Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Journal Track

Compositionality Decomposed: How do Neural Networks Generalise? (Extended Abstract)*

Dieuwke Hupkes[1†], **Verna Dankers**[2], **Mathijs Mul**[2] and **Elia Bruni**[3]
[1]ILLC, University of Amsterdam
[2]University of Amsterdam
[3]University of Pompeu Fabra
{dieuwkehupkes, vernadankers, mathijsmul, elia.bruni}@gmail.com

Abstract

Despite a multitude of empirical studies, little consensus exists on whether neural networks are able to generalise *compositionally*. As a response to this controversy, we present a set of tests that provide a bridge between, on the one hand, the vast amount of linguistic and philosophical theory about compositionality of language and, on the other, the successful neural models of language. We collect different interpretations of compositionality and translate them into five theoretically grounded tests for models that are formulated on a task-independent level. To demonstrate the usefulness of this evaluation paradigm, we instantiate these five tests on a highly compositional data set which we dub PCFG SET, apply the resulting tests to three popular sequence-to-sequence models, and provide an in-depth analysis of the results.

1 Introduction

Most current models of natural language processing use *artificial neural networks* (ANNs). The architectural design of such models is not motivated by knowledge about linguistics or human processing, but they are nevertheless more successful than earlier-age (sub)symbolic models on a variety of natural language processing tasks. However, it remains difficult to assess if the composition functions that ANNs implement are truly appropriate for natural language and, importantly, to what extent they are in line with the vast amount of knowledge and theories about semantic composition from formal semantics and (psycho)linguistics.

One particular question that has recently attracted the attention of several researchers is whether ANNs are capable of learning *compositional* solutions. Despite a multitude of empirical studies on this topic, little concensus exists. One issue standing in the way of more clarity on this matter is that different researchers have different interpretations of what exactly it means to say that a model is or is not compositional, a point exemplified by the vast number of different tests that exist for compositionality [Lake and Baroni, 2018; Hupkes *et al.*, 2018; Johnson *et al.*, 2017; Bahdanau *et al.*, 2018; Saxton *et al.*, 2019; Loula *et al.*, 2018; Dessì and Baroni, 2019; Liška *et al.*, 2018; Bowman *et al.*, 2015; Mul and Zuidema, 2019]. We argue that to empirically test models for compositionality, it is necessary to first establish *what* is to be considered compositional behaviour.

With this work, we aim to contribute to clarity on this point, by presenting a study in which we collect different aspects of and intuitions about compositionality of language from linguistics and philosophy. We translate them into concrete tests that provide insight into the composition functions learned by neural models trained end-to-end on a downstream task.

2 Testing Compositionality

We individuate five aspects of compositionality that are explicitly motivated by theoretical literature on this topic.

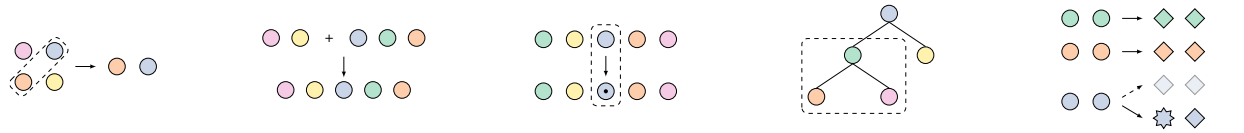
Systematicity. The first property we test for is *systematicity*. The term was introduced by Fodor and Pylyshyn, who used it to denote that "*[t]he ability to produce/understand some sentences is intrinsically connected to the ability to produce/understand certain others*" [Fodor and Pylyshyn, 1988]. This ability concerns the recombination of known parts and rules: anyone who understands a number of complex expressions also understands other complex expressions that can be built up from the constituents and syntactical rules employed in the familiar expressions.

Here, we ask not only if a model infers a systematic solution, but also whether the rules and constituents the model uses are in line with what we believe to be the actual rules and constituents underlying a particular data set or language. We test for systematicity by testing if a model can recombine constituents that have not been seen together during training. In particular, we focus on combinations of words *a* and *b* that meet the requirements that the model has only been familiarised with *a* in contexts excluding *b* and vice versa but the combination *a b* is plausible given the rest of the corpus.

Productivity. A notion closely related to systematicity is *productivity*, which concerns the open-ended nature of natural language: language appears to be infinite, but has to be stored with finite capacity. Hence, there must be some productive way to generate new sentences from this finite storage

*This paper is an extended abstract of a paper published in the Journal of Artificial Intelligence Research [Hupkes *et al.*, 2020].

†Contact author



(a) Systematicity (b) Productivity (c) Substitutivity (d) Localism (e) Overgeneralisation

Figure 1: A schematic depiction of our five compositionality tests.

Non-terminal rules

$S \rightarrow F_U\ S \mid F_B\ S\ ,\ S$
$S \rightarrow X$
$X \rightarrow XX$

Lexical rules

$F_U \rightarrow$ `copy` | `reverse` | `shift` | `echo` | `swap` | `repeat`
$F_B \rightarrow$ `append` | `prepend` | `remove_first` | `remove_second`
$X \rightarrow$ `A` | `B` | ... | `Z` | `A2` | ... | `B2` | ...

Unary functions F_U:

`copy` $x_1 \cdots x_n$	$\rightarrow$	$x_1 \cdots x_n$
`reverse` $x_1 \cdots x_n$	$\rightarrow$	$x_n \cdots x_1$
`shift` $x_1 \cdots x_n$	$\rightarrow$	$x_2 \cdots x_n\ x_1$
`swap` $x_1 \cdots x_n$	$\rightarrow$	$x_n\ x_2 \cdots x_{n-1}\ x_1$
`repeat` $x_1 \cdots x_n$	$\rightarrow$	$x_1 \cdots x_n\ x_1 \cdots x_n$
`echo` $x_1 \cdots x_n$	$\rightarrow$	$x_1 \cdots x_n\ x_n$

Binary functions F_B:

`append` **x**, **y**	$\rightarrow$	**x y**
`prepend` **x**, **y**	$\rightarrow$	**y x**
`remove_first` **x**, **y**	$\rightarrow$	**y**
`remove_second` **x**, **y**	$\rightarrow$	**x**

Figure 2: The context free grammar that describes the entire space of grammatical input sequences in PCFG SET (left) and the interpretation functions describing how the meaning of PCFG SET input sequences is formed (right).

[Chomsky, 1956; von Humboldt, 1836].

Both systematicity and productivity rely on the recombination of known constituents into larger compounds. To separate systematicity from productivity, in our productivity test we specifically focus on the aspect of unboundedness. We test whether a model can understand sentences that are *longer* than the ones encountered during training.

Substitutivity. A principle closely related to the principle of compositionality (here, we consider the version of [Partee, 1995]) is the principle of *substitutivity*, which states that if an expression is altered by replacing one of its constituents with another constituent with the same meaning, this does not affect the meaning of the expression [Pagin, 2003].

We test for substitutivity by probing under which conditions a model considers two atomic units to be synonymous. To do so, we artificially introduce synonyms and consider how the prediction of a model changes when an atomic unit in an expression is replaced by its synonym. We consider two different cases. Firstly, we analyse the case in which synonymous words occur equally often and in comparable contexts. Secondly, we consider pairs of words in which one of the words occurs only in very short sentences, which we call *primitive contexts*.

Localism. The principle of compositionality does not impose any restrictions on how different elements should be combined. As a consequence, the interpretation of the principle of compositionality depends on the type of constraints that are put on the semantic and syntactic theories involved (see e.g. [Janssen, 1983; Zadrozny, 1994]). In *global* or *weak* compositionality, the meaning of an expression follows from its global structure and the meanings of its atomic parts. In this interpretation, a compound can have a different meaning, depending on the larger expression that they are a part of (for some examples, see [Carnap, 1947]).

We test if a model's composition operations are local or global by comparing the meanings the model assigns to standalone sequences to those it assigns to the same sequences when they are part of a larger compound. More specifically, we compare a model's output when it is given a composed sequence *X*, built up from two parts *A* and *B* with the output the same model gives when it is forced to first separately process *A* and *B* in a local fashion.

Overgeneralisation. Lastly, we include also a notion that concerns the acquisition of the language by a model: we consider if models exhibit *overgeneralisation* when faced with *non*-compositional phenomena. Overgeneralisation is a language acquisition term, which refers to the scenario in which a language learner applies a general rule in a case that forms an exception to this rule. The relation of overgeneralisation with compositionality comes from the supposed evidence that overgeneralisation errors provide for the presence of symbolic rules in the human language system (see e.g. [Penke, 2012]). We follow this line of reasoning and take the application of a rule in a case where this is contradicted by the data as evidence that the model in fact internalised this rule.

We propose an experimental setup where a model's tendency to overgeneralise is evaluated by monitoring its behaviour on exceptions. We identify samples that do not adhere to the rules underlying the data distribution – *exceptions* – in the training data sets and assess a model's tendency to overgeneralise by observing how they respond to these exceptions during training.

3 Data

We consider an artificial task, which we dub PCFG SET.

Input sequences: syntax. The input alphabet of PCFG SET contains three types of words: words for unary and binary functions that represent *string edit operations*, elements to form the string sequences that these functions can be applied to, and a separator to separate the arguments of a binary function. The input sequences formed with this alphabet describe how a series of such operations are to be applied to a string argument. We generate input sequences with a PCFG, shown in Figure 2 (production probabilities are omitted).

Output sequences: semantics. The meaning of a PCFG SET input sequence is constructed by recursively applying

Experiment	LSTMS2S	ConvS2S	Transformer
Task accuracy*	0.79 ± 0.01	0.85 ± 0.01	0.92 ± 0.01
Systematicity*	0.53 ± 0.03	0.56 ± 0.01	0.72 ± 0.00
Productivity*	0.30 ± 0.01	0.31 ± 0.02	0.50 ± 0.02
Substitutivity, *ED* †	0.80 ± 0.00	0.95 ± 0.00	0.98 ± 0.00
Substitutivity, *P* †	0.60 ± 0.01	0.58 ± 0.01	0.90 ± 0.00
Localism†	0.46 ± 0.00	0.59 ± 0.01	0.54 ± 0.02
Overgeneralisation*	0.68 ± 0.04	0.79 ± 0.06	0.88 ± 0.07

Table 1: General task accuracy and performance per test for PCFG SET, averaged over three runs. Two performance measures are used: *sequence accuracy*, indicated by *, and *consistency score*, indicated by †.

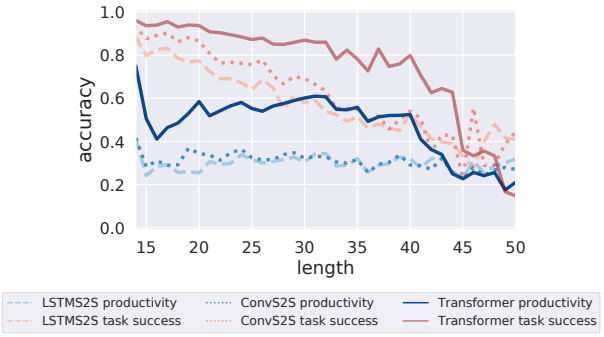


Figure 3: Task accuracy (in red) and productivity scores (in blue) of the three architectures as a function the length of the input sequence.

the string edit operations specified in the sequence. This mapping is governed by the interpretation functions listed in Figure 2 (right).

Data construction. We use the probabilistic nature of the PCFG SET input grammar to enforce a distribution of lengths and parse tree depth of a more natural data set (WMT2017, [Bojar *et al.*, 2017]). We set the size of the string alphabet to 520 and create a base corpus of around 100 thousand distinct input-output pairs, limiting the length of the string arguments given to the functions to 5. We use 85% of this corpus for training, 5% for validation and 10% for testing.

4 Experiments and Results

We compare three currently popular neural architectures for sequence-to-sequence language processing tasks: a recurrent architecture (LSTMS2S) [Sutskever *et al.*, 2014], a convolution-based architecture (ConvS2S) [Gehring *et al.*, 2017] and a transformer model (Transformer) [Vaswani *et al.*, 2017]. For every architecture, we train three models.[1] A summary of the results is shown in Table 1.

4.1 Systematicity

The task success results for PCFG SET (Table 1, row 1) already reflect whether models can recombine functions and input strings that were not seen together during training. In the systematicity test, we focus explicitly on models' ability to interpret pairs of functions that were never seen together while training. We select four pairs of functions to evaluate and redistribute the training and test data such that the training data does not contain any input sequences including these specific four pairs and all sequences in the test data contain at least one.

Results. Following the overall task accuracy, also for the systematicity test, Transformer obtains higher scores than both LSTMS2S and ConvS2S. Intriguingly, the systematicity scores of all models are substantially lower than their overall task accuracies. This large difference is surprising, since PCFG SET is constructed such that a high task accuracy requires systematic recombination. As such, these results serve as a reminder that models may find unexpected solutions, even when the data is very carefully constructed.

[1] All data, code and models are available online at https://github.com/i-machine-think/am-i-compositional

4.2 Productivity

Longer sequences are more difficult for all models, even if their length falls within the range of lengths observed during training (See Figure 3, red lines). With our productivity test, we test if this is due to an inherent difficulty of longer sequences or is related to models' inability to extrapolate to unseen lengths. We redistribute the train and test data such that there is no evidence at all for longer sequences in the training set. Sequences containing up to eight functions are collected in the training set, while input sequences containing at least nine functions are used for evaluation.

Results. All models have great difficulty with extrapolating to sequences with a higher length than those seen during training. Figure 3 depicts the performance of the three models in relation to the length of the input sequences (blue lines) compared with the task accuracy of the standard PCFG SET test data for the same lengths. For all models, the productivity scores are lower for almost all sequence lengths. With the difficulty of longer sequences factored out, we can conclude that this decrease in performance is solely caused by the decrease in evidence for such sequences and that models in fact struggle to productively generalise to longer sequences.

4.3 Substitutivity

To test for substitutivity, we select two binary and two unary functions, for which we artificially introduce synonyms (F_{syn}), during training. The introduced synonyms have the same interpretation functions as the terms they substitute, and are thus semantically equivalent to their counterparts. We consider two different conditions that differ in the syntactic distribution of the synonyms in the training data.

In the first condition, we randomly replace half of the occurrences of the chosen functions F with F_{syn}, keeping the target constant. In this test, F and F_{syn} are distributionally similar. In the second, more difficult condition, we introduce F_{syn} only in *primitive* contexts, where F is the only function call in the input sequence. In this *primitive* condition, the function F and its synonymous counterpart F_{syn} are distributionally not equivalent. We evaluate models on how robust

they are to the meaning-invariant synonym substitutions in the input sequence. We quantify this with a *consistency score*, which expresses a pairwise equality between the model's output before and after the synonym substitution.

Results. For the substitutivity experiment where words and synonyms are equally distributed, the scores of Transformer and ConvS2S are nearly on par. Furthermore, both architectures put words and their synonyms closely together in the embedding space (not shown). Surprisingly, even in this relatively simple condition where the words are distributionally identical, words and synonyms are at very distinct positions in the LSTMS2S embedding space.

In the primitive substitutivity test, all scores decrease substantially, although all models do still pick up that there is a similarity between a word and its synonym. This is reflected not only in the consistency scores but is also evident from the distances between words and their synonyms, which are substantially lower than the average distances to other function embeddings (not shown here). For LSTMS2S, the average distance is very comparable to the average distance observed in the equally distributed setup. Its consistency score, however, goes down substantially, indicating that word distances (computed between embeddings) give an incomplete picture of how well models can account for synonymity when there is a distributional imbalance.

4.4 Localism

We test for localism by considering models' behaviour when a subsequence in an input sequence is replaced with its meaning. More specifically, we compare the output sequence that is generated by a model for a particular input sequence with the output sequence that the same model generates when we explicitly unroll the processing of the input sequence (for an example, see Figure 4).

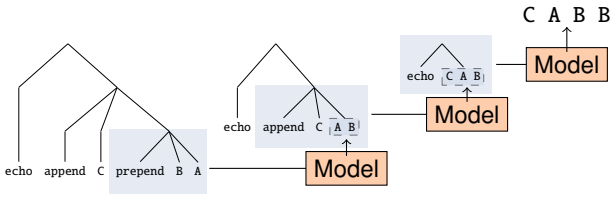


Figure 4: An example of the unrolled computation of the meaning of the sequence `echo append C , prepend B , A` for the localism test.

Results. None of the evaluated architectures obtains a high consistency score for this experiment. In a small manual analysis, we observe that the most common mistakes involve unrolled samples that contain function applications to string inputs with more than five characters.

4.5 Overgeneralisation

To test for overgeneralisation, we manually add exceptions to the data set. We select four pairs of functions that are assigned a new meaning when they appear together in an input sequence. We monitor the accuracy of both the original and the exception targets during training and compare how often

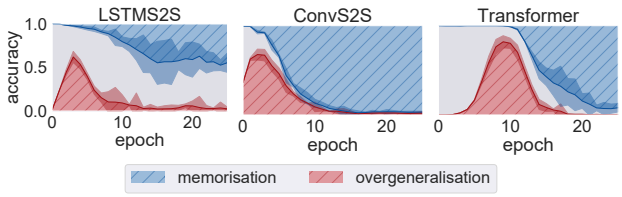


Figure 5: Overgeneralisation profiles for exception percentage 0.1%.

a model correctly memorises the exception target and how often it overgeneralises to the compositional meaning, despite the evidence in the data. We summarise a model's tendency to overgeneralise by the highest overgeneralisation accuracy encountered during training, and visualise the development of both memorisation and overgeneralisation during training.

Results. We test overgeneralisation for several different *exception percentages*, which indicate the number of occurrences of a function that is replaced by an exception. The results indicate that all architectures have a tendency to overgeneralise, but the degree of overgeneralisation depends strongly on the number of exceptions present in the data. All architectures show overgeneralisation behaviour for low exception percentages lower than 0.5%, but hardly any overgeneralisation is observed when 0.5% of a function's occurrence is an exception. When the percentage of exceptions becomes too low all models have difficulties memorising them at all. LSTMS2S, in general, appears to find it difficult to accommodate both rules and exceptions at the same time.

5 Conclusion

We proposed an evaluation framework containing a series of tests that translate theoretical concepts related to compositionality of language into behavioural tests for models of language. Our evaluation framework contains five independent tests that consider complementary aspects of compositionality that are frequently mentioned in the literature.We instantiated the five tests on a compositional artificial data set we dub PCFG SET. This data set is designed such that modelling it adequately should require a compositional solution, and it is generated such that its length and depth distributions match those of a natural corpus of English. We compared three popular sequence-to-sequence architectures: an LSTM-based, a convolution-based and an all-attention model. For each test, we conducted a number of auxiliary tests that can be used to further increase the understanding of how this aspect is treated by a particular architecture.

While the overall accuracy on PCFG SET was relatively high for all models, a more detailed picture is given by the five compositionality tests, which indicate that, despite our careful data design, high scores do still not necessarily imply that the trained models fully represent the true underlying generative system. These results illustrate well that to test for compositionality in neural networks, it does not suffice to consider an accuracy score on a single downstream task, even if this task is designed to be highly compositional. As such, the results themselves demonstrate the need for the more extensive set of evaluation criteria that we aim to provide.

References

[Bahdanau *et al.*, 2018] Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron Courville. Systematic generalization: What is required and can it be learned? In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, 2018.

[Bojar *et al.*, 2017] Ondrej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno-Yepes, Philipp Koehn, and Julia Kreutzer, editors. *Proceedings of the Second Conference on Machine Translation (WMT)*, 2017.

[Bowman *et al.*, 2015] Samuel R Bowman, Christopher D Manning, and Christopher Potts. Tree-structured composition in neural networks without tree-structured architectures. In *Proceedings of the 2015th International Conference on Cognitive Computation: Integrating Neural and Symbolic Approaches*, pages 37–42, 2015.

[Carnap, 1947] Rudolf Carnap. *Meaning and necessity: A study in semantics and modal logic*. University of Chicago Press, 1947.

[Chomsky, 1956] Noam Chomsky. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124, 1956.

[Dessì and Baroni, 2019] Roberto Dessì and Marco Baroni. CNNs found to jump around more skillfully than RNNs: Compositional generalization in seq2seq convolutional networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3919–3923, 2019.

[Fodor and Pylyshyn, 1988] Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: a critical analysis. *Cognition*, 28(1-2):3–71, 1988.

[Gehring *et al.*, 2017] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1243–1252, 2017.

[Hupkes *et al.*, 2018] Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61:907–926, 2018.

[Hupkes *et al.*, 2020] Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: how do neural networks generalise? *Journal of Artificial Intelligence Research*, (67):757–795, 2020.

[Janssen, 1983] Theo Janssen. *Foundations and applications of Montague grammar*. Mathematisch Centrum, 1983.

[Johnson *et al.*, 2017] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1988–1997, 2017.

[Lake and Baroni, 2018] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4487–4499, 2018.

[Liška *et al.*, 2018] Adam Liška, Germán Kruszewski, and Marco Baroni. Memorize or generalize? Searching for a compositional RNN in a haystack. In *Proceedings of AEGAP (FAIM Joint Workshop on Architectures and Evaluation for Generality, Autonomy and Progress in AI)*, 2018.

[Loula *et al.*, 2018] João Loula, Marco Baroni, and Brenden M Lake. Rearranging the familiar: Testing compositional generalization in recurrent networks. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 108–114, 2018.

[Mul and Zuidema, 2019] Mathijs Mul and Willem Zuidema. Siamese recurrent networks learn first-order logic reasoning and exhibit zero-shot compositional generalization. *CoRR*, abs/1906.00180, 2019.

[Pagin, 2003] Peter Pagin. Communication and strong compositionality. *Journal of Philosophical Logic*, 32(3):287–322, 2003.

[Partee, 1995] Barbara Partee. Lexical semantics and compositionality. *An invitation to cognitive science: Language*, 1:311–360, 1995.

[Penke, 2012] Martina Penke. The dual-mechanism debate. In *The Oxford handbook of compositionality*. Oxford University Press, 2012.

[Saxton *et al.*, 2019] David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3104–3112, 2014.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017.

[von Humboldt, 1836] Wilhelm von Humboldt. *On language: The diversity of human language-structure and its influence on the mental development of mankind*. 1836.

[Zadrozny, 1994] Wlodek Zadrozny. From compositional to systematic semantics. *Linguistics and philosophy*, 17(4):329–342, 1994.