RELPRON: A Relative Clause Evaluation Data Set for Compositional Distributional Semantics

Laura Rimell[*]
University of Cambridge
Computer Laboratory

Jean Maillard[*]
University of Cambridge
Computer Laboratory

Tamara Polajnar[*]
University of Cambridge
Computer Laboratory

Stephen Clark[*]
University of Cambridge
Computer Laboratory

This article introduces RELPRON, a large data set of subject and object relative clauses, for the evaluation of methods in compositional distributional semantics. RELPRON targets an intermediate level of grammatical complexity between content-word pairs and full sentences. The task involves matching terms, such as "wisdom," with representative properties, such as "quality that experience teaches." A unique feature of RELPRON is that it is built from attested properties, but without the need for them to appear in relative clause format in the source corpus. The article also presents some initial experiments on RELPRON, using a variety of composition methods including simple baselines, arithmetic operators on vectors, and finally, more complex methods in which argument-taking words are represented as tensors. The latter methods are based on the Categorial framework, which is described in detail. The results show that vector addition is difficult to beat—in line with the existing literature—but that an implementation of the Categorial framework based on the Practical Lexical Function model is able to match the performance of vector addition. The article finishes with an in-depth analysis of RELPRON, showing how results vary across subject and object relative clauses, across different head nouns, and how the methods perform on the subtasks necessary for capturing relative clause

[*] University of Cambridge Computer Laboratory, William Gates Building, 15 JJ Thomson Avenue, Cambridge, UK. E-mail: {`laura.rimell`; `jean.maillard`; `tamara.polajnar`; `stephen.clark`}`@cl.cam.ac.uk`.

Submission received: 10 April 2015; revised version received: 15 April 2016; accepted for publication: 25 May 2016.

doi:10.1162/COLI_a_00263

semantics, as well as providing a qualitative analysis highlighting some of the more common errors. Our hope is that the competitive results presented here, in which the best systems are on average ranking one out of every two properties correctly for a given term, will inspire new approaches to the RELPRON ranking task and other tasks based on linguistically interesting constructions.

1. Introduction

The field of compositional distributional semantics (Mitchell and Lapata 2008; Clark, Coecke, and Sadrzadeh 2008; Baroni, Bernardi, and Zamparelli 2014) integrates distributional semantic representations of words (Schütze 1998; Turney and Pantel 2010; Clark 2015) with formal methods for composing word representations into larger phrases and sentences (Montague 1970; Dowty, Wall, and Peters 1981). In recent years a number of composition methods have been proposed, including simple arithmetic operations on distributional word vectors (Mitchell and Lapata 2008, 2010), multi-linear operations involving higher-order representations of argument-taking words such as verbs and adjectives (Baroni and Zamparelli 2010; Coecke, Sadrzadeh, and Clark 2010), and composition of distributed word vectors learned with neural networks (Socher, Manning, and Ng 2010; Mikolov, Yih, and Zweig 2013). To compare such approaches it is important to have high-quality data sets for evaluating composed phrase representations at different levels of granularity and complexity.

Existing evaluation data sets fall largely into two categories. Some data sets focus on two to three word phrases consisting of content words only (i.e., no closed-class function words), including subject–verb, verb–object, subject–verb–object, and adjective–noun combinations (Mitchell and Lapata 2008, 2010; Baroni and Zamparelli 2010; Vecchi, Baroni, and Zamparelli 2011; Grefenstette and Sadrzadeh 2011; Boleda et al. 2012; Boleda et al. 2013; Lazaridou, Vecchi, and Baroni 2013). Such data sets have been essential for evaluating the first generation of compositional distributional models, but the relative grammatical simplicity of the phrases—in particular, the absence of function words—leaves open a wide range of compositional phenomena in natural language against which models must be evaluated.[1]

Other data sets, including those used in recent SemEval and *SEM Shared Tasks (Agirre et al. 2012, 2013; Marelli et al. 2014; Agirre 2015), focus on pairs of full sentences, some as long as 20 words or more. The evaluation is based on a numeric similarity measure for each sentence pair. These data sets represent a more realistic task for language technology applications, but reducing sentence similarity to a single value makes it difficult to identify how a model performs on subparts of a sentence, or on specific grammatical constructions, masking the areas where models need improvement.[2]

In the domain of syntactic parsing, a call has been made for "grammatical construction-focused" parser evaluation (Rimell, Clark, and Steedman 2009; Bender et al. 2011), focusing on individual, often challenging syntactic structures, in order to tease out parser performance in these areas from overall accuracy scores. We make an analogous call here, for a wider range of compositional phenomena to be investigated in compositional distributional semantics in the near future.

1 Exceptions are the data sets of Bernardi et al. (2013) and Pham et al. (2013), which include determiners such as *two*, *most*, and *no* in the phrases to be composed; see Section 2.1.

2 The pilot subtask on interpretable Semantic Textual Similarity (Agirre 2015) begins to address this problem.

This article begins to answer that call by presenting RELPRON, a data set of noun phrases consisting of a noun modified by a relative clause. For example, *building that hosts premieres* is a noun phrase containing a subject relative clause (a relative clause with an extracted subject), and in our data set describes a *theater*; while *person that helicopter saves* contains an object relative clause, and in our data set describes a *survivor*. The RELPRON data set is primarily concerned with the analysis of a particular type of closed-class function word, namely, relative pronouns (*that*, in our examples); function words have traditionally been of greater concern than content words for formal semantics, and we address how this focus can be extended to distributional semantics.

Relative clauses, although still fairly short and therefore more manageable than full sentences, are nevertheless more grammatically complex than the short phrases in previous data sets, because of the relative pronoun and the long-distance relationship between the verb and the extracted argument—known as the head noun of the relative clause (*building, person*). The aim of RELPRON is to expand the variety of phrase types on which compositional distributional semantic methods can be evaluated, thus helping to build these methods up step-by-step to the full task of compositional sentence representations.

RELPRON is a large (1,087 relative clauses), corpus-based, naturalistic data set, including both subject and object relative clauses that modify a variety of concrete and abstract nouns. The relative clauses are matched with terms and represent distinctive properties of those terms, such as *wisdom: quality that experience teaches*, and *bowler: player that dominates batsman*. A unique feature of RELPRON is that it is built from attested properties of terms, but without the properties needing to appear in relative clause form in the source corpus.

This article also presents some initial experiments on RELPRON, using a variety of composition methods. We find that a simple arithmetic vector operation provides a challenging baseline, in line with existing literature evaluating such methods, but we are able to match this baseline using a more sophisticated method similar to the Practical Lexical Function (PLF) model of Paperno, Pham, and Baroni (2014). We hope that the compositional methods presented here will inspire new approaches.

The remainder of the article is organized as follows. Section 2 provides some motivation for the RELPRON data set and the canonical ranking task that we imagine RELPRON being used for, as well as a description of existing data sets designed to test compositional distributional models. Section 3 describes the data set itself, and provides a detailed description of how it was built. Section 4 surveys some of the existing compositional methods that have been applied in distributional semantics, including a short historical section describing work from cognitive science in the 1980s and 1990s. Section 4.2 provides a fairly detailed description of the Categorial framework, which provides the basis for the more complex composition methods that we have tested, as well as a description of existing implementations of the framework. Section 5 provides further details of the composition methods, including lower-level details of how the vectors and tensors were built. Section 6 presents our results on the development and test portions of RELPRON, comparing different composition methods and different methods of building the vectors and tensors (including count-based vectors vs. neural embeddings). Finally, Section 7 provides an in-depth analysis of RELPRON, showing how results vary across subject and object relative clauses, across the different head nouns, and how the methods perform on the subtasks necessary for capturing relative pronoun semantics, as well as providing a qualitative analysis highlighting some of the more common errors. Section 8 concludes.

2. Motivation for RELPRON

Relative clauses have been suggested as an interesting test case for compositional distributional semantic methods by Sadrzadeh, Clark, and Coecke (2013) and Baroni, Bernardi, and Zamparelli (2014). One of the main inspirations for RELPRON is the pilot experiment of Sadrzadeh, Clark, and Coecke, in which terms such as *carnivore* are matched with descriptions such as *animal who eats meat*. However, the data set is small and all examples are manually constructed. RELPRON aims to provide a larger, more realistic, and more challenging testing ground for relative clause composition methods.

RELPRON falls into the general category of *cross-level semantic similarity* tasks (Jurgens, Pilehvar, and Navigli 2014), which involve comparing phrases of different lengths; in this case, a single word with a short phrase. The task of matching a phrase with a single word can be thought of as a special case of paraphrase recognition. However, restricting one of the phrases to a word means that the experiment is better controlled with regard to composition methods; because we know that we have available high-quality, state-of-the-art distributional representations of single words, the only unknown is the composed phrase representation, whereas a comparison of two composed phrases can be more difficult to interpret.

The RELPRON task is similar to the definition classification task of Kartsaklis, Sadrzadeh, and Pulman (2012), which targeted verb phrase composition. In that task, methods were tested on their ability to match a term (in this case a verb) with its definition (usually of the form verb–object), such as *embark: enter boat or vessel*. However, the definitions of Kartsaklis, Sadrzadeh, and Pulman were mined from a set of dictionaries, whereas the RELPRON relative clauses are not limited to dictionary definitions.

Because there is an intuitive definition-like "flavor" to some relative clauses, we considered using dictionary definitions as a source of relative clauses for the data set. However, in practice we found that short, natural definitions in relative clause format are rare in dictionaries. For example, definitions for the word *telescope* from three different dictionaries, two of them learner dictionaries, are shown in Example (1).

telescope	a monocular optical instrument possessing magnification for observing distant objects (Wiktionary)	
telescope	an important tool for astronomy that gathers and focuses light (Simple Wikipedia)	(1)
telescope	a cylinder-shaped device for making objects that are far away look nearer and larger, using lenses and curved mirrors (Cambridge Advanced Learner's Dictionary)	

Only the Simple Wikipedia definition uses a relative clause to define the term, and would still require significant editing to shorten the phrase. Instead, we collect relative clauses that express representative properties, not necessarily definitions, of the term, as seen in Example (2).

telescope	device that astronomers use	
telescope	device that detects planets	(2)
telescope	device that observatory has	

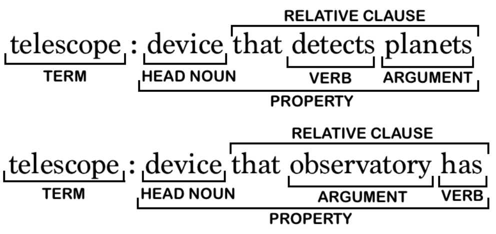

Figure 1
Terminology used to describe the terms and properties in RELPRON: subject relative clause above, object relative clause below.

Collecting properties rather than definitions allows for several phrases to be associated with each term, and substantially broadens the number and type of corpora that can be used as sources for the relative clauses.

Figure 1 introduces some terminology for describing the terms and properties in the data set.[3] A **term** is a single word which is associated with a set of descriptive phrases. Semantically, each descriptive phrase is a **property** of the term. Syntactically, each descriptive phrase is a **noun phrase containing a relative clause**. For the sake of brevity, we often refer to the whole noun phrase containing the relative clause as a **relative clause**, when we believe this will not lead to confusion. We say **subject relative clause** or **object relative clause** (or **subject property** or **object property**) to indicate which argument of the verb has been extracted. The **head noun** of the relative clause is the extracted noun, and is a hypernym of the term; for example, *device* is a hypernym of *telescope*. The subject or object that remains in situ in the clause we call the **argument** of the verb, and the general term for the subject or object role is **grammatical function**. The relative clauses in RELPRON are restrictive relative clauses, because they narrow down the meaning of the head noun. In contrast, a non-restrictive relative clause provides incidental situational information; for example, *a device, which was in the room yesterday*.

Our goal was to construct a data set containing at least 1,000 properties in the form of relative clauses. To make the ranking task more challenging, we chose groups of terms that share a head noun. For example, *telescope*, *watch*, *button*, and *pipe* are all subclasses of *device*, so their relative clauses will all begin *device that....* Because the relative clauses are not definitions, each term can be associated with multiple properties; we selected between four and ten properties per term.

The construction of RELPRON also takes a novel approach to producing evaluation data sets that target a particular grammatical construction. Rather than mining examples of the construction directly from a corpus, which may be difficult because of data sparsity, the properties in RELPRON did not have to occur in relative clause format in our source corpus. Instead, the properties were obtained from subject–verb–object (SVO) triples such as *astronomer uses telescope*, and joined with head nouns such as *device*. The head nouns are also attested in the corpus as heads of relative clauses in general, though not necessarily with these specific properties. The relative clauses in RELPRON are therefore *naturalistic, plausible*, and *corpus-based*, without having to be attested in the exact relative clause form in which they ultimately appear in the data set.

3 For readability, we sometimes present verbs and nouns in their inflected form in relative clauses in this article, e.g., *detects planets* rather than *detect planet*. In the data set, all words are lemmatized.

The canonical task on the RELPRON data set is conceived as a ranking task. Given a term, a system must rank all properties by their similarity to the term. The ideal system will rank all properties corresponding to that term above all other properties. This is analogous to an Information Retrieval task (Manning, Raghavan, and Schütze 2008), where the term is the query and the properties are the documents, with the identifying properties for a given term as the relevant documents. Evaluation can make use of standard Information Retrieval measures; in this article we use Mean Average Precision (MAP).

2.1 Existing Evaluation Data Sets

Most existing evaluation data sets for compositional distributional semantics have been based on small, fixed syntactic contexts, typically adjective–noun or subject–verb–object. Mitchell and Lapata (2008) introduce an evaluation that has been adopted by a number of other researchers. The task is to predict human similarity judgments on subject–verb phrases, where the similarity rating depends on word sense disambiguation in context. For example, *horse run* is more similar to *horse gallop* than to *horse dissolve*, whereas *colors run* is more similar to *colors dissolve* than to *colors gallop*. Compositional models are evaluated on how well their similarity judgments correlate with the human judgments. Mitchell and Lapata (2010) introduce a phrase similarity data set consisting of adjective–noun, verb–object, and noun–noun combinations, where the task again is to produce similarity ratings that correlate well with human judgments. For example, *large number* and *great majority* have high similarity in the gold-standard data, whereas *further evidence* and *low cost* have low similarity. This data set formed the basis of a shared task at the GEMS 2011 workshop (Padó and Peirsman 2011). Grefenstette and Sadrzadeh (2011) and Kartsaklis and Sadrzadeh (2014) introduce analogous tasks for subject–verb–object triples. In the disambiguation task, *people try door* is more similar to *people test door* than to *people judge door*. In the similarity task, *medication achieve result* and *drug produce effect* have high similarity ratings, whereas *author write book* and *delegate buy land* have low ratings. Sentence pairs with mid-similarity ratings tend to have high relatedness but are not mutually substitutable, such as *team win match* and *people play game*.

Other evaluations have made use of the contrast between compositional and non-compositional phrases, based on the intuition that a successful compositional model should produce phrase vectors similar to the observed context vectors for compositional phrases, but diverge from the observed context vectors for phrases known to be non-compositional. Reddy, McCarthy, and Manandhar (2011) introduce an evaluation based on compound nouns like *climate change* (compositional) and *gravy train* (non-compositional). Boleda et al. (2012, 2013) evaluate models on a data set involving intersective adjectives in phrases like *white towel* (compositional) and non-intersective adjectives in phrases like *black hole* and *false floor* (non-compositional). The data from the shared task of the ACL 2011 Distributional Semantics and Compositionality workshop (Biemann and Giesbrecht 2011) are also relevant in this context.

Also relevant for compositional distributional semantics are larger data sets for full sentence similarity and entailment, including those from recent SemEval and *SEM Shared Tasks (Agirre et al. 2012, 2013; Marelli et al. 2014; Agirre 2015). The SemEval Semantic Textual Similarity tasks make use of a number of paraphrase corpora that include text from news articles, headlines, tweets, image captions, video descriptions, student work, online discussion forums, and statistical machine translation. These corpora include sentences of widely varying length and quality. The

SICK data set (Marelli et al. 2014) is based on image and video description corpora, with named entities removed and other normalization steps applied, in order to remove phenomena outside the scope of compositional distributional semantic models. Additional sentences are added based on linguistic transformations that vary the meaning of the original sentence in a controlled fashion. However, this data set is not designed for targeted evaluation of models on specific compositional linguistic phenomena, especially because many of the sentence pairs exhibit significant lexical overlap; for example, *An airplane is in the air, An airplane is flying in the air*.

Some recent data sets have extended the range of linguistic phenomena that may be evaluated. Cheung and Penn (2012) test the ability of compositional models to detect semantic similarity in the presence of lexical and syntactic variation, for example, among the sentences *Pfizer buys Rinat*, *Pfizer paid several hundred million dollars for Rinat*, and *Pfizer's acquisition of Rinat*. Bernardi et al. (2013) introduce a data set comparing nouns to noun phrases, including determiners as function words; for example, *duel* is compared to *two opponents* (similar), *various opponents* (dissimilar), and *two engineers* (dissimilar). Pham et al. (2013) introduce a data set involving full sentences from a limited grammar, where determiners and word order vary; for example, *A man plays a guitar* is compared to *A guitar plays a man* and *The man plays a guitar*.

3. The RELPRON Data Set

One challenge in targeting specific grammatical constructions for evaluation is that the more linguistically interesting constructions can be relatively rare in text corpora. The development of RELPRON takes a novel approach, using a syntactic transformation to assemble properties in relative clause form without the properties needing to appear in this syntactic form in the source corpus. The properties were obtained from SVO triples such as *astronomer uses telescope*, and joined with head nouns such as *device*, a hypernym of *telescope*, to create the relative clause *device that astronomer uses*. The source corpus used for constructing the data set was the union of a 2010 Wikipedia download and the British National Corpus (BNC) (Burnard 2007). Here we describe the steps in the construction of RELPRON.

3.1 Selection of Candidate Head Nouns

To make the data set maximally challenging, we designed it with multiple terms (hyponyms) per head noun (hypernym). This ensures that the composition method cannot rely solely on the similarity between the term and the head noun—say, *traveler* and *person*—and fail to consider the verb and argument in the relative clause.

We began by selecting a set of candidate head nouns, each of which would need to have a number of good candidate terms. To improve the naturalness of the examples, we chose candidate head nouns from among nouns frequently used as heads of relative clauses in the source corpus. We used a simple regular expression search for nouns occurring in the pattern "a[n] NOUN that|who|which," considering as candidates the 200 most frequent results. We manually excluded nouns where the *that*-clause could be an argument of the noun, for example *statement*, *decision*, and *belief*.

One author (Rimell) used WordNet (Miller 1995) as a guide to manually narrow down the candidates, with the following two requirements. First, the head nouns should have a large number of WordNet hyponyms, to maximize the chances of finding appropriate terms. The hyponyms were required to be direct children or grandchildren, because we wanted to avoid extremely abstract nouns like *entity*, which has many distant hyponyms but few direct ones. Second, the hyponyms should be mostly unigrams, because we wanted to avoid multiword terms such as *remote control*. For simplicity, we considered only the first WordNet sense of each candidate head noun.[4] Occasionally, we utilized the WordNet level one down from the frequent head noun in the corpus; for example, the corpus search yielded *substance*, but we used its direct hyponym *material* as a head noun, because it had more direct hyponyms of its own.

3.2 Selection of Candidate Terms

Following the selection of candidate head nouns, we chose candidate terms. For each candidate head noun, one author (Rimell) manually selected candidate terms from among the full list of WordNet hyponyms. Only one word was chosen from each WordNet synset, on the grounds that members of the same synset would not be sufficiently distinguishable from one another by their properties. Generally this was the first unigram member of the synset. For example, the WordNet hyponyms for *building* include the synset in Example (3).

`dormitory, dorm, residence hall, hall, student residence` (3)

From this synset *dormitory* was kept as a candidate term. On the other hand, the WordNet hyponyms for *quality* include the synset in Example (4).

`economy, thriftiness` (4)

Here *economy* was discarded on the basis that its primary sense is not relevant, and *thriftiness* was kept as a candidate term. Entire synsets were discarded when they seemed to be a poor example of the head noun, seemed to be too ambiguous, or consisted only of multi-word or proper noun entries. A multi-word term was very occasionally edited to make a unigram.

Next, potential terms were filtered by frequency. Based on initial exploration of the data, we passed along for manual annotation only those candidate terms that occurred at least 5,000 times in the source corpus, to maximize the chances that there would be enough SVO triples for annotation (see Section 3.3). Finally, candidate terms that appeared in WordNet as hyponyms of more than one candidate head noun were removed. For example, *surgery* could be an *activity* or a *room*; *school* could be a *building* or an *organization*. We believed that the inclusion of such terms would complicate the data set and the evaluation unnecessarily.

3.3 Extraction of SVO Triples

For each candidate term, candidate properties were obtained from the source corpus by extraction of all triples in which the term occurred in either subject or object position. We used a version of the corpus lemmatized with Morpha (Minnen, Carroll, and Pearce

4 For example, for *person* the WordNet command for viewing the hyponym hierarchy was: `wn person -n1 -treen`.

Table 1
Sample SVO triples extracted from the source corpus for the candidate term *traveler*.

Object	**Subject**
it allow traveler	traveler make decision
website allow traveler	traveler make journey
inn serve traveler	traveler take road
business serve traveler	traveler take advantage
she become traveler	traveler have option
student become traveler	traveler have luggage

2001) and parsed with the C&C Tools (Clark and Curran 2007) to obtain the subject and object relations. A minimum frequency cutoff of six occurrences per verb, per grammatical function (subject/object), per candidate term was applied. Properties containing proper noun arguments were removed. Table 1 shows some sample triples extracted from the corpus prior to manual annotation.

3.4 Conversion to Relative Clause Form

The candidate properties were converted to relative clause form prior to annotation, by joining the head noun with the relevant part of the SVO triple, using the template *term: headnoun that V O* for a subject property, or *term: headnoun that S V* for an object property. Using the examples from Table 1, *inn serve traveler* thus became *traveler: person that inn serve*; and *traveler make journey* became *traveler: person that make journey*. For simplicity, *that* was used as the relative pronoun throughout the data set, although some head nouns are more likely to appear with *who* or *which*.

3.5 Manual Annotation of Properties

The manual annotation step was the stage at which candidate head nouns, terms, and properties were filtered for the final data set. Only terms with at least four good properties, and head nouns with at least four terms, were retained after this step. Annotation was continued until the data set reached the desired size.

Most of the SVO triples extracted from the corpus did not make good identifying properties for their terms. In many cases the SVO triple represented a property which, although it could be true of the term, did not distinguish it from other terms, especially from other hyponyms of the same head noun. Thus *player that wins tournament* is not a good property for *golfer*: many golfers do win tournaments, but so do other sports players. Similarly, *organization that money helps* is an accurate description of a *charity*, but does not distinguish it from other types of organizations. In other cases, the extracted triple contained a light verb and/or a pronoun in argument position, such as *golfer: player that he become*, or *charity: organization that these include*. These candidates were too vague to be meaningful properties. Some candidate properties were unsuitable because they were not generic. Among the candidates for *killer* were *person that win lottery* and *person that actor play*, because some specific killer in the corpus experienced these events. However, they are not descriptive of killers in general. Lexical ambiguity and parser errors contributed other unsuitable candidates.

Two of the authors (Rimell and Clark) selected good properties from among the candidates for each term, looking for what we called **identifying properties**. Such a property should distinguish a term from other hyponyms of its head noun, both those in RELPRON, and ideally other examples of the head noun as well (barring near-synonyms such as *traveler* and *voyager*). As a further guideline for annotation, we defined an identifying property as one that distinguishes a term because it *mainly* or *canonically* applies to the term in question, compared with other examples of the head noun. For example, *person that tavern serves* was not judged to be a good property for *traveler*. It does not distinguish a *traveler* from other examples of people, because serving *travelers* is not the main or canonical purpose of a tavern (since the majority of people who eat and drink in a tavern are not traveling). On the other hand, *person that hotel accommodates* is a good property, because, although hotels might accommodate other people (e.g., for meetings or in a restaurant), their main and canonical purpose is to accommodate *travelers*. A more subtle example is the property *device that bomb has*, a candidate property for *timer*. This property does not uniquely distinguish *timer* from among other devices, because bombs contain multiple (sub-)devices, but we felt that a timer was such a canonical part of a bomb that this could be considered a good property for *timer*.

For a given head noun (hypernym), the annotator began with a random selection of ten candidate terms (hyponyms) from the candidates selected as in Section 3.2. For each candidate term, the annotator chose up to ten good properties. Once ten had been found, annotation stopped on that term, even if more good properties might have been available, because the aim was a broad and balanced data set rather than recall of all good properties. Not all terms had ten good properties. We required a minimum of four, otherwise the candidate term was discarded and another candidate randomly selected from the list of candidates. We also required a minimum of four terms per head noun, otherwise the entire head noun was discarded and another chosen from the list of candidates.

The two annotators originally worked together on a small set of data. After the annotation guidelines were refined, the head nouns were divided between the annotators. As a final check, all properties were reviewed and discussed; any property on which the annotators could not agree was discarded. We aimed for a balance of subject and object relative clauses for each term, but this was not always possible. Some head nouns were very unbalanced in this respect: *person* terms tended to have better subject properties thanks to their agentivity, such as *philosopher: person that interprets world*; whereas *quality* terms tended to have better object properties because of their lack of agentivity, such as *popularity: quality that artist achieves*. We also tried to make the data set more challenging by choosing properties with lexical confounders when we noticed them. For example, *team: organization that loses match* and *division: organization that team joins* share the lemma *team*, which means a ranking method cannot rely on lexical overlap between the term and the property as an indicator of similarity. Table 2 shows the full list of selected properties for two terms in the development set, *navy* and *expert*.

Finding terms with enough good properties was a challenging task. Overall, we had to examine candidate properties for 284 candidate terms to yield the 138 terms in the final data set. Most head nouns in the final data set have ten terms, but two (*player* and *woman*) have only five terms, and one (*scientist*) has eight terms. The final list of head nouns, in alphabetical order, is: *activity, building, device, document,*

Table 2
Full list of selected properties for the terms *navy* and *expert* from the development set.

OBJ	navy: organization that sailor join	OBJ	expert: person that novice become
OBJ	navy: organization that fleet destroy	OBJ	expert: person that panel include
OBJ	navy: organization that vessel serve	OBJ	expert: person that lawyer consult
OBJ	navy: organization that battleship fight	OBJ	expert: person that report cite
SBJ	navy: organization that use submarine	SBJ	expert: person that have knowledge
SBJ	navy: organization that maintain blockade	SBJ	expert: person that give tutorial
SBJ	navy: organization that defeat fleet	SBJ	expert: person that describe model
SBJ	navy: organization that protect waters	SBJ	expert: person that write treatise
SBJ	navy: organization that establish blockade	SBJ	expert: person that develop curriculum
SBJ	navy: organization that blockade port	SBJ	expert: person that author monograph

mammal, material, organization, person, phenomenon, player, quality, room, scientist, vehicle, woman.[5]

3.6 Final Data Set

After annotation was complete, the data were divided into test and development sets. The division was made by head noun (hypernym), with all terms (and therefore properties) for a given head noun appearing in either the test or the development set. Assuming that terms with the same head noun are good mutual confounders, this split was chosen to maximize the difficulty of the task. We arranged the split so that both sets contained a range of concrete and abstract head nouns. We also ensured that both sets contained the maximum possible number of lexical confounders.

The final data set consists of 1,087 properties, comprising 15 head nouns and 138 terms. This is divided into a test set of 569 properties, comprising 8 head nouns and 73 terms; and a development set of 518 properties, comprising 7 head nouns and 65 terms. The total number of terms and properties by head noun is given in Table 3. Overall, the data set includes 565 subject and 522 object properties, with the distribution across test and development sets as shown in Table 4.

4. Composition Methods: Background

This section surveys some of the composition methods that have been proposed in the distributional semantics literature, focusing on the methods applied in this article. We also include a short section giving more of a historical perspective, contrasting current work in computational linguistics with that from cognitive science in the 1980s and 1990s. Although in our experiments we do use neural embeddings (Mikolov, Yih, and Zweig 2013) as input vectors to the composition, we leave the application of neural networks to the modeling of the relative pronoun itself as important future work. Hence in this section we leave out work using recurrent or recursive neural networks

5 Readers may wonder why *woman* and not *man* was included in the data set. This was a simple matter of statistics: *man* did not have enough WordNet hyponyms with sufficient corpus frequency, according to the criteria in Section 3.2.

Table 3
Total number of terms and properties by head noun in RELPRON.

Test Set			Development Set		
Head noun	**Terms**	**Props**	**Head noun**	**Terms**	**Props**
phenomenon	10	80	quality	10	81
activity	10	83	organization	10	99
material	10	82	device	10	76
room	10	80	building	10	69
vehicle	10	79	document	10	69
mammal	10	70	person	10	86
woman	5	37	player	5	38
scientist	8	58			
TOTAL	73	569	TOTAL	65	518

specifically designed for composition, such as Socher, Manning, and Ng (2010), Socher et al. (2013), and Hermann, Grefenstette, and Blunsom (2013).

In this section, we are agnostic about how the vector representations are built. Many of the composition techniques we describe can be equally applied to "classical" distributional vectors built using count methods (Turney and Pantel 2010; Clark 2015) and vectors built using neural embedding techniques. It may be that some composition techniques work better with one type of vector or the other—for example, there is evidence that elementwise multiplication performs better with count methods—although research comparing the two vector-building methods is still in its infancy (Baroni, Dinu, and Kruszewski 2014; Levy and Goldberg 2014; Milajevs et al. 2014; Levy et al. 2015). Section 5 provides details of how our vector representations are built in practice.

4.1 Simple Operators on Vectors

The simplest method of combining two distributional representations, for example, vectors for *fast* and *car*, is to perform some elementwise combination, such as vector

Table 4
Total number of subject and object properties by head noun in RELPRON.

Test Set			Development Set		
Head noun	**Sbj**	**Obj**	**Head noun**	**Sbj**	**Obj**
phenomenon	38	42	quality	16	65
activity	46	37	organization	54	45
material	30	52	device	40	36
room	38	42	building	37	32
vehicle	38	41	document	21	48
mammal	41	29	person	59	27
woman	20	17	player	29	9
scientist	58	0			
TOTAL	309	260	TOTAL	256	262

addition or elementwise multiplication. As a method of integrating formal and distributional semantics, these operators appear to be clearly inadequate a priori, not least because they are commutative, and so do not respect word order. Despite this fact, there is a large literature investigating how such operators can be used for phrasal composition, starting with the work of Mitchell and Lapata (2008, 2010) (M&L subsequently).

The additive model from M&L has the general form:

$$\mathbf{p} = A\mathbf{u} + B\mathbf{v} \tag{5}$$

where $\mathbf{u}$ and $\mathbf{v}$ are word (column) vectors in $\mathbb{R}^n$ (e.g., $\overrightarrow{\textit{fast}}$ and $\overrightarrow{\textit{car}}$), $\mathbf{p} \in \mathbb{R}^n$ is the vector for the phrase resulting from composition of $\mathbf{u}$ and $\mathbf{v}$ ($\overrightarrow{\textit{fast car}}$), and $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times n}$ are matrices that determine the contribution of $\mathbf{u}$ and $\mathbf{v}$ to $\mathbf{p}$. However, M&L make the simplifying assumption that only the ith components of $\mathbf{u}$ and $\mathbf{v}$ contribute to the ith component of $\mathbf{p}$, which yields the form:

$$\mathbf{p} = \alpha\mathbf{u} + \beta\mathbf{v} \tag{6}$$

The parameters $\alpha, \beta \in \mathbb{R}$ allow the contributions of $\mathbf{u}$ and $\mathbf{v}$ to be weighted differently, providing a minimal level of syntax-awareness to the model. For example, if $\mathbf{u}$ is the vector representation for an adjective, and $\mathbf{v}$ the vector representation for a noun, $\mathbf{v}$ may be given a higher weight because it provides the linguistic head for the resulting noun phrase; whereas if $\mathbf{u}$ is an intransitive verb and $\mathbf{v}$ a noun, $\mathbf{u}$ may be weighted more highly.

The multiplicative model presented by M&L has the general form:

$$\mathbf{p} = C\,(\mathbf{u} \otimes \mathbf{v}) \tag{7}$$

where $\mathbf{u} \otimes \mathbf{v}$ is the tensor, or outer, product of $\mathbf{u}$ and $\mathbf{v}$, and C is a tensor of order 3, which projects the tensor product of $\mathbf{u}$ and $\mathbf{v}$ onto the space of $\mathbf{p}$ (which is assumed to be the same space as that containing $\mathbf{u}$ and $\mathbf{v}$). Under similar simplifying assumptions to the additive model, the multiplicative model has the form:

$$\mathbf{p} = \mathbf{u} \odot \mathbf{v} \tag{8}$$

where $\odot$ represents elementwise multiplication, that is, $\mathbf{p}_i = \mathbf{u}_i \cdot \mathbf{v}_i$, where $\mathbf{p}_i$ is the ith coefficient of $\mathbf{p}$. In this simpler model, C simply picks out the diagonal in the $\mathbf{u} \otimes \mathbf{v}$ tensor.

One obvious question to consider is how well such simple operators scale when applied to phrases or sentences longer than a few words. Polajnar, Rimell, and Clark (2014) and Polajnar and Clark (2014) suggest that elementwise multiplication performs badly in this respect, with the quality of the composed representation degrading quickly as the number of composition operations increases. Vector addition is more stable, but Polajnar, Rimell, and Clark suggest that the quality of the composed representation degrades after around 10 binary composition operations, even with addition.

M&L also introduce a method that aims to interpret one of the constituent vectors as an operator, despite its lying in the same semantic space as the argument. A dilation operation decomposes **v** (the argument) into two components, one parallel to **u** and the other perpendicular, and dilates **v** in the direction of **u**:

$$\mathbf{p} = (\mathbf{u} \cdot \mathbf{u})\mathbf{v} + (\lambda - 1)(\mathbf{u} \cdot \mathbf{v})\mathbf{u} \tag{9}$$

M&L also consider the tensor product, which is an instance of the general multiplicative model described above (when C is the identity operator), and described in Section 4.4, but find that it performs poorly on their phrasal composition tasks and data sets, using classical count vectors. One of the potential problems with the tensor product is that the resulting phrase vector lives in a different vector space from the argument vectors, so that $\overrightarrow{fast} \otimes \overrightarrow{car}$, for example, lives in a different space from $\overrightarrow{car}$. Hence M&L also discuss circular convolution, a technique for projecting a vector in a tensor product space onto the original vector space components (Plate 1991).

4.2 The Categorial Framework

Because the majority of the methods used in our experiments are variants of the Categorial framework, this section provides a fairly detailed description of that framework.

In an attempt to unite a formal, Montague-style semantics (Montague 1970; Dowty, Wall, and Peters 1981) with a distributional semantics, Coecke, Sadrzadeh, and Clark (2010) and Baroni, Bernardi, and Zamparelli (2014) both make the observation that the semantics of argument-taking words such as verbs and adjectives can be represented using multi-linear algebraic objects.[6] This is the basis for the Categorial framework (Coecke, Sadrzadeh, and Clark 2010), which uses higher-order tensors—a generalization of matrices—for words which take more than one argument. The Categorial framework has vectors and their multi-linear analogues as native elements, and provides a natural operation for composition, namely, tensor contraction (see Section 4.2.2). It therefore stands in contrast to frameworks for compositional distributional semantics that effectively take the logic of formal semantics as a starting point, and use distributional semantics to enrich the logical representations (Garrette, Erk, and Mooney 2011; Lewis and Steedman 2013; Herbelot and Copestake 2015).

It is important to note that the Categorial framework makes no commitment to how the multi-linear algebraic objects are realized, only a commitment to their shape and how they are combined. In particular, the vector spaces corresponding to atomic categories, such as noun and sentence, do not have to be *distributional* in the classical sense; they could be *distributed* in the connectionist sense (Smolensky 1990), where the basis vectors themselves are not readily interpretable (the neural embeddings that we use in this article fall into this category).

4.2.1 Composition as Matrix Multiplication. A useful starting point in describing the framework is adjectival modification. In fact, the proposal in Baroni and Zamparelli (2010) (B&Z hereafter) to model the meanings of adjectives as matrices can be seen as

6 Another way to motivate this idea is that some words naturally have more of an "operator semantics," whereas others have more of a "content semantics." Socher et al. (2012, 2013) realize this distinction in the context of a recursive neural network, using matrices for the operator semantics and vectors for the content semantics. Every word has both associated types, but the network may learn to put more weight on one type or the other for a given word.

$$\underset{\overline{red}}{\begin{pmatrix} R_{11} & R_{12} & R_{13} & R_{14} & R_{15} \\ R_{21} & R_{22} & R_{23} & R_{24} & R_{25} \\ R_{31} & R_{32} & R_{33} & R_{34} & R_{35} \\ R_{41} & R_{42} & R_{43} & R_{44} & R_{45} \\ R_{51} & R_{52} & R_{53} & R_{54} & R_{55} \end{pmatrix}} \underset{\overrightarrow{car}}{\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix}} = \underset{\overrightarrow{red\ car}}{\begin{pmatrix} rc_1 \\ rc_2 \\ rc_3 \\ rc_4 \\ rc_5 \end{pmatrix}}$$

Figure 2
Matrix mulitiplication for adjective–noun combinations.

an instance of the Categorial framework (and also an instance of the general framework described in Baroni, Bernardi, and Zamparelli [2014]). The insight in B&Z is that, in theoretical linguistics, adjectives are typically thought of as having a functional role, mapping noun denotations to noun denotations. B&Z make the transition to vector spaces by arguing that, in linear algebra, functions are represented as matrices (the **linear maps**). This insight provides an obvious answer to the question of what the composition operator should be in this framework, namely, matrix multiplication. Figure 2 shows how the context vector for *car* is multiplied by the matrix for *red* to produce the vector for *red car*. In this simple example, the noun space containing $\overrightarrow{car} = (c_1, \ldots, c_5)^{\mathbf{T}}$ and $\overrightarrow{red\ car} = (rc_1, \ldots, rc_5)^{\mathbf{T}}$ has five basis vectors, which means that a 5×5 matrix, $\overline{red}$, is required for the adjective.

The second contribution of B&Z is to propose a method for learning the $\overline{red}$ matrix from supervised training data. What should the gold-standard representation for $\overrightarrow{red\ car}$ be? B&Z argue that, given large enough corpora, it should ideally be the context vector for the compound *red car*; in other words, the vector for *red car* should be built in exactly the same way as the vector for *car*. These context vectors for phrases are generally known as **holistic vectors**. The reason for the adjective matrix is that it allows the generalization of adjective–noun combinations beyond those seen in the training data. The details of the B&Z training process will not be covered in this section, but briefly, the process is to find all examples of adjective–noun pairs in a large training corpus, and then use standard linear regression techniques to obtain matrices for each adjective in the training data. These learned matrices can then be used to generalize beyond the seen adjective–noun pairs. So if *tasty artichoke*, for example, has not been seen in the training data (or perhaps seen infrequently), the prediction for the context vector $\overrightarrow{tasty\ artichoke}$ can be obtained, as long as there are enough examples of *tasty X* in the data to obtain the $\overline{tasty}$ matrix (via linear regression), and enough occurrences of *artichoke* to obtain the $\overrightarrow{artichoke}$ vector.

Testing is performed by using held-out context vectors for some of the adjective–noun pairs in the data, and seeing how close—according to the cosine measure—the predicted context vector is to the actual context vector for each adjective–noun pair in the test set. The learning method using linear regression was compared against various methods of combining the context vectors of the adjective and noun, such as vector addition and elementwise multiplication, and was found to perform significantly better at this prediction task.

4.2.2 An Extension Using Multi-Linear Algebra: Tensor-Based CCG Semantics. The grammatical formalism assumed in this section will be a variant of Categorial Grammar, which was also the grammar formalism used by Montague (1970). The original papers setting out the tensor-based compositional framework (Clark, Coecke, and Sadrzadeh 2008; Coecke, Sadrzadeh, and Clark 2010) used pregroup categorial grammars (Lambek 2008), largely because they share an abstract mathematical structure with vector spaces. However, other forms of categorial grammar can be used equally as well in practice, and here we use Combinatory Categorial Grammar (CCG; Steedman 2000), which has been shown to fit seamlessly with the tensor-based semantic framework (Grefenstette 2013; Maillard, Clark, and Grefenstette 2014). With a computational linguistics audience in mind, we also present the framework entirely in terms of multi-linear algebra, rather than the category theory of the original papers.

A matrix is a second-order tensor; for example, the $\overline{red}$ matrix mentioned earlier lives in the $\mathsf{N} \otimes \mathsf{N}$ space, meaning that two indices—each corresponding to a basis vector in the noun space N—are needed to specify an entry in the matrix.[7] Noting that $\mathsf{N} \otimes \mathsf{N}$ is structurally similar to the CCG syntactic type for an adjective (N/N)—both specify functions from nouns to nouns—a recipe for translating a syntactic type into a semantic type suggests itself: Replace all slash operators in the syntactic type with tensor product operators. With this translation, the combinators used by CCG to combine syntactic categories carry over seamlessly to the meaning spaces, maintaining what is often described as CCG's "transparent interface" between syntax and semantics. The seamless integration with CCG arises from the (somewhat trivial) observation that tensors are linear maps—a particular kind of function—and hence can be manipulated using CCG's combinatory rules.

Here are some example syntactic types, and the corresponding tensor spaces in which words with those types are semantically represented (using the notation *syntactic type* : *semantic type*). We first assume that all atomic types have meanings living in distinct vector spaces:[8]

- noun phrases, $NP : \mathsf{N}$
- sentences, $S : \mathsf{S}$

Replacing each slash in a complex syntactic type with a tensor product operator results in the following meaning spaces, following the CCG result-leftmost convention for both the syntactic and semantic types:

- Intransitive verb, $S\backslash NP : \mathsf{S} \otimes \mathsf{N}$
- Transitive verb, $(S\backslash NP)/NP : \mathsf{S} \otimes \mathsf{N} \otimes \mathsf{N}$
- Ditransitive verb, $((S\backslash NP)/NP)/NP : \mathsf{S} \otimes \mathsf{N} \otimes \mathsf{N} \otimes \mathsf{N}$
- Adverbial modifier, $(S\backslash NP)\backslash(S\backslash NP) : \mathsf{S} \otimes \mathsf{N} \otimes \mathsf{S} \otimes \mathsf{N}$
- Preposition modifying NP, $(NP\backslash NP)/NP : \mathsf{N} \otimes \mathsf{N} \otimes \mathsf{N}$

7 The tensor product $V \otimes W$ of two vector spaces $V \in \mathbb{R}^n$ and $W \in \mathbb{R}^m$ is an nm-dimensional space spanned by elements of the form $\overrightarrow{v} \otimes \overrightarrow{w}$; that is, pairs of basis vectors of V and W. An element $a \in V \otimes W$ can be written as $\sum a_{ij} \overrightarrow{v_i} \otimes \overrightarrow{w_j}$ where a_{ij} is a scalar.

8 We make no distinction in this section between nouns and noun phrases, as far as the corresponding semantic space is concerned.

Hence the meaning of an intransitive verb, for example, is a particular matrix, or second-order tensor, in the tensor product space $\mathsf{S} \otimes \mathsf{N}$. The meaning of a transitive verb is a "cuboid," or third-order tensor, in the tensor product space $\mathsf{S} \otimes \mathsf{N} \otimes \mathsf{N}$. In the same way that the syntactic type of an intransitive verb can be thought of as a function—taking an *NP* and returning an *S*—the meaning of an intransitive verb is also a function (linear map)—taking a vector in N and returning a vector in S. Another way to think of this function is that each element of the matrix specifies, for a pair of basis vectors (one from N and one from S), how a value on the given N basis vector contributes to the result for the given S basis vector.

In order to see how the tensors combine with their arguments, consider the following CCG derivation involving an intransitive verb, with the corresponding semantic type under its syntactic type:

$$\begin{array}{cc} \textit{Basil} & \textit{sleeps} \\ \hline NP & S\backslash NP \\ \mathsf{N} & \mathsf{S} \otimes \mathsf{N} \\ \hline \multicolumn{2}{c}{S} \\ \multicolumn{2}{c}{\mathsf{S}} \end{array} \; < \tag{10}$$

Let the meaning of *Basil* be $\overrightarrow{Basil} \in \mathsf{N}$ and *sleeps* be $\overline{sleeps} \in \mathsf{S} \otimes \mathsf{N}$; then applying the $\overline{sleeps}$ operator (matrix) to its argument $\overrightarrow{Basil}$ is an example of the more general **tensor contraction** operation from multi-linear algebra. We refer readers to Maillard, Clark, and Grefenstette (2014) for details of how the additional combinators of CCG—such as forward and backward composition, backward-crossed composition, and type-raising—naturally apply to the tensor-based semantic representations, and how tensor contraction applies in this more general setting.

4.2.3 Existing Implementations of the Categorial Framework. There are a few existing implementations of the Categorial framework, focusing mostly on adjectives and transitive verbs. The adjective implementation is that of B&Z as described in Section 4.2.1, where linear regression is used to learn each adjective as a mapping from noun contexts to adjective–noun contexts, with the observed context vectors for the noun and adjective–noun instances as training data. Because an adjective–noun combination has noun phrase meaning, the noun space is the obvious choice for the space in which the composed meanings should live. Maillard and Clark (2015) extend this idea by showing how the skip-gram model of Mikolov, Yih, and Zweig (2013) can be adapted to learn adjective matrices as part of a neural embedding objective.

For verb–argument composition, the question of sentence spaces arises.[9] A transitive verb such as $\overline{chase}$ lives in $S \otimes N \otimes N$, and can therefore be represented as

$$\sum_{ijk} C_{ijk}(\overrightarrow{s_i} \otimes \overrightarrow{n_j} \otimes \overrightarrow{n_k}) \tag{11}$$

9 We consider a verb with all of its argument positions saturated—for example, an SVO triple—to be a sentence, although real-world sentences are much longer and contain determiners and other function words.

for some choice of coefficients C_{ijk}. Grefenstette et al. (2011) choose $S = N \otimes N$, using the tensor product of the noun space with itself as the sentence space. Their concrete implementation learns the verb by counting observed $n^{(s)}$, $n^{(o)}$ pairs where $n^{(s)}$ is the subject and $n^{(o)}$ the object of V. The final representation for V is then

$$V = \sum_{i=1}^{N_V} \overrightarrow{n_i^{(s)}} \otimes \overrightarrow{n_i^{(o)}} \tag{12}$$

where N_V is the number of instances of V in the corpus, and $(n_i^{(s)}, n_i^{(o)})$ are the subject and object nouns for the ith instance of V. This method of learning the verb tensor is known as the **relational** method, because it captures relations between subject and object features. Composition with a particular subject and object reduces to taking the tensor product of the subject and object, and performing elementwise multiplication with the verb matrix, yielding a sentence meaning in $N \otimes N$. Sentence meanings can be compared with one another by taking the cosine of their angle in sentence space. One problem with this approach is that sentences with transitive verbs live in a different space than sentences with intransitive verbs, so sentences with different grammatical structures cannot be directly compared.

Kartsaklis, Sadrzadeh, and Pulman (2012) introduce another sentence space, by setting $S = N$. To achieve this, while training the verb in the same way as the relational method, they embed the $N \otimes N$ tensor into $N \otimes N \otimes N$ space, by copying the subject or object slices of the original tensor; for example, in the copy-object method the representation for V is

$$V = \sum_{i=1}^{N_V} \overrightarrow{n_i^{(s)}} \otimes \overrightarrow{n_i^{(o)}} \otimes \overrightarrow{n_i^{(o)}} \tag{13}$$

Representing sentence meaning in the noun space confers the advantage that words, phrases, and sentences of any grammatical structure can be compared with one another, but it is an open question whether the features in the noun space, whether distributional or not, are appropriate or adequate for representing sentence meaning.

An alternative way of learning the transitive verb tensor parameters is presented in Grefenstette et al. (2013), using a process analogous to B&Z's process for learning adjectives. Two linear regression steps are performed. In the first step, a matrix is learned representing verb–object phrases, that is, a verb that has already been paired with its object. For example, the matrix for $\overline{\textit{eat meat}}$ is learned as a mapping from corpus instances such as *dogs* and *dogs eat meat*. The full tensor for $\overline{\textit{eat}}$ is then learned with *meat* as input and the $\overline{\textit{eat meat}}$ matrix as output. Essentially, the subject mapping is learned in the first step, and the object mapping in the second.

Polajnar, Fagarasan, and Clark (2014), following Krishnamurthy and Mitchell (2013), investigate a non-distributional sentence space, the "plausibility space" described by Clark (2013, 2015). Here the sentence space is one- or two-dimensional, with sentence meaning either a real number between 0 and 1, or a probability distribution over the classes *plausible* and *implausible*. A verb tensor is learned using single-step linear regression, with the training data consisting of positive (plausible) and negative (implausible) SVO examples. Positive examples are attested in the corpus, and negative examples for a given verb have frequency-matched random nouns substituted in the argument positions. One tensor investigated has dimensions $K \times K \times S$, where K is the number of noun dimensions and S has two dimensions, each ranging from 0 to 1. The parameters of the tensor are learned so that when combined with a subject and object, a plausibility judgment is produced. Polajnar, Rimell, and Clark (2015) investigate a distributional sentence space based on the wider discourse context, in which the meaning of a sentence is represented as a distributional vector based on context words in the surrounding discourse, making the sentence space completely distinct from the noun space.[10]

Existing implementations of the Categorial framework have investigated learning up to third-order tensors (for transitive verbs). However, in practice, syntactic categories such as $((N/N)/(N/N))/((N/N)/(N/N))$ are not uncommon in the wide-coverage CCG grammar of Hockenmaier and Steedman (2007); such a category would require an *eighth-order* tensor. The combination of many word–category pairs and higher-order tensors results in a huge number of parameters to be learned in any implementation. As a solution to this problem, various ways of reducing the number of parameters are being investigated, for example, using tensor decomposition techniques (Kolda and Bader 2009; Fried, Polajnar, and Clark 2015), and removing some of the interactions encoded in the tensor by using only matrices to encode each predicate–argument combination (Paperno, Pham, and Baroni 2014; Polajnar, Fagarasan, and Clark 2014).

For this article, the problem of large numbers of parameters arises in the modeling of the relative pronoun, because its CCG type is $(NP\backslash NP)/(S\backslash NP)$ for subject relative clauses or $(NP\backslash NP)/(S/NP)$ for object relative clauses, resulting in a fourth-order tensor, $N \otimes N \otimes S \otimes N$. Given the limited amount of training data available (see Section 5.5.1), we do not attempt to model the full tensor, but investigate two approximations, based on the methods of Paperno, Pham, and Baroni (2014). First, by modeling the transitive verb with two matrices—one for the subject interaction and the other for the object—we are able to model the verb as a pair of matrices with semantic type $S \otimes N$. This approximation is described in Section 5.2.2, and allows us to reduce the relative pronoun to a third-order tensor, $N \otimes N \otimes S$ (Section 5.5.1). Second, we apply the same "decoupling" approximation to the relative pronoun, in order to represent the relative pronoun as a pair of matrices (Section 5.5.4), which can be learned independently of one another.

4.3 Relative Clause Composition

Two approaches to modeling relative clauses have been proposed in the compositional distributional semantics literature, both within the Categorial framework, although neither one has received a large-scale implementation. Both approaches are based on the fact that the relative clause (interpreted strictly, without the head noun) is a noun modifier, so the relative pronoun must map the meaning of the composed verb–argument phrase to a modifier of the head noun. In set theoretic terms, the relative pronoun signals the intersection between the set of individuals denoted by the head noun, and those denoted by the verb–argument phrase. For example, in *accuracy: quality that correction improves*, accuracy is both a quality and a thing that is improved by correction.

Clark, Coecke, and Sadrzadeh (2013) and Sadrzadeh, Clark, and Coecke (2013) propose an interpretation of relative clauses based on Frobenius algebra. The transitive verb is represented by a third-order tensor that combines with its argument by tensor

10 Kiros et al. (2015) investigate a similar sentence space within the context of recurrent neural networks.

contraction. The relative pronoun introduces a Frobenius operator that "discards" the third dimension of the verb tensor, allowing information to flow from the composed verb and argument to the head noun. In practice, this operator allows the verb to be represented as a matrix. The relative clause therefore takes the form of a vector, which is then composed with the head noun ($\overrightarrow{n}$) by elementwise multiplication:[11]

$$\overrightarrow{n} \odot (\overline{\text{V}}\ \overrightarrow{o}) \qquad \text{(subject relative clause, with verb V and object } o\text{)} \tag{14}$$

$$\overrightarrow{n} \odot (\overline{\text{V}}^T\ \overrightarrow{s}) \qquad \text{(object relative clause, with verb V and subject } s\text{)} \tag{15}$$

This method corresponds to an intuition that elementwise multiplication represents intersection, emphasizing those features that are shared by the head noun and the composed verb–argument phrase. As described in Section 2, this method has been tested on a toy data set by Sadrzadeh, Clark, and Coecke (2013), using relational verb matrices; we test it on RELPRON in this article.

Baroni, Bernardi, and Zamparelli (2014) propose a method for learning a relative pronoun as a fourth-order tensor that maps verb–argument phrases to noun modifiers. The proposed method would use multi-step linear regression to learn: (1) in the first step, verb–argument matrices, such as a *chase cats* matrix from training data such as ⟨*dogs, dogs chase cats*⟩; (2) in the second step, noun modifier matrices, such as a *that chase cats* matrix from training data such as ⟨*animal, animal that chases cats*⟩; and (3) in the third step, a tensor for the relative pronoun *that*, using the matrices from steps 1 and 2 with matched predicates. The final tensor would thus be learned from paired matrices such as *chases cats, that chases cats*. Under this approach, the intersective semantics of the relative pronoun must be captured within the tensor. Although the necessary training data might be relatively sparse, particularly for holistic relative clause vectors, Baroni, Bernardi, and Zamparelli (2014) hypothesize that they are sufficiently common to learn a general representation for the relative pronoun. We test a similar approach in this article, using holistic relative clause vectors to learn a tensor representing the relative pronoun (Section 5.5.1), though with single-step linear regression and a simplification of the verb representation that allows the tensor to be third- rather than fourth-order. We also learn a variant that represents the relative pronoun as a pair of matrices (Section 5.5.4) and requires learning fewer parameters.

4.4 Historical Perspective

The general question of how to combine vector-based meaning representations in a compositional framework is a relatively new question for computational linguistics, but one that has been actively researched in cognitive science since the 1980s. The question arose because of the perceived failure of distributed models in general, and connectionist models in particular, to provide a suitable account of compositionality in language (Fodor and Pylyshyn 1988). The question is also relevant to the broad enterprise of

11 In practice, the formulae work out equivalently to the copy-subject and copy-object methods used by Kartsaklis, Sadrzadeh, and Pulman (2012) for composed SVO triples, although they are arrived at in a different way. The formulation for the object relative clause also glosses over the fact that the CCG derivation requires type-raising and function composition, but it can easily be shown that the resulting semantic interpretation is equivalent based on the type-raising implementation in Maillard, Clark, and Grefenstette (2014).

artificial intelligence (AI) in that connectionist or distributed representations may seem to be in opposition to the more traditional symbolic systems often used in AI.

Smolensky argues for the tensor product as the operation that binds a predicate to an argument in the distributed meaning space (Smolensky 1990; Smolensky and Legendre 2006). Smolensky represents a structure as possessing a set of *roles*, $\mathbf{r}_i$, which, for a particular instance of the structure, also has a set of *fillers*, $\mathbf{f}_i$ (the roles are *bound* to fillers). The vector representation of a role–filler pair is obtained using the tensor product, and the representation for a complete structure, $\mathbf{s}$, of role–filler pairs is the sum of the tensor products for each pair:

$$\mathbf{s} = \sum_i \mathbf{f}_i \otimes \mathbf{r}_i \tag{16}$$

A dependency tree, for example, is naturally represented this way, with the roles being predicates paired with grammatical relations, such as *object of eat*, and the fillers being heads of arguments, such as *hotdog*. The vector representation of *eat hotdog* is $\overrightarrow{hotdog} \otimes \overrightarrow{eat\_\mathrm{dobj}}$. The vector representation of a whole dependency tree is the sum of the tensor products over all the dependency edges in the tree.

Smolensky and Legendre (2006) argue at length for why the tensor product is appropriate for combining connectionist and symbolic structures. A number of other proposals for realizing symbolic structures in vector representations are described, including Plate (2000) and Pollack (1990). Smolensky's claim is that, despite perhaps appearances to the contrary, all are special cases of a generalized tensor product representation.

Clark and Pulman (2007) suggest that a similar scheme to Smolensky's could be applied to parse trees for NLP applications, perhaps with the use of a convolution kernel (Haussler 1999) to calculate similarity between two parse trees with distributed representations at each node. A similar idea has been fleshed out in Zanzotto, Ferrone, and Baroni (2015), showing how kernel functions can be defined for a variety of composition operations (including most of the operations discussed in this article), allowing the similarity between two parse trees to be computed based on the distributional representations at the leaves and the composed representations on the internal nodes.

There are similarities between Smolensky's approach and the Categorial framework from Section 4.2, in that tensor representations are central to both. However, in Smolensky's approach, the tensor product is the operation used for composition of the predicate and its argument, whereas tensor contraction is the composition operation in the Categorial framework.

5. Composition Methods: Implementation

This section provides a more detailed description of the methods we have tested on RELPRON, ranging from some trivial lexical baselines, through to the simple arithmetic operators, and finally more sophisticated methods based on the Categorial framework. Both count-based vectors and neural embeddings have been investigated.

5.1 Word Vectors

In this section we describe how we built the word and holistic phrase vectors used by the various composition methods.

5.1.1 Count-Based Vectors (Count). For historical completeness we include classical count-based vectors in this article, although these large vectors are impractical for more sophisticated tensor learning. These vectors are used only for the baseline results (Section 6, Table 5), including the Frobenius algebra composition method of Sadrzadeh, Clark, and Coecke (2013).

Count-based word vectors were built from a 2013 download of Wikipedia, using the settings from Grefenstette and Sadrzadeh (2011). The top 2,000 words in the corpus (excluding stopwords) were used as features, with a co-occurrence window of three content words on either side of the target word (i.e., with stopwords removed before applying the window), and co-occurrence counts weighted as the probability of a context word given the target word divided by the overall probability of the context word (Mitchell and Lapata 2008). We found that these settings achieved the best result on the transitive verb composition task of Grefenstette and Sadrzadeh (2011).

5.1.2 Reduced Count-Based Vectors (Count-SVD). As a variant on the classical count-based vectors, we also present baseline results (Section 6, Table 5) on a count-based space reduced with Singular Value Decomposition (SVD), which produces dense vectors more comparable to neural embeddings.

Word vectors were built from the same 2013 Wikipedia download, using the settings of Polajnar, Fagarasan, and Clark (2014). The top 10,000 words in the corpus (excluding stopwords) were used as features, with sentence boundaries defining the co-occurrence window, and co-occurrence counts weighted with the t-test statistic (Curran 2004). SVD was used to reduce the number of dimensions to 300. Following Polajnar and Clark (2014), context selection (with $n = 140$) and row normalization were performed prior to SVD.

5.1.3 Neural Embeddings (Skip-Gram). Our main results in Section 6 are produced using neural embeddings. These types of vectors have proven useful for a wide variety of tasks in recent literature, and in our early experiments with tensor, learning provided the best results on the RELPRON development data and other composition tasks. For our methods, in addition to the word vectors, we required holistic vectors to use as targets for training the verb and relative pronoun matrices and tensors. The holistic vectors were also obtained using neural embeddings.

Word vectors were built using a re-implementation of skip-gram with negative sampling (Mikolov et al. 2013) on a lemmatized 2015 download of Wikpedia. We used a window of ten words on either side of the target, with ten negative samples per word occurrence, and 100-dimensional target and context vectors. All lemmas occurring fewer than 100 times in the corpus were ignored.[12] The context vectors produced during word vector learning were retained for use during training of the holistic vectors for phrases.

Verb matrices and relative pronoun matrices and tensors require holistic vectors to serve as training targets for linear regression. These vectors represent the observed contexts in which instances of verbs or relative pronouns with their arguments are found in the Wikipedia corpus. We tagged and parsed the corpus with the Stanford Parser (Chen and Manning 2014). For learning of verb matrices, we extracted verb–subject and verb–object pairs, ensuring that each pair occurred within a transitive verb

12 The word *slipping*, which occurs once in the RELPRON test set, was missing from the resulting corpus. We substituted the vector for *slip* at test time.

construction. For learning of relative pronoun matrices and tensors, we extracted ⟨head noun, verb, argument⟩ tuples that occurred within relative clause constructions. To identify relative clauses, we looked for characteristic dependency subtrees containing the Stanford Universal Dependency labels RELCL and PRONTYPE=REL. The grammatical role of the relative pronoun argument of the verb, and of the verb's in situ argument, determined whether it was a subject or object relative clause. We used a small number of manually defined heuristics to help limit the results to true relative clauses, for example, to exclude cases of clausal complementation on the verb, as in *an incident that left one government embarrassed*.

The contexts of our extracted verb–argument pairs and relative clause tuples were used to train holistic vectors using our implementation of skip-gram, only retaining phrases that occurred at least twice in the corpus. For holistic vectors, we followed Paperno, Pham, and Baroni (2014) and used a wider window of 15 words on either side of the target. Skip-gram has a random component in the initialization of vectors. In order to ensure that the learned holistic and word vectors were in the same vector space, we trained both using the same context vectors.

5.2 Verb Matrices

In this section we describe two methods used for constructing verb matrices for the composition methods based on the Categorial framework.

5.2.1 Relational Verb Matrices. Using the relational method of Grefenstette and Sadrzadeh (2011), as described in Section 4.2.3, verb matrices were built as a sum of outer products of their subject–object pairs. Pairs were subject to a minimum per-verb count of 2 and a minimum frequency of 100 for both nouns in the 2013 Wikipedia download, and not weighted by frequency.

5.2.2 Decoupled Verb Matrices with Linear Regression. All of the Categorial methods require a representation for the transitive verb in the relative clause. Based on the CCG category $(S\backslash NP)/NP$ of a transitive verb, this is a third-order tensor, $S \otimes N \otimes N$. The relative pronoun has a CCG type of $(NP\backslash NP)/(S\backslash NP)$ or $(NP\backslash NP)/(S/NP)$, resulting in a fourth-order tensor, $N \otimes N \otimes S \otimes N$. In order to make the learning of a relative pronoun tensor tractable, we make use of the method introduced by Paperno, Pham, and Baroni (2014) and Polajnar, Fagarasan, and Clark (2014) in order to reduce the semantic type of the transitive verb to $S \otimes N$. This method models a transitive verb as two second-order tensors (matrices), one governing the interaction of the verb with its subject and the other with its object, and has been shown to reproduce sentence semantics without loss of accuracy. This "decoupling" approximation for the transitive verb allows the relative pronoun to be modeled as a third-order tensor, $N \otimes N \otimes S$.

Specifically, we adopt the approach of Paperno, Pham, and Baroni (2014). For each verb V we learn a pair of matrices $\overline{V}^{S}, \overline{V}^{O}$. These matrices are used by Paperno, Pham, and Baroni to calculate the vector for an SVO triple (s, V, o) as in Equation (17), which also gives a lexicalized example.

$$\overline{V}^{S}\vec{s} + \overline{V}^{O}\vec{o} + \vec{v} \qquad ex.\ \overline{\text{eat}}^{S}\overrightarrow{\text{cat}} + \overline{\text{eat}}^{O}\overrightarrow{\text{mouse}} + \overrightarrow{\text{eat}} \tag{17}$$

We also adopt the modification of Gupta, Utt, and Padó (2015), in which the word vector for the verb is not added to the composition result, because this formulation was found to yield more accurate results. With this modification, we calculate the vector for an SVO triple (s, V, o) as in Equation (18).

$$\overline{V}^{S}\overrightarrow{s} + \overline{V}^{O}\overrightarrow{o} \qquad ex.\ \overline{\text{eat}}^{S}\overrightarrow{\text{cat}} + \overline{\text{eat}}^{O}\overrightarrow{\text{mouse}} \tag{18}$$

Verb matrices were learned using L_2 ridge regression (Hoerl and Kennard 1970), also known as L_2-regularized regression. For a given verb V, the training data for $\overline{V}^{S}$ consisted of (subject vector, holistic subject–verb vector) pairs, and the training data for $\overline{V}^{O}$ consisted of (object vector, holistic verb–object vector) pairs. In addition, each pair was weighted by the logarithm of its number of occurrences in the corpus, which we found improved performance on the RELPRON development set. Regression was implemented in Python using standard NumPy operations, and we optimized the regularization parameter on the RELPRON development set, using MAP scores and the SPLF composition method described in Section 5.5.3. A single regularization parameter was used globally for all verbs and set to 75 after tuning.

5.3 Simple Composition Methods

In this section we describe our relative clause composition methods. We first present the baseline methods, consisting of lexical comparisons, simple arithmetic operators, and the Frobenius algebra method of Clark, Coecke, and Sadrzadeh (2013) and Sadrzadeh, Clark, and Coecke (2013). We then introduce our main methods, all of which, like the Frobenius algebra method, are based on the Categorial framework, but which differ according to whether and how the relative pronoun is modeled. All of our main methods are implemented using neural embedding vectors.

5.3.1 Lexical Baselines. Two simple lexical similarity baselines (**Lexical**) set the vector representation of the property equal to the verb vector or the argument vector. No composition is involved. These baselines check whether a system can predict the similarity between, for example, *traveler* and *person that hotel serves* by the similarity between *traveler* and *hotel* (argument) or *traveler* and *serve* (verb). We do not have a lexical baseline consisting of the head noun, because this would not produce a well-defined ranking of properties, given that many properties share the same head noun (*person* in this example).

5.3.2 Arithmetic Operators. Arithmetic composition methods (**Arithmetic**) are the vector addition and elementwise vector product (Mitchell and Lapata 2008, 2010) of the vectors for the three lexical items in the property: head noun, verb, and argument. For vector addition, we also perform ablation to determine the contribution of each lexical item. The relative pronoun itself is not modeled in the composed representation for the arithmetic operators. Although the arithmetic operators are simple, vector addition has proven a difficult baseline to beat in many composition tasks.

5.4 Frobenius Algebra Baseline

The Frobenius algebra method of Clark, Coecke, and Sadrzadeh (2013) and Sadrzadeh, Clark, and Coecke (2013) is a previously existing implementation of the Categorial framework for relative clauses, and was described in detail in Section 4.3. The composition of the relative clause vector is shown in Equations (19)–(20), for subject and object relative clauses, respectively, where $\odot$ represents elementwise multiplication; note that unlike the other Categorial methods, there is a single verb matrix, which is learned as a relational matrix.

$$\vec{n} \odot (\overline{\mathrm{V}}\ \vec{o}) \qquad ex.\ \overrightarrow{\text{device}} \odot (\overline{\text{detect}}\ \overrightarrow{\text{planet}}) \qquad \text{(subject)} \tag{19}$$

$$\vec{n} \odot (\overline{\mathrm{V}}^T\ \vec{s}) \qquad ex.\ \overrightarrow{\text{device}} \odot (\overline{\text{play}}^T\ \overrightarrow{\text{shepherd}}) \qquad \text{(object)} \tag{20}$$

5.5 Composition Methods within the Categorial Framework

In the next sections we describe the main composition methods evaluated in this article. These methods are based on the Categorial framework, as is the Frobenius algebra baseline; however, the methods described here are all implemented with Skip-Gram vectors, and matrices and tensors learned by linear regression. Figure 3 gives a schematic view of these composition methods.

5.5.1 Tensor Composition Method (RPTensor). The **RPTensor** method models the relative pronoun as a third-order tensor $\overline{\overline{R}}$. Learning a relative pronoun tensor from holistic

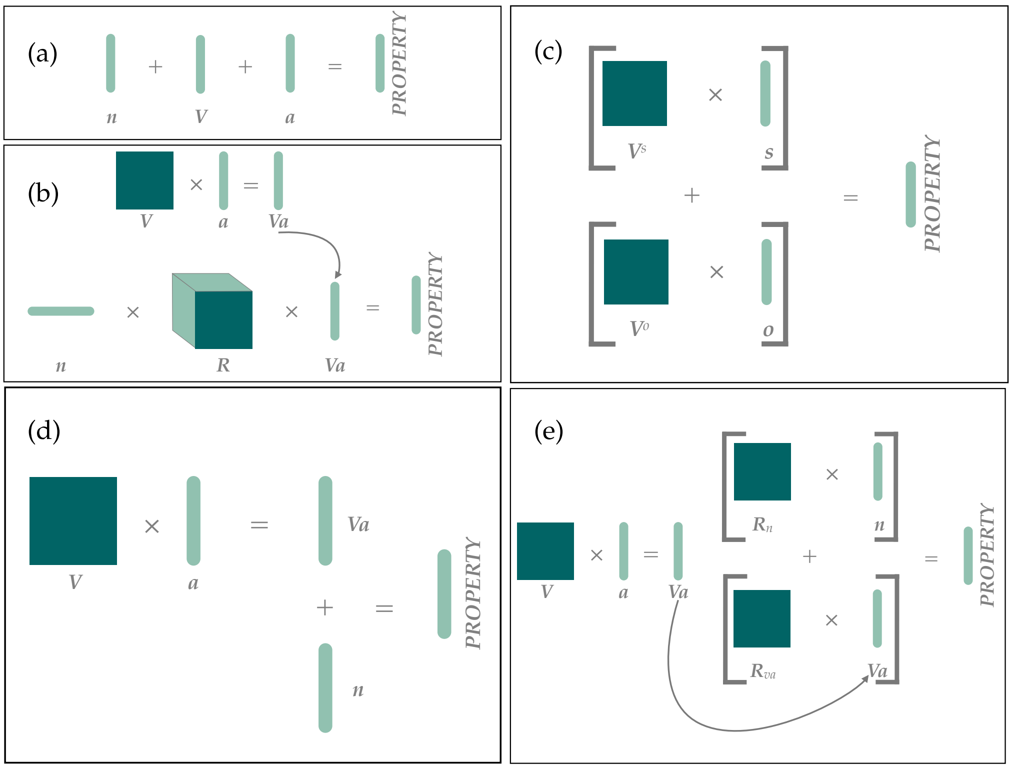

Figure 3
Schematic view of the major composition methods investigated in this article: (a) Addition, (b) RPTensor, (c) PLF, (d) SPLF, (e) FPLF. Figures are agnostic as to subject or object grammatical function for the relative clause.

vectors for relative clauses was suggested by Baroni, Bernardi, and Zamparelli (2014), but to our knowledge this is the first implementation of a function word tensor. We learn two separate tensors $\overline{\overline{R}}^{\mathrm{S}}$ and $\overline{\overline{R}}^{\mathrm{O}}$ for subject and object relative clauses, respectively, because we assume that the relative pronoun may have a different semantics in each case.

The relative pronoun tensors have the type $N \otimes N \otimes S$, and combine via tensor contraction with the head noun vector and the vector resulting from verb–argument composition. Here and in subsequent sections, $\overrightarrow{n}$ is the head noun vector, and $\overrightarrow{a}$ is the argument vector, which may be the object (in a subject relative clause) or the subject (in an object relative clause). The composition of the relative clause vector is shown in Equations (21)–(22), for subject and object relative clauses, respectively.

$$\overrightarrow{n}\ \overline{\overline{R}}^{\mathrm{S}}\ (\overline{V}^{\mathrm{O}}\ \overrightarrow{a}) \qquad ex.\ \overrightarrow{\text{device}}\ \overline{\overline{\text{that}}}^{\mathrm{S}}\ (\overline{\text{detect}}^{\mathrm{O}}\ \overrightarrow{\text{planet}}) \qquad \text{(subject)} \tag{21}$$

$$\overrightarrow{n}\ \overline{\overline{R}}^{\mathrm{O}}\ (\overline{V}^{\mathrm{S}}\ \overrightarrow{a}) \qquad ex.\ \overrightarrow{\text{device}}\ \overline{\overline{\text{that}}}^{\mathrm{O}}\ (\overline{\text{play}}^{\mathrm{S}}\ \overrightarrow{\text{shepherd}}) \qquad \text{(object)} \tag{22}$$

Note that the appropriate verb matrix, subject or object, is first contracted with its argument, so that the relative pronoun is composing the head noun with the verb–argument phrase (a vector).

As with the verb matrices, the relative pronoun tensors $\overline{\overline{R}}^{\mathrm{S}}$ and $\overline{\overline{R}}^{\mathrm{O}}$ are learned by weighted L_2 ridge regression, where the training data consist of (head noun vector, holistic verb–argument vector, holistic relative clause vector) tuples. The regularization parameter was optimized on the RELPRON development set using MAP scores and set to 80 for both tensors.

The power of the tensor comes from the fact that it captures the interaction between the head noun and the composed verb–argument phrase, and hence models the relative pronoun in a meaningful way that composition methods such as vector addition, which omits the pronoun entirely, do not. However, capturing these interactions leads to a large number of parameters, and the sparsity of training data is therefore a challenge in training the full tensor model. Our corpus of 72M sentences contained only around 800K occurrences of relative clauses, compared with over 80M verb–subject and verb–object occurrences (4M types); and this was heavily unbalanced across subject and object relative clauses, with 771K subject relative clause occurrences (222K types) compared with 21K object relative clause occurrences (only 8K types).

5.5.2 PLF Composition Method (PLF). The **PLF** method implements the Practical Lexical Function model of Paperno, Pham, and Baroni (2014). As described in Section 5.2.2, this is one of the models that decouples the subject and object interaction of the transitive verb. In PLF, function words such as relative pronouns are explicitly treated as empty elements. A noun phrase containing a relative clause is not modeled as a noun phrase, but rather an SVO sentence. For example, *device that detects planets* is equivalent to *device detects planets*, and *device that shepherd plays* is equivalent to *shepherd plays device*. The subject and object verb matrices combine with their arguments by tensor contraction (in this case, matrix multiplication), and the resulting vectors are added. The composition of the relative clause vector is shown in Equations (23)–(24), for subject and object relative clauses, respectively.

$$\overline{V}^{\mathrm{S}}\vec{n} + \overline{V}^{\mathrm{O}}\vec{a} \qquad ex.\ \overline{\text{detect}}^{\mathrm{S}}\,\overrightarrow{\text{device}} + \overline{\text{detect}}^{\mathrm{O}}\,\overrightarrow{\text{planet}} \qquad \text{(subject)} \quad (23)$$

$$\overline{V}^{\mathrm{S}}\vec{a} + \overline{V}^{\mathrm{O}}\vec{n} \qquad ex.\ \overline{\text{play}}^{\mathrm{S}}\,\overrightarrow{\text{shepherd}} + \overline{\text{play}}^{\mathrm{O}}\,\overrightarrow{\text{device}} \qquad \text{(object)} \quad (24)$$

As noted in Section 5.2.2, in this and all other PLF variants we use the modification of Gupta, Utt, and Padó (2015) to the original PLF, so the word vector for the verb is not included in the composition.

5.5.3 Simplified PLF Composition Method (SPLF). The **SPLF** method is a simplification of PLF, where the verb and argument are combined by tensor contraction as in PLF, but the resulting verb–argument vector is combined with the head noun by vector addition. The relative pronoun is not explicitly modeled, although we might think of the vector addition as a crude representation of relative pronoun semantics that "distinguishes" the roles played by the extracted head noun and the in situ argument. SPLF is also similar to Addition, except that the verb and argument are combined by tensor contraction (in this case, matrix multiplication). The composition of the relative clause vector is shown in Equations (25)–(26), for subject and object relative clauses, respectively.

$$\vec{n} + \overline{V}^{\mathrm{O}}\vec{a} \qquad ex.\ \overrightarrow{\text{device}} + \overline{\text{detect}}^{\mathrm{O}}\,\overrightarrow{\text{planet}} \qquad \text{(subject)} \quad (25)$$

$$\vec{n} + \overline{V}^{\mathrm{S}}\vec{a} \qquad ex.\ \overrightarrow{\text{device}} + \overline{\text{play}}^{\mathrm{S}}\,\overrightarrow{\text{shepherd}} \qquad \text{(object)} \quad (26)$$

5.5.4 Full PLF Composition Method (FPLF). The **FPLF** method offers a way to mitigate the sparsity of the training data for RPTensor, while retaining explicit modeling of the relative pronoun. Here we extend the intuition about modeling higher order tensors with matries to the relative pronoun itself, by decoupling the interaction of the pronoun with the head noun from its interaction with the composed verb–argument phrase. We had already reduced the relative pronoun to a third-order tensor, and we now represent it by two matrices, one for head noun interaction and one for verb–argument interaction. We further separate subject and object relative clauses, and thus end up with two pairs of matrices ($\overline{R}^{\mathrm{S}}_{\mathrm{n}}, \overline{R}^{\mathrm{S}}_{\mathrm{va}}$) and ($\overline{R}^{\mathrm{O}}_{\mathrm{n}}, \overline{R}^{\mathrm{O}}_{\mathrm{va}}$). We call this a "full" variant of PLF because it extends the PLF intuition to function words, a suggestion made by Paperno, Pham, and Baroni (2014) but which to our knowledge has not been previously implemented. The composition of the relative clause vector is shown in Equations (27)–(28), for subject and object relative clauses, respectively.

$$\overline{R}^{\mathrm{S}}_{\mathrm{n}}\vec{n} + \overline{R}^{\mathrm{S}}_{\mathrm{va}}(\overline{V}^{\mathrm{O}}\vec{a}) \qquad ex.\ \overline{\text{that}}^{\mathrm{S}}_{\mathrm{n}}\,\overrightarrow{\text{device}} + \overline{\text{that}}^{\mathrm{S}}_{\mathrm{va}}\,(\overline{\text{detect}}^{\mathrm{O}}\,\overrightarrow{\text{planet}}) \qquad \text{(subject)} \quad (27)$$

$$\overline{R}^{\mathrm{O}}_{\mathrm{n}}\vec{n} + \overline{R}^{\mathrm{O}}_{\mathrm{va}}(\overline{V}^{\mathrm{S}}\vec{a}) \qquad ex.\ \overline{\text{that}}^{\mathrm{O}}_{\mathrm{n}}\,\overrightarrow{\text{device}} + \overline{\text{that}}^{\mathrm{O}}_{\mathrm{va}}\,(\overline{\text{play}}^{\mathrm{S}}\,\overrightarrow{\text{shepherd}}) \qquad \text{(object)} \quad (28)$$

The relative pronoun matrices ($\overline{R}^{\mathrm{S}}_{\mathrm{n}}, \overline{R}^{\mathrm{S}}_{\mathrm{va}}$) and ($\overline{R}^{\mathrm{O}}_{\mathrm{n}}, \overline{R}^{\mathrm{O}}_{\mathrm{va}}$) are learned by weighted L_2 ridge regression, where the training data consists of (noun vector, holistic relative clause vector) pairs for ($\overline{R}^{\mathrm{S}}_{\mathrm{n}}, \overline{R}^{\mathrm{O}}_{\mathrm{n}}$), and (holistic verb–argument vector, holistic relative clause vector) pairs for ($\overline{R}^{\mathrm{S}}_{\mathrm{va}}, \overline{R}^{\mathrm{O}}_{\mathrm{va}}$). The regularization parameter was optimized on the RELPRON development set and set to 75 for all four matrices.

5.5.5 Categorial Baselines. We also investigate two Categorial baselines. **VArg** is the verb matrix (subject or object, as appropriate) multiplied by the in situ argument, without the head noun. For example, *person that rescuer assists* is *rescuer assists*, and *device that detects planets* is *detects planets*. The composition of the relative clause vector is shown in Equations (29)–(30), for subject and object relative clauses, respectively.

$$\overline{V}^{\mathrm{O}}\,\overrightarrow{a} \qquad ex.\ \overline{\text{detect}}^{\mathrm{O}}\,\overrightarrow{\text{planet}} \qquad \text{(subject)} \tag{29}$$

$$\overline{V}^{\mathrm{S}}\,\overrightarrow{a} \qquad ex.\ \overline{\text{play}}^{\mathrm{S}}\,\overrightarrow{\text{shepherd}} \qquad \text{(object)} \tag{30}$$

Vhn is the verb matrix (subject or object, as appropriate) multipled by the head noun. In this case, *person that rescuer assists* is *assists person*, and *device that detects planets* is *device detects*. The composition of the relative clause vector is shown in Equations (31–32), for subject and object relative clauses, respectively.

$$\overline{V}^{\mathrm{S}}\,\overrightarrow{n} \qquad ex.\ \overline{\text{detect}}^{\mathrm{S}}\,\overrightarrow{\text{device}} \qquad \text{(subject)} \tag{31}$$

$$\overline{V}^{\mathrm{O}}\,\overrightarrow{n} \qquad ex.\ \overline{\text{play}}^{\mathrm{O}}\,\overrightarrow{\text{device}} \qquad \text{(object)} \tag{32}$$

6. Evaluation and Results

This section describes the evaluation of the ranking task on RELPRON and presents the results of the various composition methods.

6.1 Evaluation

To evaluate a composition method, composed vectors are produced for all properties in the data set. For each term, the properties are ranked according to cosine similarity with the term vector. Evaluation is based on MAP, which is defined as:

$$\mathrm{MAP} = \frac{1}{N}\sum_{i=1}^{N} AP(t_i) \tag{33}$$

where N is the number of terms in the data set, and $AP(t)$ is the Average Precision (AP) for term t, defined as:

$$AP(t) = \frac{1}{P_t}\sum_{k=1}^{M} \mathrm{Prec}(k) \times \mathrm{rel}(k) \tag{34}$$

where P_t is the number of correct properties for term t in the data set, M is the total number of properties in the data set, $\mathrm{Prec}(k)$ is the precision at rank k, and $\mathrm{rel}(k)$ is an indicator function equal to one if the property at rank k is a correct property for t, and zero otherwise.

6.2 Results

Table 5 shows the results for the baseline methods, using Count, Count-SVD, and Skip-Gram vectors. The highest MAP score for all three types of vectors is achieved with vector addition, adding the vectors for head noun, verb, and argument, and the Skip-Gram vectors achieve the best result overall, with a MAP of 0.496. This is already a challenging score to beat, because it represents a system that on average places a correct property at every second position in the top-ranked properties for each term. Elementwise multiplication performs almost as well for Count vectors, but very poorly for Count-SVD and Skip-Gram vectors, which is consistent with previous findings on elementwise multiplication in dense vector spaces (Vecchi, Baroni, and Zamparelli 2011; Utsumi 2012; Milajevs et al. 2014; Polajnar and Clark 2014). We will therefore use vector addition as our primary arithmetic vector operation for the rest of the article. Moreover, we will use Skip-Gram vectors for our main Categorial framework-based experiments.

Turning to the lexical baselines and the ablation tests for vector addition, we may assess the relative contribution to the vector addition result of the three lexical items in the relative clause. We look first at the lexical baselines. Similarity between the term and argument achieves a MAP of 0.347 for Skip-Gram vectors, although similarity between the term and the verb produces a much worse ranking. Looking at the arithmetic operators, the ablation tests show that the head noun, verb, and argument all contribute to the correct ranking of properties. However, the argument is the most important, as the sum of head noun and verb vectors achieves a lower MAP (0.264 for Skip-Gram vectors) than the other two combinations (0.401 and 0.450, respectively). The importance of the argument is fairly intuitive; for example, in *family: organization that claims ancestry*, the words *family* and *ancestry* are clearly the most closely related. However, Table 5 shows that the argument alone is insufficient to rank properties and a method must perform some composition to achieve a respectable MAP score.

In Table 5 we also see that the Frobenius method, which uses a relational matrix for the verb, is not competitive with vector addition. It achieves a MAP of only 0.277 using Count vectors, and in the other two vector spaces, the MAP is almost zero, presumably because of the elementwise multiplication that is part of the method. Based on this result, we do not pursue the Frobenius algebra method further.

Table 5
MAP scores of Lexical, Arithmetic, and Frobenius algebra methods on the RELPRON development set using Count, Count-SVD, and Skip-Gram vectors, and relational verb matrices.

	Method	Count	Count-SVD	Skip-Gram
Lexical	$\overrightarrow{arg}$	0.272	0.386	0.347
	$\overrightarrow{verb}$	0.138	0.179	0.176
Arithmetic	$\overrightarrow{hn} \odot \overrightarrow{arg} \odot \overrightarrow{verb}$	0.364	0.123	0.181
	$\overrightarrow{hn} + \overrightarrow{arg} + \overrightarrow{verb}$	**0.386**	**0.442**	**0.496**
	$\overrightarrow{arg} + \overrightarrow{verb}$	0.331	0.407	0.401
	$\overrightarrow{hn} + \overrightarrow{arg}$	0.339	0.425	0.450
	$\overrightarrow{hn} + \overrightarrow{verb}$	0.231	0.229	0.264
Categorial	Frobenius	0.277	0.023	0.030

Table 6
MAP scores of composition methods on the RELPRON development and test sets using Skip-Gram vectors.

Method	Development	Test
SPLF	**0.496**	**0.497**
Addition	**0.496***	**0.472***
PLF	0.444	0.433*
FPLF	0.405	0.387*
RPTensor	0.380	0.354
VArg	0.448	0.397
Vhn	0.219	0.204

* Significantly higher than next lowest result ($p < 0.05$).

Table 6 shows the results of the composition methods and Categorial baselines on the RELPRON development and test data. The relative performance of the methods is very similar across the development and test portions of the data set. In both cases, SPLF and Addition are the top two performers, with SPLF having a higher MAP than Addition on the test data, although the difference is not significant. Both methods achieve a MAP of nearly 0.50, which corresponds to putting a correct property in every second place on average.

The next highest MAP is achieved by PLF. It is instructive to compare PLF with the Categorial baselines, VArg and Vhn, since PLF sums these two components. Vhn performs very badly, in line with the sum of the verb and head noun vectors in Table 5. However, the head noun does contribute to the composition of relative clause meaning, as we can see from the fact that on the test data, PLF achieves a significantly higher MAP than VArg. It is also relevant to compare PLF with SPLF. PLF learns a verb matrix for the interaction with the head noun, whereas SPLF approximates the head noun interaction as addition. SPLF significantly outperforms PLF on the development and test data, a difference that suggests that modeling the relative clause as an SVO sentence may not be the optimal approach for relative clause composition.

FPLF and RPTensor are the two methods that learn a relative pronoun function by regression. FPLF achieves a lower MAP than SPLF, Addition, and PLF, but a higher MAP than RPTensor, supporting the hypothesis that the available training data in the form of holistic vectors for relative clauses is insufficient for the number of parameters in the full tensor method. Both methods underperform the VArg functional baseline and some of the arithmetic baselines.

7. Error Analysis

In this section we examine in detail the performance of some of the main composition methods on the RELPRON development data. We are interested in four main points. First, we investigate whether the grammatical function (subject or object) of the relative clause plays a role in the MAP score. We find that most of the methods are relatively well-balanced across grammatical functions, but FPLF shows a lower MAP on object properties, where the amount of training data is substantially lower.

Second, we investigate whether the different head nouns in the data set have any effect on MAP score. We find that the scores are relatively well-balanced across head nouns, but that more concrete terms and their properties may be easier to model. However, based on qualitative observations, a more important factor seems to be term polysemy.

Third, we look at how well the composition methods capture the intersective semantics of the relative clause construction. We break down the evaluation into two independent components: how often the methods are able to identify the correct head noun, and how often they are able to pick out correct properties when the head noun is known. We find that all methods do very well at identifying the correct head noun, but FPLF lags behind the others at picking the correct property.

Finally, we look at some common errors on the RELPRON development data. One source of errors is lexical overlap in terms and properties. We also observe that some of the systems assign high ranks to plausible but unannotated properties, and address this issue with an evaluation scenario where the property is the query and the task is to rank the correct term highest. We find that on average all the methods rank the correct term at approximately position two.

We focus in this section on four methods: SPLF and Addition, the two methods with the highest overall MAP score; FPLF, the better-performing method of the two which explicitly learn a relative pronoun representation; and VArg, as a high-performing Categorial baseline.

7.1 Accuracy by Grammatical Function

An obvious first step in understanding the RELPRON results is to break them down by grammatical function. Because subjects and objects are asymmetric in their interaction with the verb—in particular, verbs show stronger selectional preferences for objects than for subjects (Marantz 1984; Kratzer 1996, 2002)—we might expect that relative clauses with different extraction sites are not equally easy to compose. In addition, for the methods that learn an explicit representation of the relative pronoun (FPLF in this analysis), the substantial difference in the amount of training data might affect the result.

The MAP scores for each grammatical function are shown in Table 7. The first row shows the subject results, in which only the subject properties from the development data are ranked by similarity for each term. The AP is calculated for each term, and MAP is calculated as the mean AP over the set of all terms.[13] The object results are calculated analogously, although note that the MAP scores in Table 7 are not directly comparable to the scores on the full development set because the number of confounders is lower for each term.

For SPLF and Addition, Table 7 shows that the MAP scores are well balanced. For FPLF, on the other hand, the MAP scores are unbalanced: 0.570 for subject properties, in line with SPLF and Addition; and only 0.428 for object properties. This difference provides support for the hypothesis that the amount of training data available for subject vs. object relative clauses makes a difference to the resulting accuracy.[14] The term *navy* provides an an example of the differential performance between subject and object

13 Any term that has no properties with the relevant grammatical function is omitted from the calculation; for example, *accuracy* has no subject properties, so it is not included in the MAP for Subject.

14 As further support for this hypothesis, the RPTensor method (not in the table) showed the same discrepancy, with a MAP of 0.520 on subject properties, and 0.400 on object properties.

Table 7
MAP scores by grammatical function of extracted element in relative clause, on development set.

	SPLF	Addition	FPLF	VArg
Subject	0.585	0.571	0.570	0.530
Object	0.541	0.574	0.428	0.489

properties. SPLF ranks four correct subject and three correct object properties within its top ten ranked properties, whereas FPLF ranks five correct subject properties and only one correct object property within its top ten.[15]

The MAP scores for VArg are also unbalanced in favor of subject relative clauses, though the effect is not as pronounced. We attribute this difference to the strength of the selectional preference of a verb for its object compared to its subject, since verb–object combinations (for subject properties) may have greater semantic content, enabling a more accurate ranking of properties.

7.2 Accuracy by Head Noun

Breaking down the results on RELPRON by head noun—for example, *device* vs. *quality*—is another obvious place to look for differential performance in the composition methods. Table 8 shows the MAP scores by head noun on the development set. The head nouns are ordered from left to right according to the average concreteness rating of their terms, based on the ratings of Brysbaert, Warriner, and Kuperman (2014). In Table 8, the AP for each term is based on a ranking of all properties in the development set, and the MAP for a head noun is the mean AP over all its terms.[16]

Table 8 shows that the MAP scores are relatively well-balanced across head nouns for all methods, suggesting that there are no real outliers among head nouns in the ability of the methods to compose relative clauses. The general distribution of scores across head nouns for the four methods is also roughly similar. *Building* and *player* are consistently high, with *building* in particular having a MAP of at least 0.5 for all methods. These head nouns have some of the most concrete terms, which might suggest that concrete terms have properties that are easier to model, or at least that concrete terms have better-quality word vectors that make the ranking easier. This is not a clear effect, however, because the methods mostly have average-to-high MAPs for *quality* and *organization*, which have terms with low concreteness, and average-to-low MAPs for *device*, which has the most concrete terms in the development set.

A factor that appears to play a role in MAP scores is term polysemy. All of the methods score low on *document*, and SPLF in particular achieves its lowest MAP for this

15 The correct properties for SPLF are, in order, *organization that uses submarine (S), organization that blockades port (S), organization that fleet destroys (O), organization that battleship fights (O), organization that maintains blockade (S), organization that vessel serves (O), organization that defeats fleet (S).* The correct properties for FPLF are, in order, *organization that uses submarine (S), organization that blockades port (S), organization that maintains blockade (S), organization that fleet destroys (O), organization that establishes blockade (S), organization that defeats fleet (S).*

16 Unlike in Section 7.1, these MAP scores are directly comparable to those on the full development set, since the number of confounders is the same.

Table 8
MAP scores by head noun on development set. AP is calculated over all properties for each term, and mean AP calculated for each head noun. Head nouns are ordered from left to right by increasing concreteness.

	Head Noun						
	quality	organization	person	document	building	player	device
SPLF	0.490	0.500	0.490	0.393	0.592	0.595	0.462
Addition	0.464	0.485	0.394	0.463	0.632	0.546	0.512
FPLF	0.279	0.471	0.415	0.360	0.502	0.448	0.378
VArg	0.398	0.563	0.413	0.390	0.542	0.443	0.384

head noun. Several document terms (*lease, form, assignment, bond*) exhibit very low AP. The term *form* provides an example; it has an extremely low AP of 0.01. Although *form* was chosen by the annotators in its *document* sense, with correct properties including *document that parent signs* and *document that applicant completes*, other senses may be more dominant in the source corpus, confusing the similarity ranking.[17] Prior disambiguation of words in the data set, as in Kartsaklis and Sadrzadeh (2013), might improve performance.

We believe that polysemy may come into play with *device* terms as well, though the terms have high concreteness. One term with a very low AP is *watch*. Among the properties that SPLF ranks high for *watch* are *person that police hunt (killer)* and *device that views stars (telescope)*, both of which are related to different senses of *watch*.

7.3 Capturing the Semantics of Relative Clauses

A system that successfully captures the semantics of relative clauses must integrate the semantic contribution of the head noun with the contributions of the verb and argument; for example, identifying that a *saw* is a *device*, and that among devices, it is the one that *cuts wood*. In this section we break down the results on the RELPRON development set into two measures that demonstrate performance on these subtasks independently.

We look first at how often the methods are able to identify the correct head noun. We consider the top ten ranked properties for each term from the full development set, and calculate the percentage of them that have the correct head noun, regardless of whether the whole property is correct. The results are shown in Table 9, where we omit the VArg method because it does not take the head noun into account. Table 9 shows that overall, nearly eight out of the top ten properties have the correct head noun. FPLF matches the performance of the other two methods, although it is the only one of the three that incorporates the head noun by an operation other than addition.

We next look at the MAP scores when the ranking of properties for each term is restricted to properties with the correct head noun. For example, given a *building* term such as *ruin*, the methods must rank only the *building* properties. The task here is to

17 The top four properties for *form* as ranked by SPLF are: *organization that undergoes merger (division), organization that siblings form (family), organization that infantry reinforces (army), organization that restructuring creates (division).*

Table 9
Average proportion of top ten ranked properties that have the correct head noun, on development set.

	Head Noun							Overall
	quality	organization	person	document	building	player	device	
SPLF	0.87	0.71	0.69	0.68	0.86	1.00	0.83	0.79
Addition	0.84	0.75	0.60	0.64	0.88	1.00	0.78	0.77
FPLF	0.84	0.71	0.62	0.72	0.88	0.98	0.91	0.79

Table 10
MAP scores within head nouns on development set. Only the properties with the correct head noun are ranked for each term.

	Head Noun						
	quality	organization	person	document	building	player	device
SPLF	0.638	0.738	0.616	0.511	0.666	0.596	0.564
Addition	0.587	0.686	0.563	0.602	0.711	0.551	0.580
FPLF	0.421	0.683	0.579	0.466	0.569	0.467	0.424

distinguish *ruin* from *cinema, pub, observatory, house, mosque*, and so forth. The results are shown in Table 10.[18] We again omit VArg, because SPLF is equivalent to VArg when the head noun is held fixed.

Table 10 shows that SPLF and Addition perform similarly on distinguishing properties of terms that share a head noun, while FPLF lags behind. Combined with Table 9, this result shows that FPLF's overall lower MAP on the full development set is due to difficulty distinguishing terms from one another when the head noun is known, not to difficulty in identifying the correct head noun.

Comparing Tables 9 and 10 highlights cases where the methods struggle with one relative clause composition subtask or the other, an effect that appears to vary by head noun. For example, all methods demonstrate perfect or near-perfect ability to rank *player* properties at the top for *player* terms, but average-to-low ability to distinguish players from one another. This may be a feature of the topic domain, in that the activities undertaken by different kinds of sports players—*golfer, batter, pitcher, quarterback*, and so forth—are distributionally similar. On the other hand, all methods exhibit only average ability to identify the correct head noun for *organization* terms, but relatively high ability to select the correct *organization* properties when the head noun is known. For example, the SPLF ranking for *religion* shows a large amount of confusion with

18 One question that might be asked is whether the ranking task within head nouns would be more difficult than the full ranking task, since all buildings (for example) share certain characteristics that might make their properties particularly good mutual confounders. In practice, this is difficult to measure, since the MAP scores in Table 10 are higher than the MAPs on the full development set because there are fewer confounders.

person properties, such as *person that defends rationalism (philosopher)*.[19] However, when restricted to ranking *organization* properties, as in Table 10, SPLF achieves a perfect MAP of 1.0 for *religion*.

7.4 Common Errors and Evaluation with Properties as Queries

In addition to poor performance on polysemous terms (Section 7.2), we observed two common sources of error on RELPRON. The first is lexical overlap between terms and properties, a type of confounder that we deliberately included in the data set (see Section 3.5). For example, the top two ranked properties for *batter* for all three methods are *player that walks batter (pitcher)* and *player that strikes batter (pitcher)*; and the top ranked property for *balance* for all three methods is *document that has balance (account)*. FPLF, despite its more sophisticated modeling of the relative pronoun, suffers from lexical overlap about as much as SPLF and Addition. However, we note that the RPTensor method is able to overcome this problem to a small extent, with the lexically confounding properties ranked a few positions below the top.

The second source of error is that the methods often assign high similarity to properties that are plausible descriptions for a term, but not annotated as gold positives. For example, the top-ranked property for *lease* for the RPTensor method is *document that government sells (bond)*. The highest-ranked properties for *intellectual* for SPLF include *person that promotes idea (advocate)* and *person that analyzes ontology (philosopher)*.[20] This phenomenon speaks to the difficulty of collecting distinctive properties during the manual annotation phase; future work may involve re-annotation of the data set for additional positives. At present, the effect is to artificially lower the MAP ceiling, which affects all methods equally.

To address this issue, we performed another type of evaluation, this time treating properties as queries and ranking terms by their similarity to a property. We used Mean Reciprocal Rank (MRR) as the evaluation measure, because there is only one correct term per property. MRR is given by the formula:

$$\mathrm{MRR} = \frac{1}{M}\sum_{p=1}^{M}\frac{1}{\mathrm{rank}(p)} \tag{35}$$

where M is the number of properties in the data set and rank(p) is the rank at which the correct term t is found for property p. The reasoning behind the MRR evaluation is that even if a property is applicable to more than one term, there is always a single term which is *most* similar to it. The results are shown in Table 11. For all three methods, the correct term appears on average at approximately rank 2. The MRR evaluation is also another use of RELPRON that can be exploited in future work.

19 The top five properties in the SPLF ranking are *person that defends rationalism (philosopher)*, *person that religion has (follower)*, *person that questions theology (philosopher)*, *person that teaches epistemology (philosopher)*, and *person that accepts philosophy (follower)*.

20 Some of the highly-ranked but incorrect properties are surprisingly plausible, although clearly not identifying properties for the term in question: SPLF ranks *building that sells popcorn (theater)* in second position for *pub*.

Table 11
Mean Reciprocal Rank of correct term when property is used as query, on development set.

	Head Noun							**Overall**
	quality	organization	person	document	building	player	device	
SPLF	0.531	0.615	0.559	0.525	0.535	0.506	0.458	0.539
Addition	0.589	0.591	0.461	0.613	0.692	0.535	0.575	0.579
FPLF	0.462	0.498	0.457	0.434	0.518	0.427	0.409	0.461
VArg	0.606	0.662	0.547	0.567	0.619	0.556	0.545	0.591

8. Conclusions

RELPRON was designed to test compositional distributional semantic methods on a phrase type of intermediate size and complexity: more complex than two-word combinations, and involving a function word, but less complex than full sentences. The data set is challenging, but results are promising with current techniques; a MAP score of 0.5 corresponds to every other property being ranked correctly, given a term. Again, in line with a growing body of literature, we observe that vector addition performs extremely well, but we are able to match its performance with a more complex method based on the Practical Lexical Function model.

Based on the qualitative analysis, our contention is that, in order to improve substantially on the results presented in this article, methods more linguistically sophisticated than vector addition will be required. We base this contention on a few factors. First, vector addition has little room for improvement, with potential gains limited to those that come from improving the overall quality of the vectors. It seems unlikely that vector addition can achieve a MAP much higher than its current one of around 0.5. We also note that the relative clauses used in this article are well below the ten-word length at which the quality of composed representations with vector addition has been suggested to degrade, so future data sets focusing on more complex linguistic constructions may require alternative methods.

For the more complex methods, on the other hand, there are likely to be readily achievable gains on RELPRON from increasing or changing the training data. This article represents the first large-scale implementation of methods that learn explicit categorial representations for relative pronouns, but the training data in the form of holistic vectors for relative clauses was still observed to be relatively small compared with the number of parameters learned by the methods, especially for object relative clauses. Larger corpora or different sources of data—for example, including dictionary definitions or paraphrase data sets in the training data—may improve the models.

Beyond the issue of training data, the qualitative analysis showed that the various methods were sometimes unable to integrate the semantic contributions of the head noun and the remainder of the relative clause. Vector addition will not be able to capture this relationship any better than it already does, but other learning methods (e.g., involving non-linear models) may be able to do so.

It is also possible that a learning method designed more specifically for the RELPRON ranking task would perform better, especially because many of the training examples in our existing data are non-restrictive relative clauses—although this would require an alternative data source, and RELPRON itself is too small to provide training data. We expect RELPRON and similar data sets to be important evaluation tools for future methods that combine formal and distributional semantics, and hope that the insights provided by RELPRON inspire new work focused on linguistically challenging grammatical constructions.

Data Access

The RELPRON data set is available from the University of Cambridge data repository at `http://dx.doi.org/10.17863/CAM.298`. Source code related to this article is available at `http://dx.doi.org/10.5281/zenodo.56288`.

Acknowledgments

We are grateful to Mehrnoosh Sadrzadeh for many detailed and constructive discussions on the topic of relative clause composition, and comments on earlier drafts of this article. We would like to thank Bob Coecke, Dimitri Kartsaklis, Daoud Clarke, Jeremy Reffin, Julie Weeds, David Weir, and the anonymous reviewers for their detailed and helpful comments. Laura Rimell and Stephen Clark were supported by EPSRC grant EP/I037512/1. Jean Maillard is supported by an EPSRC Doctoral Training Grant and a St. John's Scholarship. Laura Rimell, Tamara Polajnar, and Stephen Clark are supported by ERC Starting Grant DisCoTex (306920).

References

Agirre, Eneko. 2015. SemEval-2015 Task 2: Semantic Textual Similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–268, Denver, CO.

Agirre, Eneko, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 Task 6: A pilot on Semantic Textual Similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM 2012)*, pages 385–393, Montréal.

Agirre, Eneko, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and WeiWei Guo. 2013. *SEM 2013 Shared Task: Semantic Textual Similarity. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (*SEM 2013)*, pages 32–43, Atlanta, GA.

Baroni, Marco, Raffaella Bernardi, and Roberto Zamparelli. 2014. Frege in space: A program for compositional distributional semantics. *Linguistic Issues in Language Technologies*, 9:5–110.

Baroni, Marco, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 238–247, Baltimore, MD.

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

Bender, Emily M., Dan Flickinger, Stephan Oepen, and Yi Zhang. 2011. Parser evaluation over local and non-local deep dependencies in a large corpus. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 397–408, Edinburgh, UK.

Bernardi, Raffaella, Georgiana Dinu, Marco Marelli, and Marco Baroni. 2013. A relatedness benchmark to test the role of determiners in compositional distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 53–57, Sofia, Bulgaria.

Biemann, Chris and Eugenie Giesbrecht. 2011. Distributional semantics and compositionality 2011: Shared task description and results. In *Proceedings of the Workshop on Distributional Semantics and Compositionality (DiSCo 2011)*, pages 21–28, Portland, OR.

Boleda, Gemma, Marco Baroni, Nghia The Pham, and Louise McNally. 2013. Intensionality was only alleged: On adjective-noun composition in distributional semantics. In *Proceedings of the Tenth International Conference on Computational Semantics (IWCS 2013)*, pages 35–46, Potsdam.

Boleda, Gemma, Eva Maria Vecchi, Miquel Cornudella, and Louise McNally. 2012. First order vs. higher order modification in distributional semantics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1223–1233, Jeju Island.

Brysbaert, Marc, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, 46:904–911.

Burnard, Lou. 2007. The British National Corpus, version 3. Distributed by Oxford University Computing Services on behalf of the BNC Consortium.

Chen, Danqi and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha.

Cheung, Jackie C. K. and Gerald Penn. 2012. Evaluating distributional models of semantics for syntactically invariant inference. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 33–43, Avignon.

Clark, Stephen. 2013. Type-driven syntax and semantics for composing meaning vectors. In Chris Heunen, Mehrnoosh Sadrzadeh, and Edward Grefenstette, editors, *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*. Oxford University Press, pages 359–377.

Clark, Stephen. 2015. Vector space models of lexical meaning. In Shalom Lappin and Chris Fox, editors, *Handbook of Contemporary Semantic Theory - second edition*. Wiley-Blackwell, chapter 16, pages 493–522.

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

Clark, Stephen, Bob Coecke, and Mehrnoosh Sadrzadeh. 2013. The Frobenius anatomy of relative pronouns. In *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 2013)*, pages 41–51, Sofia.

Clark, Stephen and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–547.

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.

Coecke, Bob, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical foundations for a compositional distributional model of meaning. In *Linguistic Analysis 36: A Festschrift for Joachim Lambek*, pages 345–384.

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

Dowty, David R., Robert E. Wall, and Stanley Peters. 1981. *Introduction to Montague Semantics*. Dordrecht.

Fodor, Jerry and Zenon Pylyshyn. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28:3–71.

Fried, Daniel, Tamara Polajnar, and Stephen Clark. 2015. Low-rank tensors for verbs in compositional distributional semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*, pages 731–736, Beijing.

Garrette, Dan, Katrin Erk, and Raymond Mooney. 2011. Integrating logical representations with probabilistic information using Markov Logic. In *Proceedings of the Ninth International Conference on Computational Semantics (IWCS 2011)*, pages 105–114, Oxford.

Grefenstette, Edward. 2013. *Category-Theoretic Quantitative Compositional Distributional Models of Natural Language Semantics*. Ph.D. thesis, University of Oxford.

Grefenstette, Edward, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. In *Proceedings of the Tenth International Conference on Computational Semantics (IWCS 2013)*, pages 131–142, Potsdam.

Grefenstette, Edward and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1394–1404, Edinburgh.

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

Gupta, Abhijeet, Jason Utt, and Sebastian Padó. 2015. Dissecting the practical lexical function model for compositional distributional semantics. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics (*SEM 2015)*, pages 153–158, Denver, CO.

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

Herbelot, Aurelie and Ann Copestake. 2015. Lexicalised compositionality. University of Cambridge Computer Laboratory, unpublished manuscript.

Hermann, Karl Moritz, Edward Grefenstette, and Phil Blunsom. 2013. "Not not bad" is not "bad": A distributional account of negation. In *Proceedings of the ACL Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2013)*, pages 74–82, Sofia.

Hockenmaier, Julia and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Hoerl, Arthur E. and Robert W. Kennard. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67.

Jurgens, David, Mohammad Taher Pilehvar, and Roberto Navigli. 2014. SemEval-2014 Task 3: Cross-level semantic similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 17–26, Dublin.

Kartsaklis, Dimitri and Mehrnoosh Sadrzadeh. 2013. Prior disambiguation of word tensors for constructing sentence vectors. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1590–1601, Seattle, WA.

Kartsaklis, Dimitri and Mehrnoosh Sadrzadeh. 2014. A study of entanglement in a categorical framework of natural language. In *Proceedings of the 11th Workshop on Quantum Physics and Logic (QPL 2014)*, pages 249–261, Kyoto.

Kartsaklis, Dimitri, Mehrnoosh Sadrzadeh, and Stephen Pulman. 2012. A unified sentence space for categorical distributional-compositional semantics: Theory and experiments. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 549–558, Mumbai.

Kiros, Ryan, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. In *Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NIPS 2015)*, pages 3294–3302, Montréal.

Kolda, Tamara G. and Brett W. Bader. 2009. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500.

Kratzer, Angelika. 1996. Severing the external argument from its verb. In *Phrase Structure and the Lexicon*. Springer, Netherlands, pages 109–137.

Kratzer, Angelika. 2002. The event argument and the semantics of verbs. University of Massachusetts, Amherst, unpublished manuscript.

Krishnamurthy, Jayant and Tom M. Mitchell. 2013. Vector space semantic parsing: A framework for compositional vector space models. In *Proceedings of the ACL Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2013)*, pages 1–10, Sofia.

Lambek, Joachim. 2008. *From Word to Sentence. A Computational Algebraic Approach to Grammar*. Polimetrica.

Lazaridou, Angeliki, Eva Maria Vecchi, and Marco Baroni. 2013. Fish transporters and miracle homes: How compositional distributional semantics can help NP parsing. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1908–1913, Seattle, WA.

Levy, Omer, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguitics*, 3:211–225.

Levy, Omer and Yoav Goldberg. 2014. Neural word embeddings as implicit matrix factorization. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NIPS 2014)*, pages 2177–2185, Montréal.

Lewis, Mike and Mark Steedman. 2013. Combined distributional and logical semantics. *Transactions of the Association for Computational Linguistics*, 1:179–192.

Maillard, Jean and Stephen Clark. 2015. Learning adjective meanings with a tensor-based skip-gram model. In *Proceedings of the 19th Conference on Computational Natural Language Learning (CoNLL 2015)*, pages 327–331, Beijing.

Maillard, Jean, Stephen Clark, and Edward Grefenstette. 2014. A type-driven tensor-based semantics for CCG. In *Proceedings of the EACL Workshop on Type Theory and Natural Language Semantics (TTNLS 2014)*, pages 46–54, Gothenburg.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. 2008.

Introduction to Information Retrieval. Cambridge University Press.

Marantz, Alec. 1984. *On the Nature of Grammatical Relations*. MIT Press, Cambridge, MA.

Marelli, Marco, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. SemEval-2014 Task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 1–8, Dublin.

Mikolov, T., W. Yih, and G. Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 746–751, Atlanta, GA.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS 2013)*, pages 3111–3119, Lake Tahoe, NV.

Milajevs, Dmitrijs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. 2014. Evaluating neural word representations in tensor-based compositional settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 708–719, Doha.

Miller, George A. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41.

Minnen, Guido, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–223.

Mitchell, Jeff and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL 2008)*, pages 236–244, Columbus, OH.

Mitchell, Jeff and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34:1388–1439.

Montague, Richard. 1970. English as a formal language. In B. Visentini et al., editor, *Linguaggi nella Società et nella Technica*. Edizioni di Communita, Milan, pages 188–221. Reprinted in Richmond H. Thomason (ed.), 1974, *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press.

Padó, Sebastian and Yves Peirsman. 2011. Introduction. In *Proceedings of the EMNLP Workshop on Geometrical Models for Natural Language Semantics (GEMS 2011)*, page iii, Edinburgh.

Paperno, Denis, Nghia The Pham, and Marco Baroni. 2014. A practical and linguistically-motivated approach to compositional distributional semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 90–99, Baltimore, MD.

Pham, Nghia, Raffaella Bernardi, Yao Zhong Zhang, and Marco Baroni. 2013. Sentence paraphrase detection: When determiners and word order make the difference. In *Proceedings of the IWCS 2013 Towards a Formal Distributional Semantics Workshop*, pages 21–29, Potsdam.

Plate, Tony A. 1991. Holographic reduced representations: Convolution algebra for compositional distributed representations. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI 1991)*, pages 30–35, Sydney.

Plate, Tony A. 2000. Analogy retrieval and processing with distributed vector representations. *Expert Systems*, 17(1):29–40.

Polajnar, Tamara and Stephen Clark. 2014. Improving distributional semantic vectors through context selection and normalisation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, pages 230–238, Gothenburg.

Polajnar, Tamara, Luana Fagarasan, and Stephen Clark. 2014. Reducing dimensions of tensors in type-driven distributional semantics. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1036–1046, Doha.

Polajnar, Tamara, Laura Rimell, and Stephen Clark. 2014. Evaluation of simple distributional compositional operations on longer texts. In *Proceedings of the 9th Language Resources and Evaluation Conference (LREC 2014)*, pages 4440–4443, Reykjavik.

Polajnar, Tamara, Laura Rimell, and Stephen Clark. 2015. An exploration of discourse-based sentence spaces for compositional distributional semantics. In *Proceedings of the EMNLP Workshop on*

Linking Models of Lexical, Sentential and Discourse-level Semantics (LSDSem 2015), pages 1–11, Lisbon.

Pollack, Jordan B. 1990. Recursive distributed representations. *Artificial Intelligence*, 46:77–105.

Reddy, Siva, Diana McCarthy, and Suresh Manandhar. 2011. An empirical study on compositionality in compound nouns. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 210–218, Chiang Mai.

Rimell, Laura, Stephen Clark, and Mark Steedman. 2009. Unbounded dependency recovery for parser evaluation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 813–821, Singapore.

Sadrzadeh, Mehrnoosh, Stephen Clark, and Bob Coecke. 2013. The Frobenius anatomy of word meanings I: subject and object relative pronouns. *Journal of Logic and Computation*, 23:1293–1317.

Schütze, Hinrich. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124.

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

Smolensky, Paul and Geraldine Legendre. 2006. *The Harmonic Mind: From Neural Computation to Optimality-Theoretic Grammar*. MIT Press, Cambridge, MA.

Socher, Richard, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1201–1211, Jeju Island.

Socher, Richard, Christopher D. Manning, and Andrew Y. Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS 2010 Deep Learning and Unsupervised Feature Learning Workshop*, Whistler.

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Chris Manning, Andrew Ng, and Chris Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642, Seattle, WA.

Steedman, Mark. 2000. *The Syntactic Process*. The MIT Press, Cambridge, MA.

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

Utsumi, Akira. 2012. Extending and evaluating a multiplicative model for semantic composition in a distributional semantic model. In *Proceedings of the 11th International Conference on Cognitive Modeling (ICCM 2012)*, pages 243–248, Berlin.

Vecchi, Eva Maria, Marco Baroni, and Roberto Zamparelli. 2011. (Linear) maps of the impossible: Capturing semantic anomalies in distributional space. In *Proceedings of the Workshop on Distributional Semantics and Compositionality (DiSCo 2011)*, pages 1–9, Portland, OR.

Zanzotto, Fabio Massimo, Lorenzo Ferrone, and Marco Baroni. 2015. When the whole is not greater than the combination of its parts: A "decompositional" look at compositional distributional semantics. *Computational Linguistics*, 41(1):165–173.