *dog* can only be used in contexts where *animal* can be used and that the more general term *animal* might be used in all of the contexts where *dog* is used and possibly others. Thus, we might expect that distributional generality is correlated with semantic generality...

Our proposal, in the case where words are represented by frequency vectors, can be considered a stronger version of distributional generality, where the additional requirement is on the frequency of occurrences. In practice, this assumption is unlikely to be compatible with the ontological view of entailment. For example the term *entity* is semantically more general than the term *animal*; however, *entity* is unlikely to occur more frequently in each context, because it is a rarer word. A more realistic foundation for this assumption might be if we were to consider the components for a word to represent the plausibility of observing the word in each context. The question then, of course, is how such vectors might be obtained. Another possibility is to attempt to weight components in such a way that entailment becomes a plausible interpretation for the partial ordering relation.

Even if we allow for such alternatives, however, in general it is unlikely that the relation will hold between any two strings, because $u \leq v$ iff $u_i \leq v_i$ for each component, u_i, v_i, of the two vectors. Instead, we propose to allow for *degrees of entailment*. We take a Bayesian perspective on this, and suggest that the degree of entailment should take the form of a conditional probability. In order to define this, however, we need some additional structure on the vector lattice that allows it to be viewed as a description of probability, by requiring it to be an abstract Lebesgue space.

Definition 4 (Banach Lattice)
A **Banach lattice** V is a vector lattice together with a norm $\|\cdot\|$ such that V is complete with respect to $\|\cdot\|$.

Definition 5 (Abstract Lebesgue Space)
An **abstract Lebesgue** (or AL) space is a Banach lattice V such that

$$\|u + v\| = \|u\| + \|v\| \tag{18}$$

for all u, v in V with $u \geq 0, v \geq 0$ and $u \wedge v = 0$.

Example 7 (ℓ^p Spaces)
Let $u = (u_1, u_2, \ldots)$ be an infinite sequence of real numbers. We can view u_i as components of the infinite-dimensional vector u. We call the set of all such vectors the **sequence space**; it is a vector space where the operations are defined component-wise. We define a set of norms, the ℓ^p-norms, on the space of all such vectors by

$$\|u\|_p = \left(\sum_{i>0} |u_i|^p\right)^{1/p} \tag{19}$$

The space of all vectors u for which $\|u\|_p$ is finite is called the ℓ^p space. Considered as vector spaces, these are Banach spaces, because they are complete with respect to the associated norm, and under the component-wise lattice operations, they are Banach lattices. In particular, the ℓ^1 space is an abstract Lebesgue space under the ℓ^1 norm.

The finite-dimensional real vector spaces $\mathbb{R}^n$ can be considered as special cases of the sequence spaces (consisting of vectors in which all but n components are zero) and, because they are finite-dimensional, we can use any of the ℓ^p norms. Thus, our previous examples, in which ξ mapped terms to vectors in $\mathbb{R}^n$, can be considered as mapping to abstract Lebesgue spaces if we adopt the ℓ^1 norm.

2.3 Degrees of Entailment

We propose that in vector-based semantics, a degree of entailment is more appropriate than a black-and-white observation of whether or not entailment holds. If we think of the vectors as describing "degrees of meaning," it makes sense that we should then look for degrees of entailment.

Conditional probability is closely connected to entailment: If A entails B, then $P(B|A) = 1$. Moreover, if A and B are mutually exclusive, then $P(A|B) = P(B|A) = 0$. It is thus natural to think of conditional probability as a degree of entailment.

An abstract Lebesgue space has many of the properties of a probability space, where the set operations of a probability space are replaced by the lattice operations of the vector space. This means that we can think of an abstract Lebesgue space as a vector-based probability space. Here, events correspond to positive elements with the norm less than or equal to 1; the probability of an event u is given by the norm (which we shall always assume is the ℓ^1 norm), and the joint probability of two events u and v is $\|u \wedge v\|_1$.

Definition 6 (Degree of Entailment)
We define the degree to which u entails v in the form of a conditional probability:

$$\mathrm{Ent}(u, v) = \frac{\|u \wedge v\|_1}{\|u\|_1} \tag{20}$$

If we are only interested in degrees of entailment (i.e., conditional probabilities) and not probabilities, then we can drop the requirement that the norm should be less than or equal to one, because conditional probabilities are automatically normalized. This definition, together with the multiplication of the algebra, allows us to compute the degree of entailment between any two strings according to the context theory.

Example 8
The vectors given in Table 1 give the following calculation for the degree to which *cat* entails *animal*:

$$\xi(\mathit{cat}) = (0, 2, 3) \tag{21}$$

$$\xi(\mathit{animal}) = (2, 1, 2) \tag{22}$$

$$\xi(\mathit{cat}) \wedge \xi(\mathit{animal}) = (0, 1, 2) \tag{23}$$

$$\mathrm{Ent}(\xi(\mathit{cat}), \xi(\mathit{animal})) = \|\xi(\mathit{cat}) \wedge \xi(\mathit{animal})\|_1 / \|\xi(\mathit{cat})\|_1 = 3/5 \tag{24}$$

An important question is how this context-theoretic definition of the degree of entailment relates to more familiar notions of entailment. There are three main ways in which the term entailment is used:

- the **model-theoretic** sense of entailment in which a theory A entails a theory B if every model of A is also a model of B. It was shown in Clarke (2007) that this type of entailment can be described using context theories, where sentences are represented as projections on a vector space.
- entailment between terms (as expressed for example in the WordNet hierarchy), for example the hypernymy relation between the terms *cat* and *animal* encodes the fact that a cat is an animal. In Clarke (2007) we showed

that such relations can be encoded in the partial order structure of a vector lattice.

- Human common-sense judgments as to whether one sentence entails or implies another sentence, as used in the Recognising Textual Entailment Challenges (Dagan, Glickman, and Magnini 2005).

Our context-theoretic notion of entailment is thus intended to generalize both the first two senses of entailment given here. In addition, we hope that context theories will be useful in the practical application of recognizing textual entailment. Capturing this type of entailment is not our initial aim because we are interested in foundational issues, and doing well at this task poses major engineering challenges beyond the scope of our work. Nevertheless, we believe the ability to represent the preceding two types of entailment as well as standard distributional methods of composition bodes well for the possibility of using our framework for this task. In Section 4 we describe several basic approaches to textual entailment within the framework.

Our definition is more general than the model-theoretic and hypernymy notions of entailment, however, as it allows the measurement of a degree of entailment between any two strings: As an extreme example, one may measure the degree to which *not a* entails *in the*. Although this may not be useful or philosophically meaningful, we view it as a practical consequence of the fact that every string has a vector representation in our model, which coincides with the current practice in vector-based compositionality techniques (Clark, Coecke, and Sadrzadeh 2008; Widdows 2008).

2.4 Lattice Ordered Algebras

A **lattice ordered algebra** merges the lattice ordering of the vector space V with the product of $\mathcal{A}$. This structure encapsulates the ordering properties that are familiar from multiplication in matrices and elementary arithmetic. For this reason, many proposed methods of composing vector-based representations of meaning can be viewed as lattice ordered algebras. The only reason we have not included it as a requirement of the framework is because our motivating example (described in the next section) is not guaranteed to have this property, although it does give us a partially ordered algebra.

Definition 7 (Partially Ordered Algebra)
A **partially ordered algebra** $\mathcal{A}$ is an algebra which is also a partially ordered vector space, which satisfies $u \cdot v \geq 0$ for all $u, v \in \mathcal{A}^+$. If the partial ordering is a lattice, then $\mathcal{A}$ is called a **lattice-ordered algebra**.

Example 9 (Lattice-Ordered Algebra of Matrices)
The matrices of order n form a lattice-ordered algebra under normal matrix multiplication, where the lattice operations are defined as the entry-wise minimum and maximum.

Example 10 (Operators on ℓ^p Spaces)
Matrices can be viewed as operators on finite-dimensional vector spaces; in fact this lattice property extends to operators on certain infinite-dimensional spaces, the ℓ^p spaces,

by the Riesz-Kantorovich theorem (Abramovich and Aliprantis 2002). The operations are defined by:

$$(S \vee T)(u) = \sup\{S(v) + T(w) : v, w \in U^+ \text{ and } v + w = u\} \tag{25}$$

$$(S \wedge T)(u) = \inf\{S(v) + T(w) : v, w \in U^+ \text{ and } v + w = u\} \tag{26}$$

If $\mathcal{A}$ is a lattice-ordered algebra which is also an abstract Lebesgue space, then $\langle A, \mathcal{A}, \xi, \mathcal{A}, 1\rangle$ is a context theory. In this simplified situation, $\mathcal{A}$ plays the role of the vector lattice as well as the algebra; ξ maps from A to $\mathcal{A}$ as before, and 1 indicates the identity map on $\mathcal{A}$. Many of the examples we discuss will be of this form, so we will use the shorthand notation, $\langle A, \mathcal{A}, \xi\rangle$. It is tempting to adopt this as the definition of context theory; as we will see in the next section, however, this is not supported by our prototypical example of a context theory as in this case the algebra is not necessarily lattice-ordered.

3. Context Algebras

In this section we describe the prototypical examples of a context theory, the **context algebras**. The definition of a context algebra originates in the idea that the notion of "meaning as context" can be extended beyond the word level to strings of arbitrary length. In fact, the notion of context algebra can be thought of as a generalization of the **syntactic monoid** of a formal language: Instead of a set of strings defining the language, we have a *fuzzy set* of strings, or more generally, a real-valued function on a free monoid.

We call such functions **real-valued languages** and they take the place of formal languages in our theory. We attach a real number to each string which is intended as an indication of its importance or likelihood of being observed; for example, those with a value of zero are considered not to occur.

Definition 8 (Real-Valued Language)
Let A be a finite set of symbols. A real-valued language (or simply a **language** when there is no ambiguity) L on A is a function from A^* to $\mathbb{R}$. If the range of L is a subset of $\mathbb{R}^+$ then L is called a **positive language**. If the range of L is a subset of $[0, 1]$ then L is called a **fuzzy language**. If L is a positive language such that $\sum_{x \in A^*} L(x) = 1$ then L is a probability distribution over A^*, a **distributional language**.

One possible interpretation for L when it is a distributional language is that $L(x)$ is the probability of observing the string x when selecting a document at random from an infinite collection of documents.

The following inclusion relation applies among these classes of language:

$$\text{distributional} \implies \text{fuzzy} \implies \text{positive} \implies \text{real-valued} \tag{27}$$

Because A^* is a countable set, the set $\mathbb{R}^{A^*}$ of functions from A^* to $\mathbb{R}$ is isomorphic to the sequence space, and we shall treat them equivalently. We denote by $\ell^p(A^*)$ the set of functions with a finite ℓ^p norm when considered as sequences. There is another hierarchy of spaces given by the inclusion of the ℓ^p spaces: $\ell^p(A^*) \subseteq \ell^q(A^*)$ if $p \leq q$. In particular,

$$\ell^1(A^*) \subseteq \ell^2(A^*) \subseteq \ell^\infty(A^*) \subseteq \mathbb{R}^{A^*} \tag{28}$$

where the ℓ^∞ norm gives the maximum value of the function and $\ell^\infty(A^*)$ is the space of all *bounded* real-valued functions on A^*. Recall that a linear operator T from one vector space U to another V is called bounded if there exists some $\alpha > 0$ such that $\|Tu\| \leq \alpha\|u\|$ for all $u \in U$, where the norm on the left hand side is the norm in V, and that on the right hand side is in U.

Note that probability distributions are in $\ell^1(A^*)$ and fuzzy languages are in $\ell^\infty(A^*)$. If $L \in \ell^1(A^*)^+$ (the space of positive functions on A^* such that the sum of all values of the function is finite) then we can define a probability distribution p_L over A^* by $p_L(x) = L(x)/\|L\|_1$. Similarly, if $L \in \ell^\infty(A^*)^+$ (the space of bounded positive functions on A^*) then we can define a fuzzy language f_L by $f_L(x) = L(x)/\|L\|_\infty$.

Example 11
Given a finite set of strings $C \subset A^*$, which we may imagine to be a corpus of documents, define $L(x) = 1/|C|$ if $x \in C$, or 0 otherwise. Then L is a probability distribution over A^*.

In general, we think of a real-valued language as an abstraction of a corpus; in particular, we think of a corpus as a finite sample of a distributional language representing all possible documents that could ever be written.

Example 12
Let L be a language such that $L(x) = 0$ for all but a finite subset of A^*. Then $L \in \ell^p(A^*)$ for all p.

Example 13
Let L be the language defined by $L(x) = |x|$ where x is the length of (i.e., number of symbols in) string x. Then L is a positive language which is not bounded: For any string y there exists a z such that $L(z) > L(y)$, for example $z = ay$ for $a \in A$.

Example 14
Let L be the language defined by $L(x) = 1/2$ for all x. Then L is a fuzzy language but $L \notin \ell^1(A^*)$

We will assume now that L is fixed, and consider the properties of contexts of strings with respect to this language. As in a syntactic monoid, we consider the context to be everything surrounding the string, although in this case instead of a set of pairs of strings we have a function from pairs of strings to the real numbers. We emphasize the vector nature of these real-valued functions by calling them "context vectors." Our thesis is centered around these vectors, and it is their properties that form the inspiration for the context-theoretic framework.

Definition 9 (Context Vectors)
Let L be a language on A. For $x \in A^*$, we define the context of x as a vector $\hat{x} \in \mathbb{R}^{A^* \times A^*}$:

$$\hat{x}(y, z) = L(yxz) \tag{29}$$

In other words, $\hat{x}$ is a function from pairs of strings to the real numbers, and the value of $\hat{x}(y, z)$ is the value of x in the context (y, z), which is $L(yxz)$.

The question we are addressing is: Does there exist some algebra $\mathcal{A}$ containing the context vectors of strings in A^* such that $\hat{x} \cdot \hat{y} = \widehat{xy}$ where $x, y \in A^*$ and $\cdot$ indicates multiplication in the algebra? As a first try, consider the vector space $L^\infty(A^* \times A^*)$ in which the context vectors live. Is it possible to define multiplication on the whole vector space such that the condition just specified holds?

Example 15
Consider the language C on the alphabet $A = \{a, b, c, d, e, f\}$ defined by $C(abcd) = C(aecd) = C(abfd) = \frac{1}{3}$ and $C(x) = 0$ for all other $x \in A^*$. Now if we take the shorthand notation of writing the basis vector in $L^\infty(A^* \times A^*)$ corresponding to a pair of strings as the pair of strings itself then

$$\hat{b} = \tfrac{1}{3}(a, cd) + \tfrac{1}{3}(a, fd) \tag{30}$$

$$\hat{c} = \tfrac{1}{3}(ab, d) + \tfrac{1}{3}(ae, d) \tag{31}$$

$$\widehat{bc} = \tfrac{1}{3}(a, d) \tag{32}$$

It would thus seem sensible to define multiplication of contexts so that $\frac{1}{3}(a, cd) \cdot \frac{1}{3}(ab, d) = \frac{1}{3}(a, d)$. However we then find

$$\hat{e} \cdot \hat{f} = \tfrac{1}{3}(a, cd) \cdot \tfrac{1}{3}(ab, d) \neq \widehat{ef} = 0 \tag{33}$$

showing that this definition of multiplication doesn't provide us with what we are looking for. In fact, if there did exist a way to define multiplication on contexts in a satisfactory manner it would necessarily be far from intuitive, as, in this example, we would have to define $(a, cd) \cdot (ab, d) = 0$ meaning the product $\hat{b} \cdot \hat{c}$ would have to have a non-zero component derived from the products of context vectors (a, fd) and (ae, d) which don't relate at all to the contexts of bc. This leads us to instead define multiplication on a subspace of $L^\infty(A^* \times A^*)$.

Definition 10 (Generated Subspace $\mathcal{A}$)
The subspace $\mathcal{A}$ of $L^\infty(A^* \times A^*)$ is the set defined by

$$\mathcal{A} = \{a : a = \sum_{x \in A^*} \alpha_x \hat{x} \text{ for some } \alpha_x \in \mathbb{R}\} \tag{34}$$

In other words, it is the space of all vectors formed from linear combinations of context vectors.

Because of the way we define the subspace, there will always exist some basis $\mathcal{B} = \{\hat{u} : u \in B\}$ where $B \subseteq A^*$, and we can define multiplication on this basis by $\hat{u} \cdot \hat{v} = \widehat{uv}$ where $u, v \in B$. Defining multiplication on this basis defines it for the whole vector subspace, because we define multiplication to be linear, making $\mathcal{A}$ an algebra.

There are potentially many different bases we could choose, however, each corresponding to a different subset of A^*, and each giving rise to a different definition of multiplication. Remarkably, this isn't a problem.

Proposition 1 (Context Algebra)
Multiplication on $\mathcal{A}$ is the same irrespective of the choice of basis B.

Proof
We say $B \subseteq A^*$ defines a basis $\mathcal{B}$ for $\mathcal{A}$ when $\mathcal{B}$ is a basis such that $\mathcal{B} = \{\hat{x} : x \in B\}$. Assume there are two sets $B_1, B_2 \subseteq A^*$ that define corresponding bases $\mathcal{B}_1$ and $\mathcal{B}_2$ for $\mathcal{A}$. We will show that multiplication in basis $\mathcal{B}_1$ is the same as in the basis $\mathcal{B}_2$.

We represent two basis elements $\hat{u}_1$ and $\hat{u}_2$ of $\mathcal{B}_1$ in terms of basis elements of $\mathcal{B}_2$:

$$\hat{u}_1 = \sum_i \alpha_i \hat{v}_i \quad \text{and} \quad \hat{u}_2 = \sum_j \beta_j \hat{v}_j \tag{35}$$

for some $u_i \in B_1$, $v_j \in B_2$ and $\alpha_i, \beta_j \in \mathbb{R}$. First consider multiplication in the basis $\mathcal{B}_1$. Note that $\hat{u}_1 = \sum_i \alpha_i \hat{v}_i$ means that $L(xu_1y) = \sum_i \alpha_i L(xv_iy)$ for all $x, y \in A^*$. This includes the special case where $y = u_2y'$ so

$$L(xu_1u_2y') = \sum_i \alpha_i L(xv_iu_2y') \tag{36}$$

for all $x, y' \in A^*$. Similarly, we have $L(xu_2y) = \sum_j \beta_j L(xv_jy)$ for all $x, y \in A^*$ which includes the special case $x = x'v_i$, so $L(x'v_iu_2y) = \sum_j \beta_j L(x'v_iv_jy)$ for all $x', y \in A^*$. Inserting this into Equation (36) yields

$$L(xu_1u_2y) = \sum_{i,j} \alpha_i \beta_j L(xv_iv_jy) \tag{37}$$

for all $x, y \in A^*$ which we can rewrite as

$$\hat{u}_1 \cdot \hat{u}_2 = \widehat{u_1u_2} = \sum_{i,j} \alpha_i \beta_j (\hat{v}_i \cdot \hat{v}_j) = \sum_{i,j} \alpha_i \beta_j \widehat{v_iv_j} \tag{38}$$

Conversely, the product of u_1 and u_2 using the basis $\mathcal{B}_2$ is

$$\hat{u}_1 \cdot \hat{u}_2 = \sum_i \alpha_i \hat{v}_i \cdot \sum_j \beta_j \hat{v}_j = \sum_{i,j} \alpha_i \beta_j (\hat{v}_i \cdot \hat{v}_j) \tag{39}$$

thus showing that multiplication is defined independently of what we choose as the basis. ■

Example 16
Returning to the previous example, we can see that in this case multiplication is in fact defined on $L^\infty(A^* \times A^*)$ because we can describe each basis vector in terms of context vectors:

$$(a, fd) \cdot (ae, d) = 3(\hat{b} - \hat{e}) \cdot 3(\hat{c} - \hat{f}) = -3(a, d) \tag{40}$$

$$(a, cd) \cdot (ae, d) = 3\hat{e} \cdot 3(\hat{c} - \hat{f}) = 3(a, d) \tag{41}$$

$$(a, fd) \cdot (ab, d) = 3(\hat{b} - \hat{e}) \cdot 3\hat{f} = 3(a, d) \tag{42}$$

$$(a, cd) \cdot (ab, d) = 3\hat{e} \cdot 3\hat{f} = 0, \tag{43}$$

thus confirming what we predicted about the product of $\hat{b}$ and $\hat{c}$: The value is only correct because of the negative correction from $(a, fd) \cdot (ae, d)$. This example also serves to demonstrate an important property of context algebras: They do not satisfy the positivity condition; it is possible for positive vectors (those with all components greater than or equal to zero) to have a non-positive product. This means they are not necessarily partially ordered algebras under the normal partial order. Compare this to the case of matrix multiplication, for example, where the product of two positive matrices is always positive.

The notion of a context theory is founded on the prototypical example given by context vectors. So far we have shown that multiplication can be defined on the vector space $\mathcal{A}$ generated by context vectors of strings; we have not discussed the lattice properties of the vector space, however. In fact, $\mathcal{A}$ does not come with a natural lattice ordering that makes sense for our purposes, although the original space $\mathbb{R}^{A^* \times A^*}$ does—it is isomorphic to the sequence space. Thus $\langle A, \mathcal{A}, \xi, \mathbb{R}^{A^* \times A^*}, \psi \rangle$ will form our context theory, where $\xi(a) = \hat{a}$ for $a \in A$ and ψ is the **canonical map** that simply maps elements of $\mathcal{A}$ to themselves, but are considered as elements of $\mathbb{R}^{A^* \times A^*}$. There is an important caveat here, however: We required that the vector lattice be an abstract Lebesgue space, which means we need to be able to define a norm on it. The ℓ^1 norm on $\mathbb{R}^{A^* \times A^*}$ is an obvious candidate, although it is not guaranteed to be finite. This is where the nature of the underlying language L becomes important.

We might hope that the most restrictive class of the languages we discussed, the distributional languages, would guarantee that the norm is finite. Unfortunately, this is not the case, as the following example demonstrates.

Example 17
Let L be the language defined by

$$L(a^{2^n}) = 1/2^{n+1} \tag{44}$$

for integer $n \geq 0$, and zero otherwise, where by a^n we mean n repetitions of a, so for example, $L(a) = \frac{1}{2}$, $L(aa) = \frac{1}{4}$, $L(aaa) = 0$, and $L(aaaa) = \frac{1}{8}$. Then L is a probability distribution over A^*, because L is positive and $\|L\|_1 = 1$. However, $\|\hat{a}\|_1$ is infinite, because each string x for which $L(x) > 0$ contributes $1/2$ to the value of the norm, and there are an infinite number of such strings.

The problem in the previous example is that the average string length is infinite. If we restrict ourselves to distributional languages in which the average string length is finite, then the problem goes away.

Proposition 2
Let L be a probability distribution over A^* such that

$$\bar{L} = \sum_{x \in A^*} L(x)|x| \tag{45}$$

is finite, where $|x|$ is the number of symbols in string x; we will call such languages **finite average length**. Then $\|\hat{y}\|_1$ is finite for each $y \in A^*$.

Proof
Denote the number of occurrences of string y as a substring of string x by $|x|_y$. Clearly $|x|_y \leq |x|$ for all $x, y \in A^*$. Moreover,

$$\|\hat{y}\|_1 = \sum_{x \in A^*} L(x)|x|_y \leq \sum_{x \in A^*} L(x)|x| \tag{46}$$

and so $\|\hat{y}\|_1 \leq \bar{L}$ is finite for all $y \in A^*$. ■

If L is finite average length, then $\mathcal{A} \subseteq \ell^1(A^* \times A^*)$, and so $\langle A, \mathcal{A}, \xi, \ell^1(A^* \times A^*), \psi \rangle$ is a context theory, where ψ is the canonical map from $\mathcal{A}$ to $\ell^1(A^* \times A^*)$. Thus context algebras of finite average length languages provide our prototypical examples of context theories.

3.1 Discussion

The benefit of the context-theoretic framework is in providing a space of exploration for models of meaning in language. Our effort has been in finding principles by which to define the boundaries of this space. Each of the key boundaries, namely, bilinearity and associativity of multiplication and entailment through vector lattice structure, can also be viewed as limitations of the model.

Bilinearity is a strong requirement to place, and has wide-ranging implications for the way meaning is represented in the model. It can be interpreted loosely as follows: Components of meaning persist or diminish but do not spontaneously appear. This is particularly counterintuitive in the case of idiom and metaphor in language. It means that, for example, both *red* and *herring* must contain some components relating to the meaning of *red herring* which only come into play when these two words are combined in this particular order. Any other combination would give a zero product for these components. It is easy to see how this requirement arises from a context-theoretic perspective, nevertheless from a linguistic perspective it is arguably undesirable.

One potential limitation of the model is that it does not explicitly model syntax, but rather syntactic restrictions are encoded into the vector space and product itself. For example, we may assume the word *square* has some component of meaning in common with the word *shape*. Then we would expect this component to be preserved in the sentences *He drew a square* and *He drew a shape*. However, in the case of the two sentences *The box is square* and **The box is shape* we would expect the second to be represented by the zero vector because it is not grammatical; *square* can be a noun and an adjective, whereas *shape* cannot. Distributivity of meaning means that the component of meaning that *square* has in common with *shape* must be disjoint with the adjectival component of the meaning of *square*.

Associativity is also a very strong requirement to place; indeed Lambek (1961) introduced non-associativity into his calculus precisely to deal with examples that were not satisfactorily dealt with by his associative model (Lambek 1958).

Our framework provides answers to someone considering the use of algebra for natural language semantics. What field should be used? The real numbers. Need the algebra be finite-dimensional? No. Should the algebra by unital? Yes. Some of these answers impose restrictions on what is possible within the framework. The full implication of these restrictions for linguistics is beyond the scope of this article, and indeed is not yet known.

Although we hope that these features or boundaries are useful in their current form, it may be that with time, or for certain applications, there is a reason to expand or contract certain of them, perhaps because of theoretical discoveries relating to the model of meaning as context, or for practical or linguistic reasons, if, for example, the model is found to be too restrictive to model certain linguistic phenomena.

4. Applications to Textual Entailment

In this section we analyze approaches to the problem of recognizing textual entailment, showing how they can be related to the context-theoretic framework, and discussing potential new approaches that are suggested by looking at them within the framework. We first discuss some simple approaches to textual entailment based on subsequence matching and measuring lexical overlap. We then look at the approach of Glickman and Dagan (2005), showing that it can be considered as a context theory in which words are represented as projections on the vector space of documents. This leads us to an implementation of our own in which we used latent Dirichlet allocation as an alternative approach to overcoming the problem of data sparseness.

A fair amount of effort is required to describe these approaches within our framework. Although there is no immediate practical benefit to be gained from this, our main purpose in doing this is to demonstrate the generality of the framework. We also hope that insight into these approaches may be gleaned by viewing them from a new perspective.

4.1 Subsequence Matching and Lexical Overlap

A sequence $x \in A^*$ is a **subsequence** of $y \in A^*$ if each element of x occurs in y in the same order, but with the possibility of other elements occurring in between, so for example *abba* is a subsequence of *acabcba* in $\{a, b, c\}^*$. Subsequence matching compares the subsequences of two sequences: The more subsequences they have in common the more similar they are assumed to be. This idea has been used successfully in text classification (Lodhi et al. 2002) and also formed the basis of the author's entry to the second Recognising Textual Entailment Challenge (Clarke 2006).

If S is a semigroup, $\ell^1(S)$ is a lattice-ordered algebra under the multiplication of convolution:

$$(f \cdot g)(x) = \sum_{yz=x} f(y)g(z) \tag{47}$$

where $x, y, z \in S, f, g \in \ell^1(S)$.

Example 18 (Subsequence Matching)
Consider the algebra $\ell^1(A^*)$ for some alphabet A. This has a basis consisting of elements e_x for $x \in A^*$, where e_x the function that is 1 on x and 0 elsewhere. In particular e_ϵ is a unity for the algebra. Define $\xi(a) = \frac{1}{2}(e_a + e_\epsilon)$; then $\langle A, \ell^1(A^*), \xi \rangle$ is a context theory. Under this context theory, a sequence x completely entails y if and only if it is a subsequence of y. In our experiments, we have shown that this type of context theory can perform significantly better than straightforward lexical overlap (Clarke 2006). Many variations on this idea are possible: for example, using more complex mappings from A^* to $\ell^1(A^*)$.

Example 19 (Lexical Overlap)
The simplest approach to textual entailment is to measure the degree of lexical overlap: the proportion of words in the hypothesis sentence that are contained in the text sentence (Dagan, Glickman, and Magnini 2005). This approach can be described as a context theory in terms of a free commutative semigroup on a set A, defined by $A^*/\equiv$ where $x \equiv y$ in A^* if the symbols making up x can be reordered to make y. Then define ξ' by $\xi'(a) = \frac{1}{2}(e_{[a]} + e_{[\epsilon]})$ where $[a]$ is the equivalence class of a in $A^*/\equiv$. Then $\langle A, \ell^1(A^*/\equiv), \xi' \rangle$ is a context theory in which entailment is defined by lexical overlap. More complex definitions of $\hat{x}$ can be used, for example, to weight different words by their probabilities.

4.2 Document Projections

Glickman and Dagan (2005) give a probabilistic definition of entailment in terms of "possible worlds" which they use to justify their lexical entailment model based on occurrences of words in Web documents. They estimate the lexical entailment probability $\textsc{lep}(u,v)$ to be

$$\textsc{lep}(u,v) \simeq \frac{n_{u,v}}{n_v} \tag{48}$$

where n_v and $n_{u,v}$ denote the number of documents in which the word v occurs and in which the words u and v both occur, respectively. From the context-theoretic perspective, we view the set of documents in which the word occurs as its context vector. To describe this situation in terms of a context theory, consider the vector space $\ell^\infty(D)$ where D is the set of documents. With each word u in some set A we associate an operator P_u on this vector space by

$$P_u e_d = \begin{cases} e_d & \text{if } u \text{ occurs in document } d \\ 0 & \text{otherwise.} \end{cases} \tag{49}$$

where e_d is the basis element associated with document $d \in D$. P_u is a projection, that is, $P_u P_u = P_u$; it projects onto the space of documents that u occurs in. These projections are clearly commutative (they are in fact **band projections**): $P_u P_v = P_v P_u = P_u \wedge P_v$ projects onto the space of documents in which both u and v occur.

In their paper, Glickman and Dagan (2005) assume that probabilities can be attached to individual words, as we do, although they interpret these as the probability that a word is "true" in a possible world. In their interpretation, a document corresponds to a possible world, and a word is true in that world if it occurs in the document.

They do not, however, determine these probabilities directly; instead they make assumptions about how the entailment probability of a sentence depends on lexical entailment probability. Although they do not state this, the reason for this is presumably data sparseness: They assume that a sentence is true if all its lexical components are true; this will only happen if all the words occur in the same document. For any sizeable sentence this is extremely unlikely, hence their alternative approach.

It is nevertheless useful to consider this idea from a context-theoretic perspective. We define a context theory $\langle A, B(\ell^\infty(D)), \xi, \ell^\infty(D), \bar{p}\rangle$, where:

- We denote by $B(U)$ the set of bounded operators on the vector space U; in this case we are considering the bounded operators on the vector space indexed by the set of documents D. Because D is finite, all operators on this space are in fact bounded; this property will be needed when we generalize D to an infinite set, however.
- $\xi : A \to B(\ell^\infty(D))$ is defined by $\xi(u) = P_u$; it maps words to document projections.
- $\bar{p} : B(\ell^\infty(D)) \to \ell^\infty(D)$ is a map defined by $\bar{p}(T) = Tp$, where
- $p \in \ell^\infty(D)$ is defined by $p(d) = 1/|D|$ for all $d \in D$. This is defined such that $\|P_u p\|_1$ is the probability of the term u.

The degree to which x entails y is then given by $\|P_x p \wedge P_y p\|_1 / \|P_x\|_1 = \|P_x P_y p\|_1 / \|P_x\|_1$. This corresponds directly to Glickman and Dagan's (2005) entailment "confidence"; it is simply the proportion of documents that contain all the terms of x which also contain all the terms of y.

4.3 Latent Dirichlet Projections

The formulation in the previous section suggests an alternative approach to that of Glickman and Dagan (2005) to cope with the data sparseness problem. We consider the finite data available D as a sample from a distributional language D'; the vector p then becomes a probability distribution over the documents in D'. In our own experiments, we used latent Dirichlet allocation (Blei, Ng, and Jordan 2003) to build a model of the corpus as a probabilistic language based on a subset of around 380,000 documents from the Gigaword corpus. Having this model allows us to consider an infinite array of possible documents, and thus we can use our context-theoretic definition of entailment because there is no problem of data sparseness.

Latent Dirichlet allocation (LDA) follows the same vein as latent semantic analysis (LSA; Deerwester et al. 1990) and probabilistic latent semantic analysis (PLSA; Hofmann 1999) in that it can be used to build models of corpora in which words within a document are considered to be exchangeable, so that a document is treated as a bag of words. LSA performs a singular value decomposition on the matrix of words and documents which brings out hidden "latent" similarities in meaning between words, even though they may not occur together.

In contrast, PLSA and LDA provide probabilistic models of corpora using Bayesian methods. LDA differs from PLSA in that, whereas the latter assumes a fixed number of documents, LDA assumes that the data at hand are a sample from an infinite set of documents, allowing new documents to be assigned probabilities in a straightforward manner.

Figure 3 shows a graphical representation of the latent Dirichlet allocation generative model, and Figure 4 shows how the model generates a document of length N. In this model, the probability of occurrence of a word w in a document is considered to be a multinomial variable conditioned on a k-dimensional "topic" variable z. The number of topics k is generally chosen to be much fewer than the number of possible words, so that topics provide a "bottleneck" through which the latent similarity in meaning between words becomes exposed.

The topic variable is assumed to follow a multinomial distribution parameterized by a k-dimensional variable θ, satisfying

$$\sum_{i=1}^{k} \theta_i = 1 \tag{50}$$

and which is in turn assumed to follow a Dirichlet distribution. The Dirichlet distribution is itself parameterized by a k-dimensional vector α. The components of this vector can be viewed as determining the marginal probabilities of topics, because

$$p(z_i) = \int p(z_i|\theta)p(\theta)d\theta \tag{51}$$

$$= \int \theta_i p(\theta)d\theta \tag{52}$$

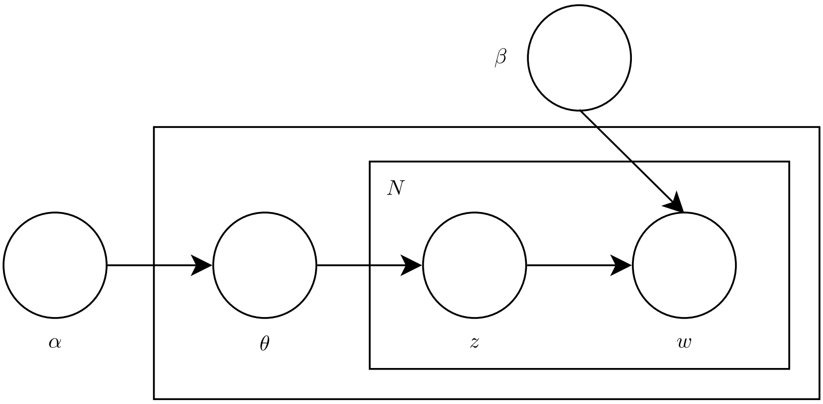

Figure 3
Graphical representation of the Dirichlet model. The inner box shows the choices that are repeated for each word in the document; the outer box shows the choice that is made for each document; the parameters outside the boxes are constant for the model.

This is just the expected value of θ_i, which is given by

$$p(z_i) = \frac{\alpha_i}{\sum_j \alpha_j} \tag{53}$$

The model is thus entirely specified by α and the conditional probabilities $p(w|z)$ that we can assume are specified in a $k \times V$ matrix β where V is the number of words in the vocabulary. The parameters α and β can be estimated from a corpus of documents by a variational expectation maximization algorithm, as described by Blei, Ng, and Jordan (2003).

LDA was applied by Blei, Ng, and Jordan (2003) to the tasks of document modeling, document classification, and collaborative filtering. They compare LDA to several techniques including PLSA; LDA outperforms these on all of the applications. LDA has been applied to the task of word sense disambiguation (Boyd-Graber, Blei, and Zhu 2007; Cai, Lee, and Teh 2007) with significant success.

Consider the vector space $\ell^\infty(A^*)$ for some alphabet A, the space of all bounded functions on possible documents. In this approach, we define the representation of a

1. Choose $\theta \sim \text{Dirichlet}(\alpha)$
2. For each of the N words:
   - (a) Choose $z \sim$ Multinomial(θ)
   - (b) Choose w according to $p(w|z)$

Figure 4
Generative process assumed in the Dirichlet model.

string x to be a projection P_x on the subspace representing the (infinite) set of documents in which all the words in string x occur. We define a vector $q(x)$ for $x \in A^*$ where $q(x)$ is the probability of string x in the probabilistic language.

Our context theory is then given by $\langle A, B(\ell^\infty(A^*)), \xi, \ell^\infty(A^*), \bar{q}\rangle$, where ξ is defined as before and $\bar{q}$ is defined as $\bar{p}$ earlier. In this case, we are considering an infinite set of possible documents, A^*, so the boundedness property becomes important. $\|P_x q\|_1$ is thus the probability that a document chosen at random contains all the words that occur in string x. In order to estimate this value we have to integrate over the Dirichlet parameter θ:

$$\|P_x q\|_1 = \int_\theta \left(\prod_{a \in x} p_\theta(a) \right) p(\theta) d\theta \tag{54}$$

where by $a \in x$ we mean that the word a occurs in string x, and $p_\theta(a)$ is the probability of observing word a in a document generated by the parameter θ. We estimate this by

$$p_\theta(a) \simeq 1 - \left(1 - \sum_z p(a|z)p(z|\theta) \right)^N \tag{55}$$

where we have assumed a fixed document length N. This formula is an estimate of the probability of a word occurring at least once in a document of length N, and the sum over the topic variable z is the probability that the word a occurs at any one point in a document given the parameter θ. We approximated the integral using Monte Carlo sampling to generate values of θ according to the Dirichlet distribution.

We built a latent Dirichlet allocation model using Blei, Ng, and Jordan's (2003) implementation on documents from the British National Corpus, using 100 topics. We evaluated this model on the 800 entailment pairs from the first Recognizing Textual Entailment Challenge test set.[1] Results were comparable to those obtained by Glickman and Dagan (2005) (see Table 2). In this table, Accuracy is the accuracy on the test set, consisting of 800 entailment pairs, and CWS is the confidence weighted score; see Dagan, Glickman, and Magnini (2005) for the definition. The differences between the accuracy values in the table are not statistically significant because of the small data set, although all accuracies in the table are significantly better than chance at the 1% level. The accuracy of the model is considerably lower than the state of the art, which is around 75% (Bar-Haim et al. 2006). We experimented with various document lengths and found very long documents ($N = 10^6$ and $N = 10^7$) to work best.

It is important to note that because the LDA model is commutative, the resulting context algebra must also be commutative, which is clearly far from ideal in modeling natural language.

5. The Model of Clark, Coecke, and Sadrzadeh

One of the most sophisticated proposals for a method of composition is that of Clark, Coecke, and Sadrzadeh (2008) and the more recent implementation of Grefenstette et al.

[1] We have so far only used data from the first challenge, because we performed the experiment before the other challenges had taken place.

Table 2
Results obtained with our latent Dirichlet projection model on the data from the first Recognizing Textual Entailment Challenge for two document lengths $N = 10^6$ and $N = 10^7$ using a cut-off for the degree of entailment of 0.5 at which entailment was regarded as holding.

Model	**Accuracy**	**CWS**
Dirichlet (10^6)	0.584	0.630
Dirichlet (10^7)	0.576	0.642
Bayer (MITRE)	0.586	0.617
Glickman (Bar Ilan)	0.586	0.572
Jijkoun (Amsterdam)	0.552	0.559
Newman (Dublin)	0.565	0.6

(2011). In this section, we will show how their model can be described as a context theory.

The authors describe the syntactic element of their construction using pregroups (Lambek 2001), a formalism which simplifies the syntactic calculus of Lambek (1958). These can be described in terms of **partially ordered monoids**, a monoid G with a partial ordering $\leq$ satisfying $x \leq y$ implies $xz \leq yz$ and $zx \leq zy$ for all $x, y, z \in G$.

Definition 11 (Pregroup)
Let G be a partially ordered monoid. Then G is called a pregroup if for each $x \in G$ there are elements x^l and x^r in G such that

$$x^l x \leq 1 \tag{56}$$

$$xx^r \leq 1 \tag{57}$$

$$1 \leq xx^l \tag{58}$$

$$1 \leq x^r x \tag{59}$$

If $x, y \in G$, we call y a **reduction** of x if y can be obtained from x using only Rules (56) and (57).

Pregroup grammars are defined by freely generating a pregroup on a set of basic grammatical types. Words are then represented as elements formed from these basic types, for example:

$$\begin{array}{ccc} \text{John} & \text{likes} & \text{Mary} \\ \pi & \pi^r s o^l & o \end{array} \tag{60}$$

where π, s, and o are the basic types for first person singular, statement, and object, respectively. It is easy to see that this sentence reduces to type s under the pregroup reductions.

As Clark, Coecke, and Sadrzadeh (2008) note, their construction can be generalized by endowing the grammatical type of a word with a vector nature, in addition to its semantics. We use this slightly more general construction to allow us to formulate it in the context-theoretic framework. We define an **elementary meaning space** to be the tensor product space $V = S \otimes P$ where S is a vector space representing meanings of words and P is a vector space with an orthonormal basis corresponding to the basic grammatical types in a pregroup grammar and their adjoints. We assume that meanings of words live in the tensor algebra space $T(V)$, defined by

$$T(V) = \mathbb{R} \oplus V \oplus (V \otimes V) \oplus (V \otimes V \otimes V) \oplus \cdots$$

For an element v in a particular tensor power of V, such that $v = (s_1 \otimes p_1) \otimes (s_2 \otimes p_2) \otimes \cdots \otimes (s_n \otimes p_n)$, where the p_i are basis vectors of P, then we can recover a complex grammatical type for v as the product $\gamma(v) = \gamma_1 \gamma_2 \cdots \gamma_n$, where γ_i is the basic grammatical type corresponding to p_i. We will call the vectors such as this which have a single complex type (i.e., they are not formed from a weighted sum of more than one type) **unambiguous**.

We also assume that words are represented by vectors whose grammatical type is **irreducible**: There is no pregroup reduction possible on the type. We define $\Gamma(T(V))$ as the vector space generated by all such vectors.

We will now define a product $\cdot$ on $\Gamma(T(V))$ that will make it an algebra. To do this, it suffices to define the product between two elements u_1, u_2 which are unambiguous and whose grammatical type is basic, so that they can be viewed as elements of V. The definition of the product on the rest of the space follows from the assumption of distributivity. We define

$$u_1 \cdot u_2 = \begin{cases} u_1 \otimes u_2 & \text{if } \gamma(u_1 \otimes u_2) \text{ is irreducible} \\ \langle u_1, u_2 \rangle & \text{otherwise.} \end{cases} \tag{61}$$

This product is bilinear, because for a particular pair of basis elements, only one of these two conditions will apply, and both the tensor and inner products are bilinear functions. Moreover, it corresponds to composed and reduced word vectors, as defined in Clark, Coecke, and Sadrzadeh (2008).

To see how this works on our example sentence, we assume we have vectors for the meanings of the three words, which we write as v_{word}. We assume for the purpose of this example that the word *like* is represented as a product state composed of three vectors, one for each basic grammatical type. This removes any potentially interesting semantics, but allows us to demonstrate the product in a simple manner. We write this as follows:

$$\begin{array}{ccccc} \text{John} & & \text{likes} & & \text{Mary} \\ (v_{\text{John}} \otimes e_\pi) & \cdot & (v_{\text{likes},1} \otimes e_{\pi^r}) \cdot (v_{\text{likes},2} \otimes e_s) \cdot (v_{\text{likes},3} \otimes e_{o^l}) & \cdot & (v_{\text{Mary}} \otimes e_o) \end{array} \tag{62}$$

where e_γ is the orthonormal basis vector corresponding to basic grammatical type γ. More interesting representations of *like* would consist of sums over similar vectors. Computing this product from left to right:

$$\begin{aligned} & (v_{\text{John}} \otimes e_\pi) \cdot (v_{\text{likes},1} \otimes e_{\pi^r}) \cdot\ (v_{\text{likes},2} \otimes e_s) \ \cdot \ (v_{\text{likes},3} \otimes e_{o^l}) \cdot (v_{\text{Mary}} \otimes e_o) \\ = \ & \langle v_{\text{John}}, v_{\text{likes},1} \rangle \ (v_{\text{likes},2} \otimes e_s) \ \cdot \ (v_{\text{likes},3} \otimes e_{o^l}) \cdot (v_{\text{Mary}} \otimes e_o) \\ = \ & \langle v_{\text{John}}, v_{\text{likes},1} \rangle \ (v_{\text{likes},2} \otimes e_s) \otimes (v_{\text{likes},3} \otimes e_{o^l}) \cdot (v_{\text{Mary}} \otimes e_o) \\ = \ & \langle v_{\text{John}}, v_{\text{likes},1} \rangle \langle v_{\text{likes},3}, v_{\text{Mary}} \rangle \ (v_{\text{likes},2} \otimes e_s) \end{aligned} \tag{63}$$

As we would expect in this simplified example the product is a scalar multiple of the second vector for *like*, with the type of a statement.

This construction thus allows us to represent complex grammatical types, similar to Clark, Coecke, and Sadrzadeh (2008), although it also allows us to take weighted sums of these complex types, giving us a powerful method of expressing syntactic and semantic ambiguity.

6. Conclusions and Future Work

We have presented a context-theoretic framework for natural language semantics. The framework is founded on the idea that meaning in natural language can be determined by context, and is inspired by techniques that make use of statistical properties of language by analyzing large text corpora. Such techniques can generally be viewed as representing language in terms of vectors. These techniques are currently used in applications such as textual entailment recognition, although the lack of a theory of meaning that incorporates these techniques means that they are often used in a somewhat ad hoc manner. The purpose behind the framework is to provide a unified theoretical foundation for such techniques so that they may be used in a principled manner.

By formalizing the notion of "meaning as context" we have been able to build a mathematical model that informs us about the nature of meaning under this paradigm. Specifically, it gives us a theory about how to represent words and phrases using vectors, and tells us that the product of two meanings should be distributive and associative. It also gives us an interpretation of the inherent lattice structure on these vector spaces as defining the relation of entailment. It tells us how to measure the size of the vector representation of a string in such a way that the size corresponds to the probability of the string.

We have demonstrated that the framework encompasses several related approaches to compositional distributional semantics, including those based on a predefined composition operation such as addition (Landauer and Dumais 1997; Foltz, Kintsch, and Landauer 1998; Mitchell and Lapata 2008) or the tensor product (Smolensky 1990; Clark and Pulman 2007; Widdows 2008), matrix multiplication (Rudolph and Giesbrecht 2010), and the more sophisticated construction of Clark, Coecke, and Sadrzadeh (2008).

6.1 Practical Investigations

Section 4 raises many possibilities for the design of systems to recognize textual entailment within the framework.

- Variations on substring matching: experiments with different weighting schemes for substrings, allowing partial commutativity of words or phrases, and replacing words with vectors representing their context, using tensor products of these vectors instead of concatenation.
- Extensions of Glickman and Dagan's approach and our own context-theoretic approach using LDA, perhaps using other distributional languages based on *n*-grams or other models in which words do not commute, or a combination of context theories based on commutative and non-commutative models.

- The LDA model we used is a commutative one. This is a considerable simplification of what is possible within the context-theoretic framework; it would be interesting to investigate methods of incorporating non-commutativity into the model.
- Implementations based on the approach to representing uncertainty in logical semantics similar to those described in Clarke (2007).

All of these ideas could be evaluated using the data sets from the Recognising Textual Entailment Challenges.

There are many approaches to textual entailment that we have not considered here; we conjecture that variations of many of them could be described within our framework. We leave the task of investigating the relationship between these approaches and our framework to further work.

Other areas that we are investigating, together with researchers at the University of Sussex, is the possibility of learning finite-dimensional algebras directly from corpus data, along the lines of Guevara (2011) and Baroni and Zamparelli (2010).

One question we have not addressed in this article is the feasibility of computing with algebraic representations. Although this question is highly dependent on the particular context theory chosen, it is possible that general algorithms for computation within this framework could be found; this is another area that we intend to address in further work.

6.2 Theoretical Investigations

Although the context-theoretic framework is an abstraction of the model of meaning as context, it would be good to have a complete understanding of the model and the types of context theories that it allows. Tying down these properties would allow us to define algebras that could truly be called "context theories."

The context-theoretic framework shares a lot of properties with the study of free probability (Voiculescu 1997). It would be interesting to investigate whether ideas from free probability would carry over to context-theoretic semantics.

Although we have related our model to many techniques described in the literature, we still have to investigate its relationship with other models such as that of Song and Bruza (2003) and Guevara (2011).

We have not given much consideration here to the issue of multi-word expressions and non-compositionality. What predictions does the context-theoretic framework make about non-compositionality? Answering this may lead us to new techniques for recognizing and handling multi-word expressions and non-compositionality.

Of course it is hard to predict the benefits that may result from what we have presented, because we have given a way of thinking about meaning in natural language that in many respects is new. This new way of thinking opens the door to the unification of logic-based and vector-based methods in computational linguistics, and the potential fruits of this union are many.

Acknowledgments

The ideas presented here have benefitted enormously from the input and support of my DPhil supervisor, David Weir, without whom this work would not exist; Rudi Lutz; and Stephen Clark, who really grokked this and made many excellent suggestions for improvements. I am also grateful for the advice and encouragement of Bill Keller, John Carroll, Peter Williams, Mark W. Hopkins, Peter Lane, Paul Hender, and Peter Hines. I am indebted to the anonymous reviewers; their suggestions have undoubtedly improved this article beyond measure; the paragraph on the three uses of the term *entailment* was derived directly from one of their suggestions.

References

Abramovich, Yuri A. and Charalambos D. Aliprantis. 2002. *An Invitation to Operator Theory*. American Mathematical Society, Providence, RI.

Bar-Haim, Roy, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 1–9, Venice.

Baroni, Marco and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 1183–1193, East Stroudsburg PA.

Bellegarda, Jerome R. 2000. Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE*, 88(8):1279–1296.

Blackburn, Patrick and Johan Bos. 2005. *Representation and Inference for Natural Language*. CSLI Publications, Stanford, CA.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Boyd-Graber, Jordan, David Blei, and Xiaojin Zhu. 2007. A topic model for word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1024–1033, Prague.

Cai, Junfu, Wee Sun Lee, and Yee Whye Teh. 2007. Improving word sense disambiguation using topic features. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1015–1023, Prague.

Choi, Freddy, Peter Wiemer-Hastings, and Johanna Moore. 2001. Latent Semantic Analysis for text segmentation. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 109–117, Ithaca, NY.

Clark, Stephen, Bob Coecke, and Mehrnoosh Sadrzadeh. 2008. A compositional distributional model of meaning. In *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*, pages 133–140, Oxford.

Clark, Stephen and Stephen Pulman. 2007. Combining symbolic and distributional models of meaning. In *Proceedings of the AAAI Spring Symposium on Quantum Interaction*, pages 52–55, Stanford, CA.

Clarke, Daoud. 2006. Meaning as context and subsequence analysis for textual entailment. In *Proceedings of the Second PASCAL Recognising Textual Entailment Challenge*, pages 134–139, Venice.

Clarke, Daoud. 2007. *Context-theoretic Semantics for Natural Language: An Algebraic Framework*. Ph.D. thesis, Department of Informatics, University of Sussex.

Collins, Michael. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23, Madrid.

Dagan, Ido, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 1–8, Southampton, UK.

Deerwester, Scott, Susan Dumais, George Furnas, Thomas Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Erk, Katrin and Sebastian Padó. 2009. Paraphrase assessment in structured vector space: Exploring parameters and datasets. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 57–65, Athens.

Firth, John R. 1968. A synopsis of linguistic theory, 1930–1955. In John R. Firth, editor, *Selected Papers of JR Firth, 1952–59*. Indiana University Press, Bloomington, pages 168–205.

Foltz, Peter W., Walter Kintsch, and Thomas K. Landauer. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse Process*, 15:285–307.

Geffet, M. and I. Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 107–114, Ann Arbor, MI.

Glickman, O. and I. Dagan. 2005. A probabilistic setting and lexical cooccurrence model for textual entailment. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 43–48, Ann Arbor, MI.

Grefenstette, Edward, Mehrnoosh Sadrzadeh, Stephen Clark, Bob Coecke, and Stephen Pulman. 2011. Concrete sentence spaces for compositional distributional models of meaning. *Proceedings of the 9th International Conference on Computational Semantics (IWCS 2011)*, pages 125–134, Oxford.

Grefenstette, Gregory. 1994. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, Dordrecht, Netherlands.

Guevara, Emiliano. 2011. Computing semantic compositionality in distributional semantics. In *Proceedings of the 9th International Conference on Computational Semantics (IWCS 2011)*, pages 135–144, Oxford.

Halmos, Paul. 1974. *Finite Dimensional Vector Spaces*. Springer, Berlin.

Harris, Zellig. 1968. *Mathematical Structures of Language*. Wiley, New York.

Hofmann, Thomas. 1999. Probabilistic latent semantic analysis. In *Proceedings of the 15th Conference on Uncertainty in AI*, pages 289–296, Stockholm.

Kamp, Hans and Uwe Reyle. 1993. *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, volume 42 of *Studies in Linguistics and Philosophy*. Kluwer, Dordrecht.

Kintsch, Walter. 2001. Predication. *Cognitive Science*, 25:173–202.

Lambek, Joachim. 1958. The mathematics of sentence structure. *American Mathematical Monthly*, 65:154–169.

Lambek, Joachim. 1961. On the calculus of syntactic types. In Roman Jakobson, editor, *Structure of Language and Its Mathematical Aspects*, pages 166–178, American Mathematical Society, Providence, RI.

Lambek, Joachim. 2001. Type grammars as pregroups. *Grammars*, 4(1):21–39.

Landauer, Thomas K. and Susan T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104(2):211–240.

Lee, Lillian. 1999. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL-1999)*, pages 23–32, College Park, MD.

Lin, Dekang. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL '98)*, pages 768–774, Montreal.

Lin, Dekang. 2003. Dependency-based evaluation of MINIPAR. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 317–330, Kluwer, Dordrecht.

Lodhi, Huma, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444.

McCarthy, Diana, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)*, pages 279–286, Barcelona.

Miller, George A. and Walter G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.

Mitchell, Jeff and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, OH.

Preller, Anne and Mehrnoosh Sadrzadeh. 2011. Bell states and negative sentences in the distributed model of meaning. *Electronic Notes in Theoretical Computer Science*, 270(2):141–153.

Rudolph, Sebastian and Eugenie Giesbrecht. 2010. Compositional matrix-space models of language. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 907–916, Uppsala.

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

Smolensky, Paul. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46(1-2):159–216.

Song, Dawei and Peter D. Bruza. 2003. Towards context-sensitive information inference. *Journal of the American Society for Information Science and Technology (JASIST)*, 54:321–334.

Voiculescu, Dan-Virgil. 1997. *Free Probability Theory*. American Mathematical Society, Providence, RI.

Weeds, Julie, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of CoLING 2004*, pages 1015–1021, Geneva.

Widdows, Dominic. 2008. Semantic vector products: Some initial investigations. In *Proceedings of the Second Symposium on Quantum Interaction*, pages 1–8, Oxford.

Wittgenstein, Ludwig. 1953. *Philosophical Investigations*. Macmillan, New York. G. Anscombe, translator.