Evaluating Composition Models for Verb Phrase Elliptical Sentence Embeddings

Anonymous NAACL submission

Abstract

Ellipsis is a natural language phenomenon where part of a sentence is missing and its information must be recovered from its surrounding context, as in "Cats chase dogs and so do foxes.". Formal semantics has different methods for resolving ellipsis and recovering the missing information, but the problem has not been considered for distributional semantics, where words have vector embeddings and combinations thereof provide embeddings for sentences. In elliptical sentences these combinations go beyond linear as copying of elided information is necessary. In this paper, we develop different models for embedding VP-elliptical sentences. We extend existing verb disambiguation and sentence similarity datasets to ones containing elliptical phrases and evaluate our models on these datasets for a variety of non-linear combinations and their linear counterparts. We compare results of these compositional models to state of the art holistic sentence encoders. Our results show that non-linear addition and a non-linear tensor-based composition outperform the naive non-compositional baselines and the linear models, and that sentence encoders perform well on sentence similarity, but not on verb disambiguation.

1 Introduction

Compositional distributional semantics has so far relied on a tight connection between syntactic and semantic resources. Based on the assembly principle of compositionality, these models assign a sentence vector by applying a linear map to the individual word embeddings therein. The meaning of "cats chase dogs" is as follows in (1) additive, (2) multiplicative, and (3) tensor-based models:

(1) $\overrightarrow{cats} + \overrightarrow{chase} + \overrightarrow{dogs}$

(2) $\overrightarrow{cats} \odot \overrightarrow{chase} \odot \overrightarrow{dogs}$

(3) $\overrightarrow{cats}^{\top} \times (\overrightarrow{chase} \times \overrightarrow{dogs})$

Some linguistic phenomena, however, rely on copying resources while computing meaning; canonical examples thereof are anaphora and ellipsis, exemplified below:

(a) Cats clean themselves.

(b) Cats chase dogs, children do too.

More complex examples involve a structural ambiguity such as the following:

(c) Cats chase their tail, dogs too.

These lend themselves to a strict (dogs chase the cat's tail) and a sloppy reading (dogs chase their own tail). In these examples, the meaning of at least one part of the sentence is used twice, e.g. the subject in a, the verb phrase "chase dogs" in b. Such cases can often be extended to a situation in which a meaning is used more than twice, e.g. in "Cats chase their tail, dogs too, and so do foxes".

In order to develop distributional semantics for such sentences while respecting the principle of compositionality, one has a choice between a linear or a non-linear composition of resources. In the linear case, no information is copied, resulting in vector embeddings such as the following one (when only considering content words):

$$\overrightarrow{cats} + \overrightarrow{chase} + \overrightarrow{dogs} + \overrightarrow{children}$$

In the non-linear case, the necessary resources are copied to resolve the ellipsis, resulting in vectors embeddings such as:

$$\overrightarrow{cats} + \overrightarrow{chase} + \overrightarrow{dogs} + \overrightarrow{children} + \overrightarrow{chase} + \overrightarrow{dogs}$$

One has the same choice when dealing with multiplicative and tensor-based models. The question is which of these composition frameworks, i.e. linear versus non-linear, provides a better choice for embedding elliptical sentences.

In this paper, we provide some answers. Our starting point is the lambda logical forms of sentences, e.g. those produced by the approach of

Dalrymple et al. (1991), which uses a higher order unification algorithm to resolve ellipsis. We apply to these the lambdas-to-vectors mapping of Muskens and Sadrzadeh (2016, 2017) to homomorphically map the lambda terms into concrete vector embeddings resulting from a multitude of composition operators, such as addition, multiplication, and tensor-based. We work with four vector spaces (count-based, Word2Vec, GloVe, FastText) and three different verb embeddings, and contrast our compositional models with state of the art holistic sentence encoders.

We evaluate the sentence embeddings by using them in a verb disambiguation and in a sentence similarity task, created by extending previous SVO tasks from Grefenstette and Sadrzadeh (2011a) and Kartsaklis and Sadrzadeh (2013) to an elliptical setting, and obtaining new human judgements using the Amazon Mechanical Turk crowdsourcing tool. Our experiments show that in both tasks, the models that use a non-linear form of composition perform better than the models whose composition framework is linear, suggesting that resolving ellipsis contributes to the quality of the sentence embedding.

2 Background

Single-Word Embeddings: Distributional semantics on the word level relies on the embedding of word meaning in a vectorial form: by taking context words as the basis of a vector space one computes the vector components of each word by considering its distribution among corpus data. Then a similarity measure is defined on the vector space via the cosine similarity. In a count-based model, the context is taken to be a linear window and the corpus is traversed to collect raw co-occurrence counts. Then, a weighting scheme is applied to smooth the raw frequencies in the meaning representation. More discussion on count-based vector space models can be found in (Turney and Pantel, 2010), and a systematic study of the parameters of count-based word embeddings is given by (Kiela and Clark, 2014).

With the rise of deep learning techniques, much attention has been given to neural word embeddings (Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2016), which try to predict rather than observe, the context of a word by optimising an objective function based on the probability of observing a context.

Compositional Models: The key idea of compositional models is that the meaning of elementary constituents can be combined in a structured way to obtain a representation for larger phrases. In a distributional setting, having a compositional operator is imperative: a data-driven model would not be adequate given the sparsity of full sentences in a corpus. Moreover, it is not clear that sentences follow the distributional hypothesis.

Concrete composition operators can roughly be classified as simple and tensor-based. Simple models add or multiply the word vectors to obtain a sentence vector. The work of Mitchell and Lapata (2010) experiments with these models. Tensor-based models differ in that they represent complex words as vectors of a higher order: Baroni and Zamparelli (2010) represents adjectives as matrices which, applied to a word vector produce a vector representation of the compound adjective-noun combination. The account of (Coecke et al., 2010, 2013; Clark, 2015) generalises this to higher-order tensors, e.g. cubes for transitive verbs and hypercubes for ditransitive verbs. The benefit of a type-driven approach over the simple models is that they respect the grammatical structure of sentences: the meaning of "man bites dog" is distinct from that of "dog bites man" whereas in an additive/multiplicative model they would be identical. The trade-off is that the tensors themselves have to be learnt; where Baroni and Zamparelli (2010) apply regression learning to learn the content of adjective matrices, for transitive verbs there have been several approaches using multi-step regression learning (Grefenstette et al., 2013), relational learning (Grefenstette and Sadrzadeh, 2011a), or a combination of co-occurrence information with machine learning techniques (Polajnar et al., 2014a,b; Fried et al., 2015). A comparative study between count-based and neural embeddings in a compositional setting was carried out by (Milajevs et al., 2014).

Neural composition turns the problem of compositionality around by learning the composition operator instead of predicting the result. Examples are Skip-Thought Vectors (Kiros et al., 2015), the Distributed Bag of Words model (Le and Mikolov, 2014), InferSent (Conneau et al., 2017), and Universal Sentence Encoder (Cer et al., 2018).

Ellipsis, Formally: There exists many formal approaches to ellipsis and anaphora in the literature. These have generally taken either a syntactic

or a semantic form. Examples of the syntactic approaches are in the work of Hendriks and Dekker (1995); Morrill and Valentín (2015); Jäger (2006); Kubota and Levine (2017); these use directional extensions of categorial grammars that allow for the syntactic types at the site of ellipsis be unified with copies of the types at the antecedent of the elliptical phrase. Another approach deletes the syntactic structure at the ellipsis site and reconstruct it by copying across the antecedent structure (Fiengo and May, 1994; Merchant, 2004).

Semantic approaches (Dalrymple et al., 1991; Szabolcsi, 1987; Pulman, 1997) assume that ellipsis involves underspecification of content and resolve this by producing a predicate via a suitable abstraction from the antecedent. For instance, the elliptical phrase (b) "Cats chase dogs, children do too", will take an initial logical form (b_1); a resolution step (b_2) provides it with the lambda term in (b_3), which constitutes its final semantic form:

$$(b_1) \quad chase(cats, dogs) \wedge P(children)$$
$$(b_2) \quad P = \lambda x.chase(x, dogs)$$
$$(b_3) \quad (b_1) \rightsquigarrow_\beta chase(cats, dogs) \wedge chase(children, dogs)$$

he ambiguous example (d) "Cats chase their tails, dogs too" is treated similarly, but can now obtain its respective strict and sloppy readings by producing predicates (d_1) and (d_2) below:

$$(d_2) \quad P = \lambda x.chase(x, tail(cats))$$
$$(d_3) \quad P = \lambda x.chase(x, tail(x))$$

Mixed syntactic/semantic approaches have also been proposed to cover wider ranges of phenomena; see Kempson et al. (2015) for an overview.

The only existing work attempting to join ellipsis analysis with vector embeddings is the proposal of (Kartsaklis et al., 2016), which is preliminary work and gives unwanted results[1]. Below, we develop a new such approach.

3 Embeddings for Elliptical Phrases

Vectors and their basic operations can be *emulated* using a lambda calculus with constants for the relevant operations, as shown in (Muskens and Sadrzadeh, 2016). They assume a type I (a finite index set) and R (modelling the real numbers) and model any vector as a term of type $V := IR$; that is, as a function from indices to real numbers. Matrices can then be represented by types $M := IIR$ and in general a tensor of rank n will have type $T^n := I_1...I_nR$. The standard operations like scalar multiplication, addition, element wise multiplication and tensor contraction can be modelled with lambda terms as follows:

$$\begin{array}{lcll} \cdot & := & \lambda rvi.r \cdot v_i & : RVV \\ + & := & \lambda vwi.v_i + w_i & : VVV \\ \odot & := & \lambda vwi.v_i \cdot w_i & : VVV \\ \times_1 & := & \lambda mvij.\sum_j m_{ij} \cdot v_j & : MVV \\ \times_2 & := & \lambda cvijk.\sum_k c_{ijk} \cdot v_k & : T^3VM \end{array}$$

The first three definitions above extend the arithmetic operations of addition and multiplication on real numbers in R to lists of numbers in IR and define corresponding definitions on vectors, and so $\odot$ defines the pointwise multiplication of two vectors. The operation $\times_1$ defines matrix multiplication; $\times_2$ defines the tensor contraction between a cube c (in I^3R) and a list of numbers v.

c	$\mathcal{H}(c)$	$\mathcal{T}(c)$
`cn`	**cn**	V
`adj`	$\lambda v.$(**adj** $\times_1 v$)	VV
`adv`	$\lambda v.$(**adv** $\times_1 v$)	VV
`itv`	$\lambda v.$(**itv** $\times_1 v$)	VV
`tv`	$\lambda uv.$(**tv** $\times_2 v$) $\times_1 u$	VVV
`coord`	$\lambda P.\lambda Q.P \nabla Q$	VVV
`quant`	$\lambda vZ.Z$(**quant** $\times_1 v$)	$V(VV)V$

Table 1: Lambda Vector look up table for a tensor-based composition model. `cn`: a common noun, `adj`: adjective, `adv`: adverb, `itv`: intransitive verb, `tv`: transitive verb, `coord`: coordinator, `quant`: generalised quantifier; P, Q are variables of type V, and so are v, u, Z is a variable of type VV, ∇ is either $\odot$ or $+$.

The vector semantics of a lambda term m is computed by taking a homomorphic image over the set of its constants c. This image is computed compositionally from the vector or tensor embeddings of the constants c of m via their homomorphic images $\mathcal{H}(c)$, whose types are denoted by $\mathcal{T}(c)$.

Examples of these are given in Table 1 for a tensor-based composition model, where the boldface **c** denotes the vector/tensor embedding of c.

Using this table, we obtain homomorphic images of any lambda term over the constants. For instance, the lambda term of our exemplary resolved ellipsis phrase (b_3) $chase(cats, dogs) \wedge chase(children, dogs)$ is given the following semantic, obtained by computing $\mathcal{H}(b_3)$:

$$((\textbf{chase} \times_2 \textbf{dogs}) \times_1 \textbf{cats}) \nabla$$
$$((\textbf{chase} \times_2 \textbf{dogs}) \times_1 \textbf{children})$$

[1] The meaning of "Bill brought apples and John pears" coincides with that of "Bill and John brought apples and pears".

The constituents of the $\mathcal{H}(c)$ entries of Table 1 are only exemplary. Many other interpretations are possible. For instance, taking vector embeddings for all words and replacing all tensor contractions and ∇ by $+$ defines a purely additive model. The concrete models for transitive sentences that were evaluated by Milajevs et al. (2014) can all be derived by varying the $H(c)$ entries. Below are the sentences obtained by using the Copy Object (CO), Frobenius Additive (FA), Frobenius Multiplicative (FM) and Frobenius Outer (FO) instantiations of the verb, respectively:

$$\textbf{CO}: \quad \lambda os.o \odot (\textbf{verb} \times^{\top} s)$$
$$\textbf{FA}: \quad \lambda os.s \odot (\textbf{verb} \times o) + o \odot (\textbf{verb} \times^{\top} s)$$
$$\textbf{FM}: \quad \lambda os.s \odot (\textbf{verb} \times o) \odot o \odot (\textbf{verb} \times^{\top} s)$$
$$\textbf{FO}: \quad \lambda os.s \odot (\textbf{verb} \times o) \otimes o \odot (\textbf{verb} \times^{\top} s)$$

The vector semantics of the extensions of transitive sentences with VP elliptical phrases are obtained by taking each of the above as the semantics of each conjunct of the lambda logical form and interpreting the conjunction operation of $\wedge$ as either sum or multiplication.

4 Experimental Evaluation

For the evaluation of the model(s) of in the previous section, we built two new datasets and experimented with count based and neural vector spaces, and sentence encoders.

4.1 Building new datasets

In order to experiment with ellipsis, we extended the verb disambiguation dataset of Grefenstette and Sadrzadeh (2011a) and the transitive sentence similarity dataset of Kartsaklis and Sadrzadeh (2013), henceforth GS2011 and KS2013[2].

4.1.1 GS2011

The GS2011 verb disambiguation dataset contains 10 verbs, each with two possible interpretations. For each verb v and its two interpretations v_1 and v_2, the dataset contains human similarity judgments for 10 subject-object combinations. For instance, for the verb *meet* – ambiguous between *visit* and *satisfy* – the dataset contains the pairs $\langle$*system meet requirements, system satisfy requirements*$\rangle$ and $\langle$*system meet requirements, system visit requirements*$\rangle$. The more likely interpretation is marked as HIGH whereas the unlikely interpretation is marked LOW.

We extended this dataset as follows: for each combination of a verb triple (v, v_1, v_2) and a subject-object pair (s, o), where $\langle s\ v\ o,\ s\ v_1\ o\rangle$ is expected to have LOW similarity in the dataset and $\langle s\ v\ o,\ s\ v_2\ o\rangle$ is thus expected to have HIGH similarity, we selected a new subject s^* from the list of most frequent subjects for the verb v_2 such that it was significantly more frequent for v_2 than for v_1[3]. By doing so we strengthened the disambiguating effect of the context for each verb. The subject was selected such that the resulting elliptical phrase pairs made sense. For each combination and new subject considered, we added the two sentence pairs in the elliptical form

$$\langle s\ v\ o \textit{ and } s^* \textit{ does too, } s\ v_1\ o \textit{ and } s^* \textit{ does too}\rangle$$
$$\langle s\ v\ o \textit{ and } s^* \textit{ does too, } s\ v_2\ o \textit{ and } s^* \textit{ does too}\rangle$$

For example, for the verb triple (*draw, depict, attract*), and original sentence pairs

$$\langle \textit{man draw sword, man depict sword}\ \rangle$$
$$\langle \textit{man draw sword, man attract sword}\ \rangle$$

we selected the new subject *artist* and added two pairs, comparing *man draw sword and artist does too* with

man depict sword and artist does too
man attract sword and artist does too

We selected two new subjects for each combination, and in this way we obtained a dataset of roughly 400 entries. New human judgments were collected through Amazon Mechanical Turk, by prepending *the* to each noun and putting the phrase in the past tense. As with the original dataset, participants were asked to judge the similarity between sentence pairs using a discrete number between 1 and 7; 1 for highly dissimilar, 7 for highly similar. By inserting gold standard pairs of identical sentences we checked if participants were trustworthy. We collected 25 judgments per sentence pair but excluded participants that annotated less than 20 entries of the total dataset. We ended up with 55 different participants who ranked more than 20 entries of the total dataset, to give a final amount of ca. 9200 annotations. As an example, the verb *show* was a very hard case to disambiguate in the GS2011 dataset: *child show sign* had an average score of 2.5 with both *child picture sign* and *child express sign*. In the new dataset, with the extra subject *patient*, it got much clearer that the verb had to be interpreted as *express* with an average score of 5.869, versus 4.875 for *picture*.

[2]The new datasets will be made available online.

[3]As found in the combined ukWaC+WackyPedia corpus.

4.1.2 KS2013

The KS2013 sentence similarity dataset contains 108 transitive sentence pairs annotated with human similarity judgments. As opposed to the GS2011 dataset, subjects and objects of each sentence pair are not the same, so several different contexts get compared to one another. We extend this dataset to cover VP ellipsis by following a similar procedure as for GS2011. For each transitive sentence of the form $s\ v\ o$ in the dataset, we selected a new subject s^* from a list of most frequent subjects of the verb[4] and built elliptical entries $s\ v\ o$ *and* s^* *does too* in such a way that the meaning of the original transitive sentence got changed as little as possible and that the resulting elliptical phrase made sense. We then considered every transitive sentence pair in the dataset and added the new respective subjects to both sentences. For example, for the pair

$\langle$*school encourage child, employee leave company*$\rangle$

we selected *parent* and *student* to get the new pair

$\langle$*school encourage child and parent does too, employee leave company and student does too*$\rangle$

We chose two subjects for every original sentence, generating four possibilities for each sentence pair, and a new dataset of 432 entries. This dataset was also annotated using Amazon Mechanical Turk, after putting each verb in the past tense and prepending *the* to each noun in the dataset. Gold standard pairs of identical sentences were inserted to validate trustworthiness of participants. The final dataset contains ca. 9800 annotations by 42 different participants.

4.2 Vector Spaces

To provide a comprehensive study with robust results, we used four vector spaces: a count based vector space, and newly trained Word2Vec, GloVe, and FastText spaces, as detailed below.

Count-Based: We used the combined ukWaC and Wackypedia corpora[5] to extract raw co-occurrence counts, using as a basis the 2000 most frequently occurring tokens (after excluding the 50 most frequent ones). When extracting counts, we disregarded a list of stopwords that do not contribute to the content of the vectors. We used a context window of 5 around the focus word, and PPMI as weighting scheme. These settings were use in the original KS2013 dataset (Kartsaklis and Sadrzadeh, 2013).

Word2Vec: The Word2Vec embeddings we used were trained with the continuous bag of words model of (Mikolov et al., 2013) (CBOW). We trained this model on the combined and lemmatised ukWaC and Wackypedia corpora, using the implementation for Python available in the `gensim` package[6], with a minimum word frequency of 50, a window of 5, dimensionality 300, and 5 training iterations.

GloVe: The GloVe model (Pennington et al., 2014) considers the *ratio* of co-occurrence probabilities by minimising the least-squares objective between the dot product of two word embeddings and the log-probability of the words' co-occurrence. We trained a GloVe space on the combined and lemmatised ukWaC and Wackypedia corpora, using the code provided by the original authors[7]. Similar to the Word2Vec settings above, we trained 300 dimensional vectors with a minimum word frequency of 50 and a window of 5, but we trained with 15 iterations.

FastText: The FastText vectors are like Word2Vec, except the word vector takes into account subword information: words are represented as n-grams, for which vectors are trained. The final word vector will then be the sum of its constituent n-gram vectors (Bojanowski et al., 2016). We trained a FastText space with the same settings as the Word2Vec space (CBOW, minimum word frequency 50, dimensions 300, window 5, with 5 iterations), again using `gensim`.

4.2.1 Verb Matrices

In order to work with tensor-based models we had to represent verbs as matrices rather than as vectors. We generated verb tensors using two methods that have been used previously in the literature (Grefenstette and Sadrzadeh, 2011a; Kartsaklis and Sadrzadeh, 2014).

Relational: For each verb, its corresponding matrix is obtained by summing over the tensor product of the respective subject and object vectors of the verb (subjects and objects collected from the corpus):

$$\overline{verb} = \sum_i subj_i \otimes obj_i$$

[4] Again taken from the ukWaC+Wackypedia corpus.

[5] `wacky.sslmit.unibo.it`

[6] `radimrehurek.com/gensim`

[7] `nlp.stanford.edu/projects/glove`

Kronecker: For each verb, its corresponding matrix is obtained by taking the tensor product of the verb vector with itself:

$$\widetilde{verb} = \overrightarrow{verb} \otimes \overrightarrow{verb}$$

In the case of the count-based space, we trained verb matrices of dimensions 2000×2000, for the neural word embeddings the matrices had dimensions 300×300. We also experimented with the skip-gram extension of Maillard and Clark (2015) and the plausibility model of Polajnar et al. (2014a) but excluded the results because the obtained verb matrices were far below par.

4.3 Concrete Models

For the experiments, we had two main goals in mind: primarily, we wanted to verify that resolving ellipsis contributes to the performance of a compositional model. For this purpose we experimented with non-linear models, i.e. models that resolve the ellipsis (and thus use the verb and object resources twice) versus linear models, which do not resolve the ellipsis (and thus only use the verb and object once). Our second goal was to investigate whether amongst the models that resolve the ellipsis, the ones that did so in a tensor-based way, i.e. using tensors instead of vectors to represent the verbs, performed better than additive and multiplicative models, and how these compare to holistic sentence encoders. Hence, we considered three classes of models: linear vector models, non-linear vector models and tensor-based models.

Linear Vector Models: These models use every resource exactly once, following the pattern $\overrightarrow{w_1} \star \overrightarrow{w_2} ... \star \overrightarrow{w_n}$ for any sequence of words $w_1 w_2 ... w_n$. For an elliptical phrase "*subj verb obj and subj* does too*" it will compute the vector

$$\overrightarrow{subj} \star \overrightarrow{verb} \star \overrightarrow{obj} \star \overrightarrow{and} \star \overrightarrow{subj^*} \star \overrightarrow{does} \star \overrightarrow{too}$$

where $\star$ denotes either addition or multiplication.

Non-Linear Vector Models: Here, the assumption is that ellipsis is resolved but models do not respect word order. The meaning of "*subj verb obj and subj* does too*" now is

$$\overrightarrow{subj} \star \overrightarrow{verb} \star \overrightarrow{obj} \star \overrightarrow{subj^*} \star \overrightarrow{verb} \star \overrightarrow{obj}$$

Tensor-Based Models: These models all are assumed to resolve ellipsis and are based on various previous models (Grefenstette and Sadrzadeh, 2011b,a; Kartsaklis et al., 2012; Kartsaklis and Sadrzadeh, 2014). Essentially, the tensor-based meaning of "*subj verb obj and subj* does too*" is

$$T(\overrightarrow{subj}, \overline{verb}, \overrightarrow{obj}) \star T(\overrightarrow{subj^*}, \overline{verb}, \overrightarrow{obj})$$

where T is a transitive model from (Milajevs et al., 2014) and $\star$ interprets the conjunction of the two subclauses. For the verb matrix we used either the relational verb or the Kronecker verb, and for $\star$ we tried both addition and multiplication. We did consider a model which simply adds or multiplies the second subject without duplicating the verb phrase, but it performed worse than non-linear addition and multiplication so we did not include it in this paper.

Sentence Encoders: To compare the mentioned compositional models with state of the art neural baselines, we carried out our experiments with a four types of holistic sentence encoders, that take arbitrary text as input and produce an embedding. To properly compare with the compositional models above, we gave three different inputs to the encoders: a baseline encoding (Base), a resolved encoding (Res), and an encoding without functional words (Abl), all as below:

Base: "*subj verb obj and subj* does too*"
Res: "*subj verb obj and subj* verb obj*"
Abl: "*subj verb obj subj**"

We used six concrete pretrained encoders, available online: 4800-dimensional embeddings from the Skip-Thought model[8], 300-dimensional embeddings from two Doc2Vec implementations (Lau and Baldwin, 2016)[9], 4096-dimensional embeddings from two InferSent encoders[10], and 512-dimensional embeddings from Universal Sentence Encoder[11].

5 Results

To validate the quality of the trained word spaces, we evaluate on several standard word similarity tasks: we used Rubenstein & Goodenough (RG, 1965), WordSim353 (WS353, 2001), Miller & Charles (MC, 1991), SimLex-999 (SL999, 2015), and the MEN dataset (Bruni et al., 2012). The results are displayed in Table 2, for the spaces described in the previous section.

[8] github.com/ryankiros/skip-thoughts
[9] github.com/jhlau/doc2vec
[10] github.com/facebookresearch/InferSent
[11] tfhub.dev/google/universal-sentence-encoder

	RG	WS353	MC	SL999	MEN
Count	.6081	.3583	.5455	.2593	.5527
Word2Vec	.8227	**.6983**	**.7682**	**.4026**	**.7810**
GloVe	**.8312**	.6180	.7377	.3902	.7727
FastText	.7724	.5461	.6961	.4021	.7683

Table 2: Spearman ρ scores on word similarity tasks.

	CB	W2V	GloVe	FT
Verb Only Vector	.4363	.2406	.4451	.2290
Verb Only Tensor	.3295	.4376	.3942	.3876
Add. Linear	.4416	.2728	.3046	.1409
Mult. Linear	3250	-.0123	.1821	.2928
Add. Non-Linear	.4448	.3275	.3262	.1399
Mult. Non-Linear	.5029	.2087	.2446	.0440
Best Tensor	**.5385**	.4621	.3688	.4937
2nd Best Tensor	.5263	.4544	.3581	.4652

Table 3: Spearman ρ scores for the ellipsis disambiguation experiment. **CB**: count based, **W2V**: Word2Vec, **FT**: FastText.

Verb Disambiguation: Table 3 shows the results of the linear, non-linear and tensor-based models for this task, compared against a baseline in which only the verb vector or verb matrix is compared.

Our first observation is that generally, the highest performing models were tensor-based. The highest found correlation score was 0.5385 in the count based space for a tensor-based model (**CO** model above, Kronecker matrix, $\nabla = +$), with the Frobenius Additive model giving the second best result of 0.5263 (**FA** model above, Kronecker matrix, $\nabla = +$). For the neural spaces, the highest performing models were mostly tensor-based as well; they were always the Frobenius Additive (**FA**) model and the Frobenius Outer (**FO**) model, using the relational tensor and addition for the coordinator, except in the case of GloVe, where the Copy Object (**CO**) model was the second best. The only exception to this observation is the GloVe space, for which the baseline Vector Only model in fact has a higher correlation than any other model on that space.

Our second observation is that the non-linear variants of the additive and multiplicative models (which resolve ellipsis but in a naive way) show an increased performance over the linear models (which do not resolve ellipsis). All of this holds for all the four vector spaces, except for the Fast-Text space where the linear multiplicative model achieves significantly higher correlation (0.2834) than its non-linear counterpart (0.0249).

	D2V1	D2V2	ST	IS1	IS2	USE
Base	.1448	.2432	-.1932	.3471	**.3841**	.2693
Res	.2340	.2980	-.1720	.3436	.3373	.2770
Abl	.1899	.2423	-.1297	.3525	.3571	.2402

Table 4: Spearman ρ scores for the ellipsis disambiguation experiment. **D2V1**: Doc2Vec1, **D2V2**: Doc2Vec 2, **ST**: Skip-Thought, **IS1**: InferSent 1, **IS2**: InferSent 2, **USE**: Universal Sentence Encoder.

Overall, these results suggests that a logical resolving of ellipsis and further grammatical sensitivity benefits the performance of composition.

One interesting fact about our results is that the best compositional methods across the board were those that interpret the coordinator 'and' as addition; in set-theoretic semantics one interprets this coordinator as set intersection, which corresponds to multiplication rather than addition in a vectorial setting. We suggest that the feature intersection approach using multiplication leads to sparsity in the resulting vectorial representation, which then has a negative effect on the overall result. This would explain the case of FastText, since those vectors take into account subword information one would expect them to be more fine-grained and therefore conflate more of their features under multiplication.

The choice of verb matrix was mixed: for the count-based models the Kronecker matrix worked best, for the neural embeddings it was best to use the relational matrix.

In comparison, the sentence encoder results of Table 4 show the same trend that suggests that resolving ellipsis improves the quality of the embeddings: with the exception of the two InferSent encoders, the resolved models gave a higher correlation than their linear baseline. However, none of the encoder models come near the results achieved using the compositional models. Since the verb disambiguation dataset contains pairs of sentence that only differ in the verb, the task becomes very much grammar-oriented, and so we argue that the tensor-based models work better since they explicitly emphasise syntactic structure.

Sentence Similarity: For the extension of the KS2013 sentence similarity dataset, the results are shown in Table 5. We again wanted to see if resolving ellipsis benefits the compositional process.

	CB	W2V	GloVe	FT
Verb Only Vector	.4562	.5833	.4348	.6513
Verb Only Tensor	.3946	.5664	.4426	.5337
Add. Linear	.7000	.7258	.6964	.7408
Mult. Linear	.6330	.1302	.3666	.1995
Add. Non-Linear	.6808	**.7617**	.7103	.7387
Mult. Non-Linear	.7237	.3550	.2439	.4500
Best Tensor	.7410	.7061	.4907	.6989
2nd Best Tensor	.7370	.6713	.4819	.6871

Table 5: Spearman ρ scores for the ellipsis similarity experiment.

	D2V1	D2V2	ST	IS1	IS2	USE
Base	.5901	.6188	.5851	.7785	.7009	.6463
Res	.6878	.6875	.6039	**.8022**	.7486	.6791
Abl	.1840	.6599	.4715	.7815	.7301	.6397

Table 6: Spearman ρ scores for the ellipsis similarity experiment. **D2V1**: Doc2Vec1, **D2V2**: Doc2Vec 2, **ST**: Skip-Thought, **IS1**: InferSent 1, **IS2**: InferSent 2, **USE**: Universal Sentence Encoder.

This was in general true, although we observed a different pattern to the previous experiment.

In all cases, except for the FastText space, we saw that non-linear models in fact perform better than their linear counterparts. But this time the best tensor-based models only outperformed addition for the count-based space: the best models scored 0.7410 and 0.7370 (respectively for the **FO** and **FA** models above, Kronecker matrix, $\nabla = \odot$). Both Word2Vec and GloVe worked best with a non-linear additive model, with Word2Vec achieving the overall highest correlation score of 0.7617, and GloVe achieving 0.7103. For FastText, the highest score of 0.7408 was achieved by linear addition. What is more, the multiplicative model did not benefit from a non-linear approach in the case of GloVe (from 0.3666 to 0.2439), and the additive model had a similar decline in performance for the count-based space (from 0.7000 to 0.6808) and FastText (0.7408 to 0.7387). We can see that for the neural word embeddings the additive models work best, with all of them seeing a drop in performance for the tensor-based models.

Again, the best count-based models use the Kronecker matrix whereas the neural models benefit the most from using the relational matrix. However, this time the best count-based models used multiplication for coordination, the neural models preferring addition.

The sentence encoders worked a lot better in the similarity task, with all non-linear resolved models outperforming the baseline model, and the InferSent model even outperforming non-linear addition on a Word2Vec space. We argue this is the case for two reasons: first, the similarity dataset is more diffuse than the verb disambiguation dataset since sentence pairs now differ for every word in the sentence, giving more opportunity to exploit semantic similarity rather than syntactic similarity. Second, the embeddings from the sentence encoder are larger (4096), allowing them to effectively store more information to benefit the similarity score.

Overall we conclude again that resolving ellipsis improves the performance of composition, but this time the InferSent sentence encoder seems to work best, followed by the non-linear additive compositional model on Word2Vec, with tensor-based models only performing well in a count-based space.

6 Conclusion

In this paper we experimented with vector space semantics for VP ellipsis, working with a large variety of compositional models. We created two new datasets and compared the performance of several compositional methods, both linear and non-linear, across four vector spaces, and against state of the art holistic sentence encoders.

Our main conclusion is that resolving ellipsis improves performance: non-linear models almost always performed better than linear ones in both a verb disambiguation and a sentence similarity task. The highest performance on the verb disambiguation task was given by a grammar-driven, tensor-based model in a count-based vector space, whereas for the similarity task, the highest performance was achieved by the InferSent sentence encoder, followed by a non-linear additive model on a Word2Vec space. Although the neural word embeddings and sentence encoders were largely outperformed on the disambiguation dataset that places more emphasis on syntactic structure than on semantic similarity, they generally performed better in the sentence similarity case, where the distinction between syntactic and semantic similarity is more diffuse.

References

Marco Baroni 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*, pages 1183–1193. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 136–145. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Stephen Clark. 2015. Vector space models of lexical meaning. *Handbook of Contemporary Semantic Theory, The*, pages 493–522.

Bob Coecke, Edward Grefenstette, and Mehrnoosh Sadrzadeh. 2013. Lambek vs. lambek: Functorial vector space semantics and string diagrams for lambek calculus. *Annals of pure and applied logic*, 164(11):1079–1100.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical foundations for a compositional distributional model of meaning. *arXiv preprint arXiv:1003.4394*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

Mary Dalrymple, Stuart M Shieber, and Fernando CN Pereira. 1991. Ellipsis and higher-order unification. *Linguistics and philosophy*, 14(4):399–452.

Robert Fiengo and Robert May. 1994. *Indices and Identity*. MIT Press, Cambridge, Mass.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM.

Daniel Fried, 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 and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 731–736.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. *arXiv preprint arXiv:1301.6939*.

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

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011b. Experimenting with transitive verbs in a discocat. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 62–66. Association for Computational Linguistics.

Herman Hendriks and Paul Dekker. 1995. Links without locations. In *Proceedings of the Tenth Amsterdam Colloquium*, pages 339–358. Citeseer.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Gerhard Jäger. 2006. *Anaphora and type logical grammar*, volume 24. Springer Science & Business Media.

Dimitri Kartsaklis, Matthew Purver, and Mehrnoosh Sadrzadeh. 2016. Verb phrase ellipsis using frobenius algebras in categorical compositional distributional semantics. *DSALT Workshop, European Summer School on Logic, Language and Information*.

Dimitri Kartsaklis 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*, pages 1590–1601.

Dimitri Kartsaklis 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)*. Kyoto Japan.

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

Ruth Kempson, Ronnie Cann, Arash Eshghi, Eleni Gregoromichelaki, and Matthew Purver. 2015. Ellipsis. In S. Lappin and C. Fox, editors, *Handbook of Contemporary Semantic Theory*, 2nd edition, chapter 4. Wiley.

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

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Yusuke Kubota and Robert Levine. 2017. Pseudo-gapping as pseudo-vp-ellipsis. *Linguistic Inquiry*, 48(2):213–257.

Jey Han Lau and Timothy Baldwin. 2016. An empirical evaluation of doc2vec with practical insights into document embedding generation. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 78–86.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 302–308.

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

J. Merchant. 2004. Fragments and ellipsis. *Linguistics and Philosophy*, 27:661–738.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. 2014. Evaluating neural word representations in tensor-based compositional settings. *arXiv preprint arXiv:1408.6179*.

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

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Glyn Morrill and Oriol Valentín. 2015. Computational coverage of tlg: Nonlinearity. In *Proceedings of NLCS'15. Third Workshop on Natural Language and Computer Science*, volume 32, pages 51–63. EasyChair Publications.

Reinhard Muskens and Mehrnoosh Sadrzadeh. 2016. Context update for lambdas and vectors. In *International Conference on Logical Aspects of Computational Linguistics*, pages 247–254. Springer.

Reinhard Muskens and Mehrnoosh Sadrzadeh. 2017. Lambdas, vectors, and word meaning in context. In *Proceedings of the 21st Amsterdam Colloquium*, pages 65–74.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Tamara Polajnar, Luana Fagarasan, and Stephen Clark. 2014a. 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.

Tamara Polajnar, Laura Rimell, and Stephen Clark. 2014b. Using sentence plausibility to learn the semantics of transitive verbs. *arXiv preprint arXiv:1411.7942*.

Stephen G. Pulman. 1997. Higher order unification and the interpretation of focus. *Linguistics and Philosophy*, 20(1):73–115.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Anna Szabolcsi. 1987. Bound variables in syntax (are there any?). *Sixth Amsterdam Colloquium Proceedings*.

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