# You Shall Know a Construction by the Company it Keeps: Computational Construction Grammar with Embeddings

**Lara Verheyen**
Artificial Intelligence Laboratory
Vrije Universiteit Brussel, Belgium
lara.verheyen@ai.vub.ac.be

**Jonas Doumen**
Faculty of Arts
KU Leuven, Belgium
jonas.doumen@kuleuven.be

**Paul Van Eecke***
Artificial Intelligence Laboratory
Vrije Universiteit Brussel, Belgium
paul@ai.vub.ac.be

**Katrien Beuls***
Faculté d'informatique
Université de Namur, Belgium
katrien.beuls@unamur.be

## Abstract

Linguistic theories and models of natural language can be divided into two categories, depending on whether they represent and process linguistic information *numerically* or *symbolically*. Numerical representations, such as the embeddings that are at the core of today's large language models, have the advantage of being learnable from textual data, and of being robust and highly scalable. Symbolic representations, like the ones that are commonly used to formalise construction grammar theories, have the advantage of being compositional and interpretable, and of supporting sound logic reasoning. While both approaches build on very different mathematical frameworks, there is no reason to believe that they are incompatible. In the present paper, we explore how numerical, in casu distributional, representations of linguistic forms, constructional slots and grammatical categories can be integrated in a computational construction grammar framework, with the goal of reaping the benefits of both symbolic and numerical methods.[1]

## 1 Introduction

Linguistic theories and models of natural language typically fall into one of two categories. The first category represents and processes linguistic information *symbolically*, adopting formal logic as the underlying framework. The second category represents and processes linguistic information *numerically*, adopting the framework of linear algebra.

The symbolic approach is widely used to formalise construction grammar theories (Fillmore, 1988; Kay and Fillmore, 1999; Steels and De Beule, 2006; Michaelis, 2008; Sag, 2012), with symbolic programming techniques forming the backbone of their computational implementations (Bergen and Chang, 2005; Steels and De Beule, 2006; van Trijp et al., 2022). Symbolic representations bring the advantage of being compositional and interpretable, and of supporting sound logic reasoning.

The numerical approach is widely adopted in the field of natural language processing (NLP), and lies for example at the core of today's large language models (LLMs) (Mikolov et al., 2013; Vaswani et al., 2017; Lenci, 2018; Devlin et al., 2019). In essence, numerical representations of linguistic information are learnt from textual data, thus based on the distribution of tokens with respect to each other. Apart from being learnable from raw textual input, distributional representations bring the advantage of being robust against noise, of generalising well to new data, and of scaling effectively with respect to growing amounts of input data from different domains.

As both approaches are rooted in very different mathematical frameworks, namely formal logic versus linear algebra, the integration of concepts and techniques from both fields is not straightforward. At the same time, logic-based and distributional approaches are widely regarded as complementary, and there exists no a priori reason to believe that they would be in any way incompatible.

In this paper, we explore how distributional representations can be integrated in a computational construction grammar framework, and how this integration of symbolic and numerical methods can enhance the robustness and generality of constructional language processing. In particular, we show how distributional representations of (i) linguistic forms, (ii) constructional slots, and (iii) gram-

*Joint last authors.
[1]The authors declare that this paper was conceived and written without the assistance of generative writing aids.



*Proceedings of the Second International Workshop on Construction Grammars and NLP*, pages 75–83
September 24, 2025, 

matical categories can be integrated into the data structures and algorithms that underlie Fluid Construction Grammar (FCG) (Steels, 2004; Beuls and Van Eecke, 2023). Through a variety of examples, we demonstrate how this integration can benefit a broad-coverage FCG grammar learnt from PropBank-annotated corpora. Finally, we conclude that the future of construction grammar is neither symbolic nor numerical, but lies in a combination of both paradigms.

## 2 Background

For the purposes of this exploration, we start from a symbolic construction grammar that was learnt from a collection of corpora in which English utterances were semantically annotated with PropBank rolesets (Palmer et al., 2005).[2] The grammar was learnt using the Fluid Construction Grammar framework (Beuls and Van Eecke, 2023) and holds 21,052 constructions that can be used to annotate open-domain English utterances with argument structure information in the form of semantic frames.

The basic architecture of the grammar is laid out in Figure 1. The input to the grammar consists of an utterance, in this case *"The doctor wrote him a prescription."*, which is analysed on the fly into its immediate constituents using the Berkeley neural parser (Stern et al., 2017) (see Step 1). A first type of construction identifies possible frame-evoking elements based on their lemma and part-of-speech tag. Here, the WRITE(V)-CXN indicates that the constituent 'unit-4' might represent a frame-evoking element by adding the for now underspecified `roleset` feature to this unit, along with a lexical category proper to the WRITE(V)-CXN (see Step 2). The resulting unit is shown as 'unit-4a'. The addition of a lexical category unlocks the application of a second type of construction that attributes semantic roles based on an utterance's constituent layout. In the example, a ditransitive construction that is compatible with the category contributed by the WRITE(V)-CXN respectively attributes the roles 'arg0' (prototypical agent),'arg1' (prototypical patient) and 'arg2' (prototypical beneficiary) to the constituents 'unit-2', 'unit-6' and 'unit-5'. The ditransitive construction also contributes its own grammatical category to the unit containing the frame-evoking element (see Step 3). The result is shown as 'unit-4b'. A final construction that is compatible with both the lexical category contributed by the WRITE(V)-CXN and the grammatical category contributed by the ditransitive construction fills out the value of the `roleset` feature, in this case PropBank's `write.01` roleset (see Step 4). The result is shown as 'unit-4c'. As the example utterance only expresses a single frame, the construction application process stops here. The resulting frame is then extracted and rendered into a more human-readable format (see Step 5). All constructions as well as the categorial links that express compatibility between constructional units were learned from corpus data using FCG's `fcg-propbank` subsystem (Van Eecke and Beuls, 2025).

## 3 Distributional Representations of Linguistic Forms

A classical argument against symbolic methods revolves around their reliance on exact matches between symbols. For example, the symbol DOG is in its representation not any more closely related to the symbol PUPPY than it is to the symbols CAT or PHILOLOGY. Representationally, symbols are either equal to or different from each other. Standard FCG builds on this property for implementing the process of construction application, where features and values in the pre- and postconditions of constructions are unified with their counterparts in the transient structure based on the equality of symbols (Steels and De Beule, 2006). The classical argument against symbolic methods points to the brittleness of relying on exact matches, as symbolic models tend to have difficulties handling input that even slightly deviates from what is expected. Distributional methods on the other hand represent linguistic forms in a vector space, where forms are compared in terms of distributional similarity rather than representational equality. In such models, the distance between DOG and PUPPY will effectively be smaller than the distance between DOG and PHILOLOGY.

Take for example the utterance "*So I mean that right there it enraged me.*" (OntoNotes `bc/cnn/00/cnn_0000`), which expresses an instance of the `mean.01` roleset and an instance of the `enrage.01` roleset. The base grammar from the previous section however, only retrieves the

[2] In particular, the examples throughout this paper were selected from the test sets of the OntoNotes (Weischedel et al., 2013) and English Web Treebank (EWT) (Bies et al., 2012) corpora, while the grammar itself was learnt from the training sets of the same corpora.

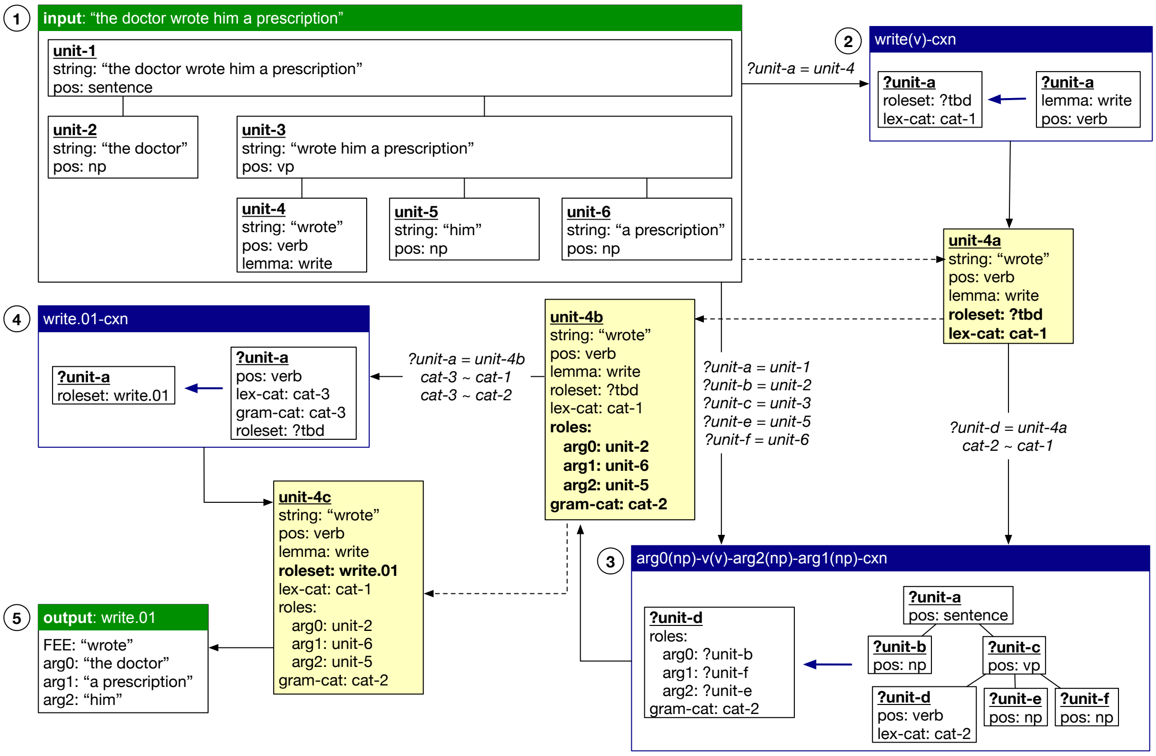


Figure 1: Illustrative example of the symbolic base grammar comprehending "*The doctor wrote him a prescription.*" (1). The WRITE(V)-CXN identifies a potential frame-evoking element (2). A ditransitive construction then attributes the semantic roles of agent ('arg0'), patient ('arg1') and beneficiary ('arg2') to particular constituents (3). The WRITE.01-CXN determines the roleset (`write.01`) of the evoked frame (4), after which the result is shown (5).

instance of the `mean.01` roleset. Upon closer inspection, it turns out that the verb "*enrage*" did not occur anywhere in the training corpus and that consequently no construction was learnt that identifies "*enraged*" as a possible frame-evoking element. At the same time, many constructions were learnt for other verbs that are distributionally close to "*enrage*" (such as "*anger*", "*madden*" or "*infuriate*") and that even appear in similar argument structure constructions ("`NP:Arg0` *(angers — maddens — infuriates)* `NP:Arg1`"). The reason why these constructions cannot apply is simply that there is no exact match between the lemma of the observed token ("*enrage*") and the lemmas incorporated in the constructions ("*anger*", "*madden*" and "*infuriate*").

As a first step in the integration of symbolic and distributional methods, we will represent lemmata distributionally rather than symbolically in FCG constructions and transient structures. Concretely, we substitute the `lemma` features in the units of the input transient structure by `embedding` features that hold as their value pointers to pre-trained, 100-dimensional GloVe embeddings of the original lemmata (Pennington et al., 2014). This is shown for the example utterance "*So I mean that right there it enraged me.*" in Step (1) of Figure 2. Likewise, we substitute the `lemma` features in the constructions of the grammar by `embedding` features that point to pre-trained GloVe embeddings (see Step (2)). The INFURIATE(V)-CXN thereby does not match on the symbol INFURIATE any more but on the GloVe embedding for the form "*infuriate*". We also modify FCG's unification algorithms in such a way that they no longer compute symbol equality when handling vectors, but compute their cosine similarity. These similarities are then used to create scored unification results and rank possible construction applications. In the example, the highest-ranked result is yielded by the INFURIATE(V)-CXN, which matches on the unit holding "*enraged*" with a cosine similarity of 0.84. Then, the transitive construction that was learned during training to be compatible with the INFURIATE(V)-CXN can apply, followed by the INFURIATE.01-CXN. This results in the extraction of an instance of the `infuriate.01` roleset, with "*enraged*" as the frame-evoking element and "*it*" and "*me*" respectively as its 'arg0' ('*causer of anger*') and 'arg1' ('*angry entity*').

This example demonstrates how constructions

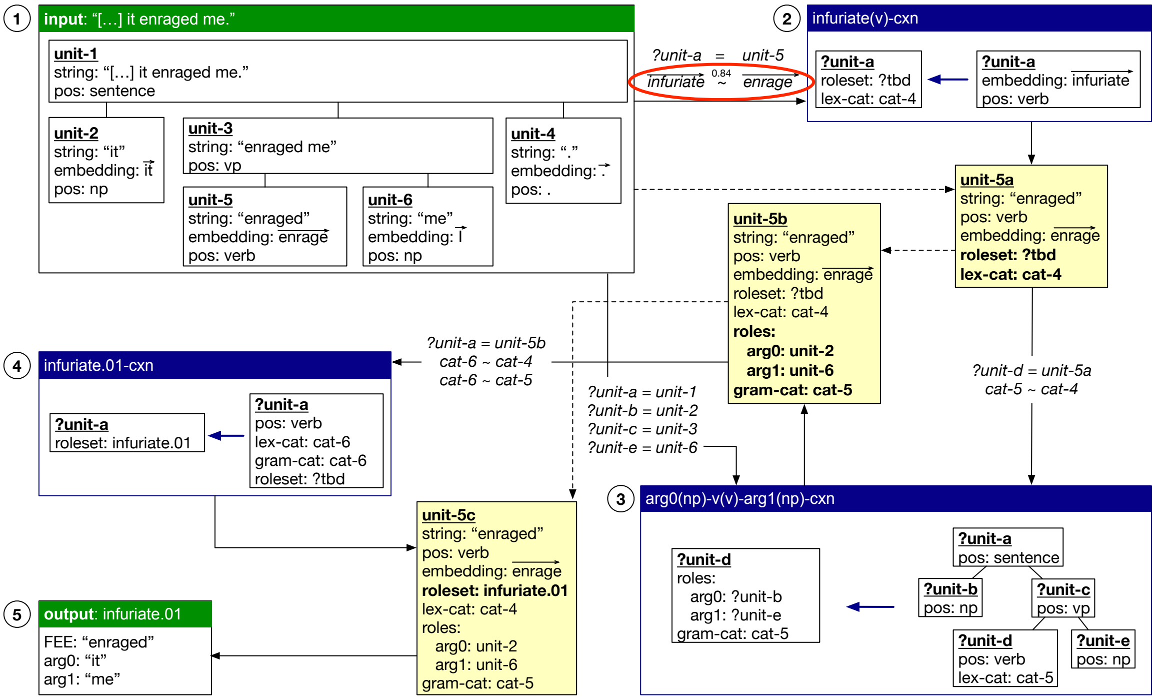


Figure 2: Schematic illustration of the integration of distributional token representations in constructional language processing. The INFURIATE(V)-CXN identifies "*enraged*" as a possible frame-evoking element based on the high cosine similarity between the embeddings for "*enrage*" and "*infuriate*", recovering from the absence of the token "*enrage*" in the training corpus.

can apply without requiring a perfect symbolic match, relying on the distributional closeness of forms, in this case the lemmata of potential frame-evoking elements. This was achieved by integrating numerical representations of linguistic information (i.c. word embeddings) and operations over them (i.c. cosine computation) with symbolic representations (i.c. feature structures) and operations over these (i.c. unification). In fact, this integration can be considered an extension of the way matches between categories in the categorial network of a grammar were already integrated into FCG's unification algorithms (see Van Eecke, 2018).

## 4 Distributional Representations of Constructional Slots

Now that we have represented the substantive material in constructions, such as word forms and lemmata, using word embeddings, we take the same idea a step further and integrate distributional representations of constructional slots. Let us consider as an example the utterance "*Jesus taught the people in the Temple area every day.*" (OntoNotes `ontonotes/pt/nt/42/nt_4219`). The base grammar yields two competing analyses which it considers equally fit. Both analyses identify an instance of the `teach.01` roleset, in which "*Jesus*" takes up the role of 'arg0' ('*teacher*'). One analysis assigns the role of 'arg1' ('*subject*') to "*the people*", while the other assigns it the role of 'arg2' ('*student(s)*'). The two analyses differ in the argument structure construction that is used. In the first analysis, a transitive construction applies that maps the noun phrase after the verb to the 'arg1' role, whereas in the second analysis, a construction applies that maps this noun phrase to the 'arg2' role. Both constructions can be traced back to utterances in the training corpus, such as "*Her mother taught [Sunday School]$_{arg1}$ for 50 years.*" (OntoNotes `bn/cnn/03/cnn_0324`) and "*You teach [others]$_{arg2}$, so why don't you teach [yourself]$_{arg2}$ ?*" (OntoNotes `pt/nt/45/nt_4502`). This ambiguity cannot be resolved on the level of the morphosyntactic structure of the utterances and necessitates modelling the lexical content of the slot fillers.

We extend the idea of including an `embedding` feature to the units in the initial transient structure also to phrasal units. The embeddings on phrasal level are in this prototype computed as the sum of the GloVe embeddings of the lemmas of their constituent parts (see Step 1 in Figure 3). In each

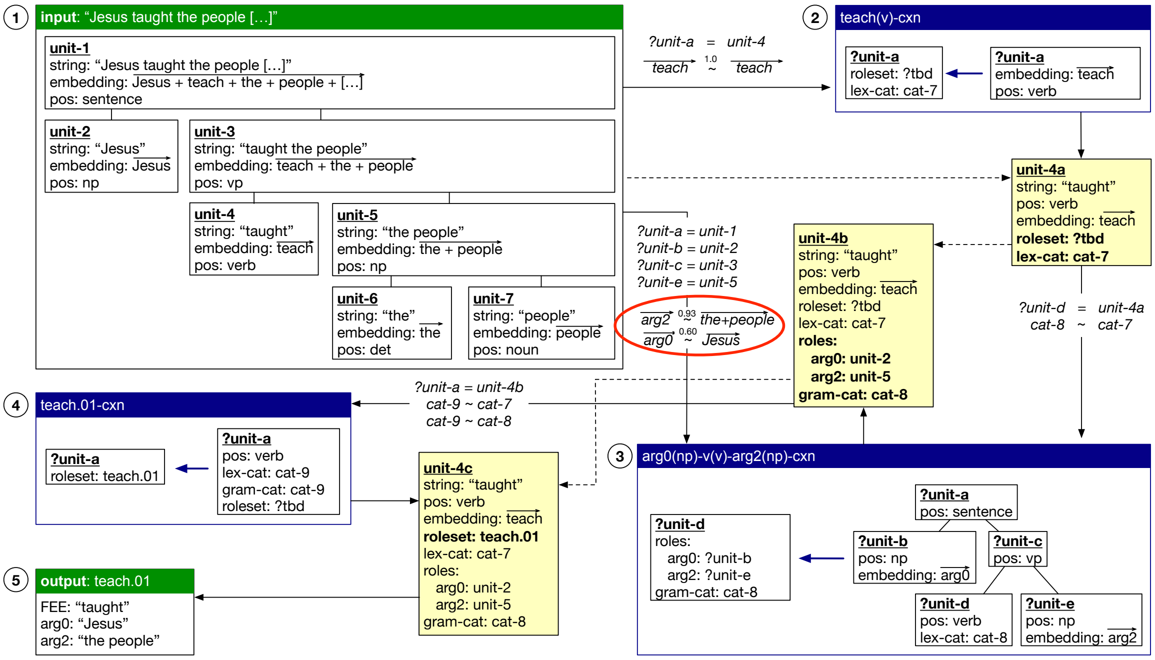


Figure 3: Schematic illustration of the integration of distributional information for representing prototypical slot fillers within argument structure constructions. The embeddings in the argument structure constructions are computed based on their fillers as observed in the training corpus.

argument structure construction, we also add an `embedding` feature to all units that are assigned a role (see Step 3). These embeddings are computed by averaging the summed embeddings of all lemmata for all fillers observed in a particular slot during training. For example, the value of the `embedding` feature in the 'arg1' slot of a transitive construction would point to a vector representing the prototypical patient/undergoer that fills that slot. The unification algorithm described in the previous section, which computes cosine similarities when handling vectors, is again used. In our example, this leads to two construction application results, one for each of the two argument structure constructions, with the one where the 'arg2' role is taken up by "*the people*" is ranked highest. Indeed, the match between "*the people*" and the prototypical vector of the 'arg2' slot of this construction is considerably higher than the match between "*the people*" and the prototypical vector for the 'arg1' slot in the other construction. The highest-ranked solution thereby yields a correct semantic role assignment.

While the previous section and the current section have both integrated distributional representations into FCG constructions, the impact on the grammar is quite different. In the previous section, symbols representing substantive material in constructions were substituted by pointers to embeddings. This has rendered the constructions more general and less specific to particular input structures, as exact matches between symbols are no longer a hard constraint. In the present section, the embeddings were introduced to represent the prototypical lexical content of constructional slots and do not replace a feature that was present in the base grammar. The constructions have thereby become more specific, allowing for a more fine-grained disambiguation between possible construction application results. The integration of embeddings should thus not be seen solely as a means to make symbolic grammars more general, but it can also serve to integrate more specific information into constructions that would be considered too specific when relying on exact matching.

## 5 Distributional Representations of Grammatical Categories

In the previous sections, we have integrated pretrained GloVe embeddings in the base grammar to distributionally represent linguistic forms and prototypical slot fillers. These embeddings were trained independently from the base grammar on large amounts of text and mainly reflect the lexical content of words and phrases. In this section, we explore a different approach to integrating distributional representations in constructions. We no longer rely on externally trained embeddings, but model the similarity between grammatical categories based on the constructional slots they are compatible with. A weighted graph capturing the frequency of these slot-filler relations is built up while the grammar is being learnt from corpus data.

Let us consider the example utterance "*Try googling it for more info.*" (English Web Treebank `answers/00/20080426141111AAgPUwU_ans`). The base grammar identifies "*googling*" as a potential frame-evoking element, but holds no argument structure construction that is both compatible with the lemma `google` and the imperative transitive structure in which it appears syntactically. Consequently, no instance of the `google.01` roleset is being detected using the base grammar and no semantic roles are assigned. Importantly, the reason is not that the imperative transitive construction was not learnt during training, but that it was not learnt to be compatible with the category proper to the GOOGLE(V)-CXN.

Based on the weighted graph that captures the distribution of slot-filler categories over constructional slots, similarity between categories can be computed using the weighted cosine similarity metric. As such, slot-filler categories that are similarly distributed over constructional slots will be closer to each other than categories that rarely occur in the same constructions. In the base grammar, the category proper to the GOOGLE(V)-CXN bears a high similarity to the category proper to the DISREGARD(V)-CXN. Intuitively, this is not surprising, as both verbs are strictly transitive. If the distributions of two categories are close to each other, which means that the two categories behave similarly in the grammar, one could infer that if one category is compatible with a specific constructional slot, the other category is also likely to be compatible with it. In our example, the compatibility of the category proper to the DISREGARD(V)-CXN with the category matched by the frame-evoking element unit of the imperative transitive construction can be taken as an indication that this specific argument structure construction might also provide a correct role assignment for the GOOGLE(V)-CXN. Indeed, the imperative transitive construction here correctly assigns the 'arg1' role ('*target of search*') to "*it*". The processing of this example utterance is schematically depicted in Figure 4. The link in the categorial network between `cat-10` (GOOGLE(V)-CXN) and `cat-11` (V(V)-ARG1(NP)-CXN), which is necessary to apply the imperative transitive construction is inferred on the fly with a graph cosine similarity score of 0.3 based on the distributional similarity between `cat-10` (GOOGLE(V)-CXN) and `cat-21` (DISREGARD(V)-CXN).

## 6 Related Work

While we provide to the best of our knowledge the first fully operational and computationally implemented prototype of a symbolic construction grammar that integrates distributional representations and processing mechanisms to enhance its robustness and generality, many scholars have already addressed in one way or another the challenge of combining construction grammar with distributional semantics. Levshina and Heylen (2014) pioneered the use of distributional representations to represent the prototypical slot-fillers of constructions in a corpus-linguistic study. Hilpert and Perek (2015) and Perek (2016) have used distributional representations to track changes in the slot-fillers of constructions over time. In the same spirit, Lebani and Lenci (2018) make use of distributional representations to represent thematic roles. Rambelli et al. (2019) and Blache et al. (2024) make a case for integrating distributional representations into construction grammar and present a theoretical proposal of how distributional representations could be integrated into Sign-Based Construction Grammar to represent word forms and slots. Finally, Dunn (2017, 2024) provides a grammar induction algorithm that makes use of distributional representations to model the prototypical content of constructional slots. A related body of research is not directly concerned with construction grammar, but with the integration of formal and distributional semantics (for an overview, see Boleda and Herbelot, 2016, and other papers in the same special issue). The goal is again to combine the compositional and inferential aspects of logic-based representations with the machine learnability and lexical modelling capacities of distributional representations.

A more distantly related line of research that is concerned with both construction grammar and word embeddings investigates the linguistic capabilities of large language models from a construction grammar perspective. The goal is not to integrate symbolic and distributional approaches, but to assess to what extent distributional approaches, in

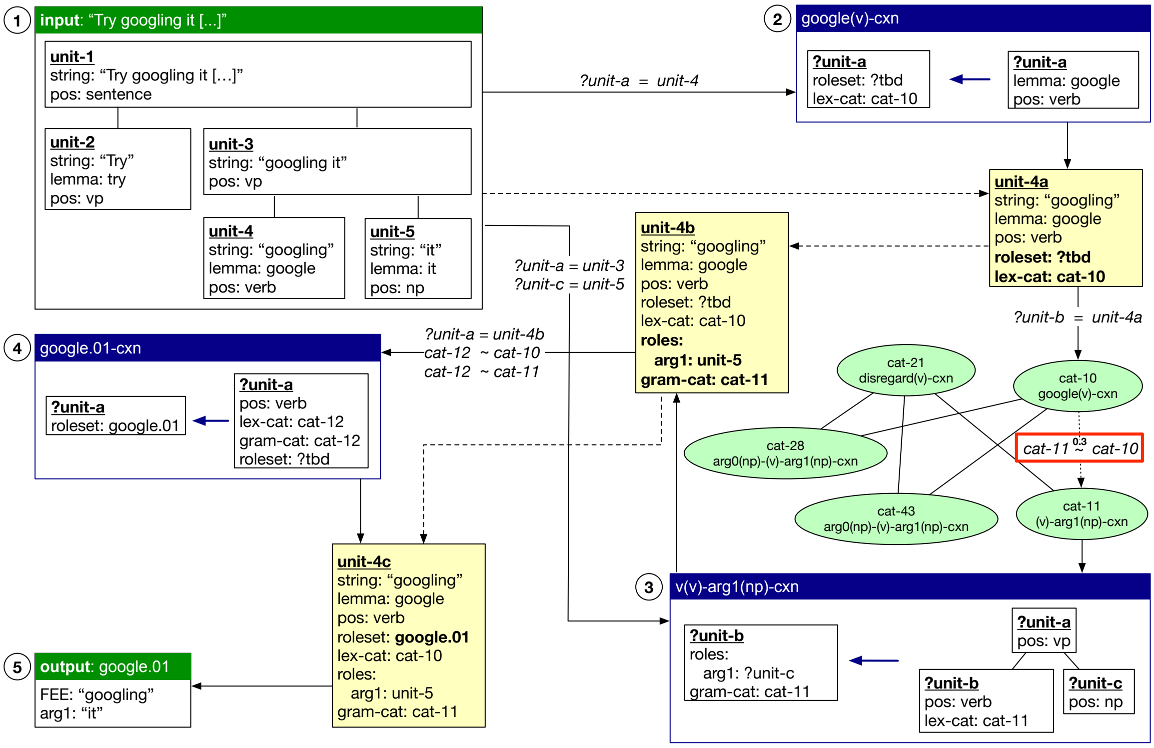


Figure 4: Schematic illustration of the integration of distributional representations of grammatical categories. The category that is proper to the GOOGLE(V)-CXN is not directly compatible with the category of the frame-evoking element slot of the imperative transitive construction (see ③). However, this categorial link is inferred on the fly based on the close distributional similarity between `cat-10` and `cat-21`.

particular large language models, capture the constructional knowledge that is typically represented symbolically in the construction grammar literature (see e.g. Tayyar Madabushi et al., 2020; Weissweiler et al., 2022, 2023; Bonial and Tayyar Madabushi, 2024; Zhou et al., 2024; Tayyar Madabushi et al., 2025).

## 7 Discussion and Conclusion

We have started from the observation that linguistic theories and models of natural language typically adopt either a symbolic or a numerical approach. At the same time, symbolic and numerical approaches are widely acknowledged to be complimentary to each other (see e.g. Boleda and Herbelot, 2016). Symbolic approaches have the advantage of supporting compositionality, interpretability and sound logic inference, whereas numerical approaches have the advantage of being more scalable, robust and easier to learn from data. The integration of symbolic and numerical approaches is however complicated by the fact that they are rooted in very different mathematical frameworks, namely formal logic versus linear algebra.

In this paper, we have explored the integration of numerical representations, in this case distributional representations of word forms, constructional slots and grammatical categories, in a symbolic computational construction grammar framework. Concretely, we have shown how such representations can be operationalised in Fluid Construction Grammar and enhance the robustness and generality of learned FCG grammars. In a first experiment, we have replaced the substantive material in the constructions of a learned, symbolic base grammar by pre-trained GloVe embeddings of the same material. By extending FCG's unification algorithms to compute cosine similarities instead of symbol equalities during the construction application process, we obtained a range of ranked construction application results in cases where there was no exact match, but a close match, between the lemma required by a construction and the one observed in the input utterance. In a second experiment, we have integrated vector representations of the prototypical lexical content of constructional slots to aid disambiguation where competing constructions could apply. The vectors were computed while the grammar was being learned, based on pre-trained GloVe embeddings of the words and phrases that were observed in the respective slots of the construction. By aggregating the cosine similarities of slots and their fillers during construction application, we again obtained a range of construction application results ranked according to their lexical fit with the applied constructions. In a third experiment, we no longer relied on externally trained embeddings, but have modelled the similarity between grammatical categories based on their observed distribution over constructional slots. This distribution was then used to create links on the fly in the categorial network that were never learnt during training.

The experiences gained while working on this initial prototype have convinced us that the future of computational construction grammar will be hybrid. Yet, further research is now needed to scale this prototype for large-scale evaluation, where the advantages of integrating distributional representations can also be shown quantitatively.

## Acknowledgements

The research reported on in this paper was funded by the F.R.S.-FNRS-FWO WEAVE project HERMES I under grant numbers T002724F (F.R.S.-FNRS) and G0AGU24N (FWO), the Flemish Government under the Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen programme, and the AI Flagship project ARIAC by DigitalWallonia4.ai.

## References

Benjamin K. Bergen and Nancy Chang. 2005. Embodied Construction Grammar in simulation-based language understanding. In Mirjam Fried and Jan-Ola Östman, editors, *Construction Grammars: Cognitive Grounding and Theoretical Extensions*, pages 147–190. John Benjamins, Amsterdam, Netherlands.

Katrien Beuls and Paul Van Eecke. 2023. Fluid Construction Grammar: State of the art and future outlook. In *Proceedings of the First International Workshop on Construction Grammars and NLP (CxGs+NLP, GURT/SyntaxFest 2023)*, pages 41–50. Association for Computational Linguistics.

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English web treebank ldc2012t13. Philadelphia, Linguistic Data Consortium.

Philippe Blache, Emmanuele Chersoni, Giulia Rambelli, and Alessandro Lenci. 2024. Composing or Not Composing? Towards Distributional Construction Grammars. *arXiv preprint arXiv:2412.07419*.

Gemma Boleda and Aurélie Herbelot. 2016. Formal Distributional Semantics: Introduction to the Special Issue. *Computational Linguistics*, 42(4):619–635.

Claire Bonial and Harish Tayyar Madabushi. 2024. Constructing understanding: on the constructional information encoded in large language models. *Language Resources and Evaluation*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis.

Jonathan Dunn. 2017. Computational learning of construction grammars. *Language and Cognition*, 9(2):254–292.

Jonathan Dunn. 2024. *Computational construction grammar: A usage-based approach*. Cambridge University Press.

Charles J. Fillmore. 1988. The mechanisms of "construction grammar". In *Annual Meeting of the Berkeley Linguistics Society*, volume 14, pages 35–55.

Martin Hilpert and Florent Perek. 2015. Meaning change in a petri dish: constructions, semantic vector spaces, and motion charts. *Linguistics Vanguard*, 1(1):339–350.

Paul Kay and Charles Fillmore. 1999. Grammatical constructions and linguistic generalizations: The what's X doing Y? construction. *Language*, 75(1):1–33.

Gianluca Lebani and Alessandro Lenci. 2018. A distributional model of verb-specific semantic roles inferences. In Thierry Poibeau and Aline Villavicencio, editors, *Language, cognition, and computational models*, pages 118–158. Cambridge University Press.

Alessandro Lenci. 2018. Distributional models of word meaning. *Annual review of Linguistics*, 4(1):151–171.

Natalia Levshina and Kris Heylen. 2014. A radically data-driven construction grammar: Experiments with dutch causative constructions. In Ronny Boogaart, Timothy Colleman, and Gijsbert Rutten, editors, *Extending the Scope of Construction Grammar*, pages 17–46. De Gruyter Mouton, Berlin, Germany.

Laura A. Michaelis. 2008. Entity and event coercion in a symbolic theory of syntax. In Jan-Ola Östman and Mirjam Fried, editors, *Construction grammars: Cognitive grounding and theoretical extensions*, pages 45–88. John Benjamins Publishing Company.

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 26 (NIPS 2013)*, pages 1–9, Red Hook.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Jeffrey Pennington, Richard Socher, and Christopher D. 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. Association for Computational Linguistics.

Florent Perek. 2016. Using distributional semantics to study syntactic productivity in diachrony: A case study. *Linguistics*, 54(1):149–188.

Giulia Rambelli, Emmanuele Chersoni, Philippe Blache, Chu-Ren Huang, and Alessandro Lenci. 2019. Distributional semantics meets Construction Grammar. Towards a unified usage-based model of grammar and meaning. In *First international workshop on designing meaning representations (DMR 2019)*.

Ivan A. Sag. 2012. Sign-based construction grammar: An informal synopsis. In Hans C. Boas and Ivan A. Sag, editors, *Sign-based construction grammar*, pages 69–202. CSLI Publications/Center for the Study of Language and Information, Stanford, CA, USA.

Luc Steels. 2004. Constructivist development of grounded construction grammar. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 9–16. Association for Computational Linguistics.

Luc Steels and Joachim De Beule. 2006. Unify and merge in Fluid Construction Grammar. In *Symbol Grounding and Beyond, Third International Workshop on the Emergence and Evolution of Linguistic Communication, EELC 2006, Rome, Italy, September 30 - October 1, 2006, Proceedings*, volume 4211 of *Lecture Notes in Computer Science*, pages 197–223. Springer.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 818–827.

Harish Tayyar Madabushi, Laurence Romain, Dagmar Divjak, and Petar Milin. 2020. CxGBERT: BERT meets construction grammar. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4020–4032. International Committee on Computational Linguistics.

Harish Tayyar Madabushi, Laurence Romain, Petar Milin, and Dagmar Divjak. 2025. Construction grammar and language models. In Mirjam Fried and Kiki Nikiforidou, editors, *The Cambridge Handbook of Construction Grammar*, pages 572–595. Cambridge University Press, Cambridge, United Kingdom.

Paul Van Eecke. 2018. *Generalisation and specialisation operators for computational construction grammar and their application in evolutionary linguistics Research*. Ph.D. thesis, Vrije Universiteit Brussel, Brussels: VUB Press.

Paul Van Eecke and Katrien Beuls. 2025. PyFCG: Fluid Construction Grammar in Python. *arXiv preprint arXiv:2505.12920*.

Remi van Trijp, Katrien Beuls, and Paul Van Eecke. 2022. The FCG Editor: An innovative environment for engineering computational construction grammars. *PLOS ONE*, 17(6):e0269708.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 6000–6010, Long Beach.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. Ontonotes release 5.0 ldc2013t19. Philadelphia, Linguistic Data Consortium.

Leonie Weissweiler, Taiqi He, Naoki Otani, David R. Mortensen, Lori Levin, and Hinrich Schütze. 2023. Construction grammar provides unique insight into neural language models. In *Proceedings of the First International Workshop on Construction Grammars and NLP (CxGs+NLP, GURT/SyntaxFest 2023)*, pages 85–95. Association for Computational Linguistics.

Leonie Weissweiler, Valentin Hofmann, Abdullatif Köksal, and Hinrich Schütze. 2022. The better your syntax, the better your semantics? Probing pretrained language models for the English comparative correlative. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10859–10882. Association for Computational Linguistics.

Shijia Zhou, Leonie Weissweiler, Taiqi He, Hinrich Schütze, David R. Mortensen, and Lori Levin. 2024. Constructions are so difficult that even large language models get them right for the wrong reasons. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 3804–3811. Association for Computational Linguistics.