`semantic-features`: A User-Friendly Tool for Studying Contextual Word Embeddings in Interpretable Semantic Spaces

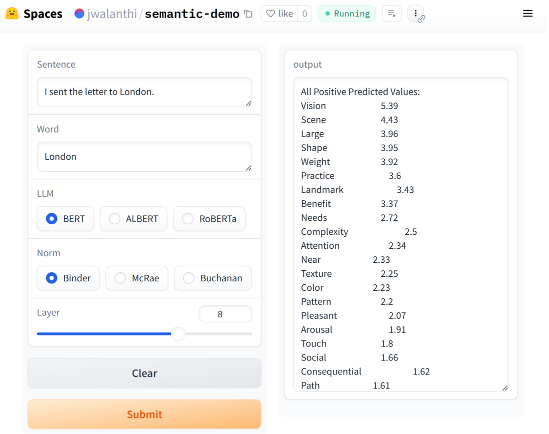
Jwalanthi Ranganathan[1] **Rohan Jha**[1] **Kanishka Misra**[1,2,★] **Kyle Mahowald**[1]
[1]The University of Texas at Austin [2]Toyota Technological Institute at Chicago
`{jwalanthi,rjha,kyle}@utexas.edu` `{kanishka}@ttic.edu`

1 Introduction

The advent of distributional semantic embeddings has enabled major progress in the computational understanding of word meaning by enabling precise statistical explorations of semantic spaces (Erk, 2009; Mikolov et al., 2013; Pennington et al., 2014). More recently, the rise of LMs have made it possible to study embeddings of words in *context*. Chronis et al. (2023) developed a method for projecting *contextual word embeddings (CWEs)* into a interpretable semantic feature space defined by one of three different semantic norms (Binder et al., 2016; Buchanan et al., 2019; McRae et al., 2005). This is achieved by training feed-forward models which map from CWEs from BERT to a vector whose values correspond to feature norms.

Our goal in this paper is twofold: first, we introduce `semantic-features`[1] as an extensible, easy-to-use library based on Chronis et al. (2023) for studying word embeddings from any LM in context. Second, we show its ease of use through an online application which researchers can use without additional programming. We demonstrate these tools with a linguistic experiment that uses this method to measure the contextual effect of the choice of dative construction (prepositional or double object) on the semantic interpretation of utterances.

The dative construction has been of particular interests to theoretical (Goldberg, 1995; Hovav and Levin, 2008; Beavers, 2011) and computational linguists (Bresnan, 2007; Hawkins et al., 2020; Liu and Wulff, 2023; Jumelet et al., 2024; Misra and Kim, 2024; Yao et al., 2025). This is primarily due to its several interesting properties such as its participation in alternation behavior (Levin, 1993), flexible interpretation of the event it describes—caused motion vs. caused possession (Goldberg, 1995; Hovav and Levin, 2008; Beavers, 2011), and interesting feature specific preferences that humans demonstrate while choosing between two dative constructions during production (Bresnan, 2007).



Sentence: I sent the letter to London.

Word: London

LLM: ● BERT ○ ALBERT ○ RoBERTa

Norm: ● Binder ○ McRae ○ Buchanan

Layer: 8

output

```
All Positive Predicted Values:
Vision          5.39
Scene           4.43
Large           3.96
Shape           3.95
Weight          3.92
Practice          3.6
Landmark          3.43
Benefit         3.37
Needs           2.72
Complexity          2.5
Attention         2.34
Near          2.33
Texture         2.25
Color         2.23
Pattern         2.2
Pleasant          2.07
Arousal           1.91
Touch         1.8
Social          1.66
Consequential           1.62
Path          1.61
```

Figure 1: Interactive Demo in Use

Our case study focuses on the semantics of the arguments of the dative construction—in particular its *recipient* argument (Beavers, 2011; Petty et al., 2022). Specifically, we hypothesize that "London" in "I sent **London** the letter." (*double object*; DO) should be more likely to be interpreted as an animate referent (e.g., as the name of a person) than in "I sent the letter to **London**." (*prepositional object*; PO) This is because the DO dative is more canonically associated with possession transfer events, which constrains the recipient to be animate (Beavers, 2011). The PO dative, on the other hand, is associated with both possession transfer and 'caused-motion' (Goldberg, 1995) and allows for inanimate recipients. We test whether LMs learn this distinction by projecting the embedded representation from the token "London" into a more interpretable semantic space and analyze it for animate vs. inanimate features. We include a full demonstration of how to easily obtain such measures from models that have already

[★] Work partly done at UT-Austin before joining TTIC.

[1] `semantic-features` is available at `https://github.com/jwalanthi/semantic-features`

been trained, in addition to describing our full system for training projections from scratch.

2 semantic-features

Our extensible system for training models and analyzing embeddings performs three main tasks: embedding extraction, model training, and hyperparameter tuning. Below, we summarize our methodology; more details can be found in the README.

Embedding Extraction The first step is preparing the CWEs which serve as the 'source' for the training data. Given a (user-provided) corpus and an LM whose weights/embeddings are accessible, semantic-features extracts an embedding for each word in the corpus using minicons (Misra, 2022). We average the embeddings across all contexts to obtain one vector per word, as in Chronis et al. (2023). While any LM can theoretically serve as the source for word embeddings, autoregressive LMs like GPT-2 are not well-suited for this application because their embeddings only capture left-context for a given word.[2]

Model training All models use a multi-layer perceptron (MLP) to perform feature prediction. All hyperparameters can be user-specified except for the MLP architecture. While Chronis et al. (2023) experimented with other architectures, we choose MLPs to maintain a fully neural system end-to-end. Models are trained with a 80-20 train-validation split, and loss is calculated as mean-squared error between the predicted vector and the ground-truth feature-norm vector.

Hyperparameter tuning Our system allows for hyperparameter tuning by using optuna (Akiba et al., 2019). We specifically use the TPESampler module, which searches for the combination of hyperparameters which minimizes validation loss using a Tree-Structured Parzen Estimator algorithm. optuna searches for the optimal values for hidden size, batch size, and learning rate over a specified set of ranges in Table 3. If enabled, the MedianPruner is used to determine which trials to prune. After running 100 trials, the model with the lowest validation loss is saved.

Interactive Demo An interactive demonstration of a selection of models trained using semantic-features is available on HuggingFace Spaces as a Gradio app,[3] shown in Figure 1. Users can retrieve a model which maps from the CWE of a user-specified word in context from any layer of BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), or ALBERT (Lan et al., 2020) to any of the three semantic feature spaces used by Chronis et al. (2023).[4] The models were trained using the British National Corpus as the source text. For each word that has a pre-defined feature vector, semantic-features extracts the embeddings in each context provided by BNC, averaging the embeddings per word across all contexts. This serves as the source for training, and the feature vector itself serves as the target. Further training details, including hyperparameter specification and GPU training hours, are provided in Appendix B. The output of the demo is a list of the predicted features sorted greatest to least.

3 A Small Case Study on Recipient Semantics in Dative Constructions

Using the tools developed in the previous section, we ask if LMs are sensitive to context-dependent semantics in linguistic constructions. Consider the dative alternation: some ditransitive verbs can take two different argument structures. The first is the *double object (DO)* construction and the second is *prepositional object (PO)* construction.

(1) a. I sent **London** the letter. *DO*
b. I sent the letter to **London**. *PO*

While both are near synonymous, they apply different contextual constraints on their arguments. For instance, in the PO, *London* takes on its "standard" definition as an inanimate place/location, but in the DO, it seems that *London* is an animate recipient (Beavers, 2011; Hovav and Levin, 2008). To what extend do LMs learn this distinction? To test this, we project embeddings from LMs to the Binder features (Binder et al., 2016) space. We choose the Binder Norms here specifically because each feature has a concrete definition provided by the researchers, which can allow for finer grained personhood vs. place-hood distinction. We use these definitions (reproduced in Table 1) to identify Binder features which capture place-hood (Landmark and

[2] As a note, while semantic-demo provides the ability to train these MLPs, it does not provide the raw training data itself. Users must obtain corpus data and feature norm data from their respective sources.

[3] Available at https://huggingface.co/spaces/jwalanthi/semantic-demo

[4] While these are the models we currently focus on, in principle this can be applied on any masked language model.

Feature	Definition
Biomotion	showing movement like that of a living thing
Body	having human or human-like body parts
Human	having human or human-like intentions, plans, or goals
Face	having a human or human-like face
Speech	someone or something that talks
Landmark	having a fixed location, as on a map
Scene	bringing to mind a particular setting or physical location

Table 1: Feature definitions from Binder et al. (2016).

Feature	DO	PO
Biomotion	**1.19**	0.43
Body	**1.00**	0.26
Human	**0.89**	0.48
Face	**0.71**	0.19
Speech	**0.68**	0.13
Landmark	1.83	**3.43**
Scene	2.59	**4.43**

Table 2: Relevant Binder features predicted for "London" in (1) using CWEs from BERT layer 8. The PO construction lends itself more towards "location" features, and the DO more towards animate features.

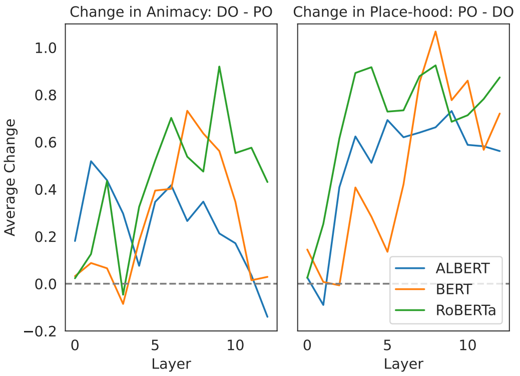


Figure 2: For each layer of an LM, we extract the CWE and project it into Binder space. **Left:** we measure the average change across the test sentences in Person features from PO to DO. The positive values indicate that recipients in the DO are found to be more animate. **Right:** we measure the average change across test sentences in Place features from DO to PO. Here, the positive values indicate that the recipients were found to be more place-like in the PO.

Scene) and person-hood (Biomotion, Body, Human, Face, and Speech) to reflect the two possible salient readings. Higher values for a Binder feature from the projected embedding is taken to mean greater activation of the specific feature. We choose features which capture person-hood and place-hood distinctively, not those which are applicable for both readings. For example, the Vision feature, which is defined as "something that you can easily see," can be activated in both contexts, and is therefore not included in either category. We then extract the embeddings for the recipient word each layer of the LM in each context and project them to the Binder space, observing changes in the relevant features. Table 2 shows an example set of predictions for (1) using BERT layer 8. We see that, consistent with our predictions, "London" is construed as more person-like in the DO and more place-like in the PO.

To test this phenomenon more robustly, we use a method similar to the experiment for studying grammatical roles in Chronis et al. (2023), which requires a balanced dataset of *DO* and *PO* sentences. While the dataset provided by Hawkins et al. (2020) is balanced in terms of the two constructions, it is not well-suited to our needs because the variation in recipient animacy is not focused on the place-like versus animate distinction observed in (1). Instead, we generate 450 alternating pairs in which the recipient is interpreted by a human evaluator to be a person in the DO and a place in the PO. We do this by querying ChatGPT to come up with proper nouns that can be interpreted as places or people, ending up with 15 different such names, all of which were manually checked. This included names of states, as in "Dakota", and names of countries, as in "Jordan", in addition to names of cities, as in "London". We then paired them with 6 different alternating verbs (lemma: send, mail, order, bring, fax, deliver) along with a host of corresponding indirect objects which could also be plausibly received by a place or person. Finally, we choose from five different agents (names), leading to our 450 pairs of sentences, each of the form `[agent]` `[verb]`$_{past}$ `[recipient]` `[theme]` for DO and `[agent]` `[verb]`$_{past}$ `[theme]` `to` `[recipient]` for PO. We project the embeddings of the recipients in context from BERT, RoBERTa, and ALBERT to the Binder feature space and average across construction (DO or PO) and feature set (Person or Place). Fig. 2 shows the average change in feature values for person-hood features vs. place-hood features across the alternants of the dative construction. That is, a value of 0.75 in the animacy panel (left) suggests that the average difference in the activation value of the recipient's animacy features in the DO and PO constructions was 0.75 units on a scale of 0 to 6 (as provided by Binder et al. (2016)) with positive values indicating "more animate in DO than in PO." Similar interpretation (though in the reversed direction) can be made for the right panel, which focuses on place-hood change when switching from DO to PO.

Results As expected, almost all of the models predict an increase in animacy in the DO compared to the PO and an increase in place-hood in the PO compared to the DO (Figure 2) across most layers. There are some exceptions where the change in person-hood/animacy features is in the opposite direction, though these are in the tiny minority (i.e., a total of 3 times out of a total 36 possible model and layer combination). Corroborating with Chronis et al. (2023), we observe particularly high activation-change of the relevant features in layers 6–9 as opposed to the final layer, suggesting possible concentration of semantic sensitivity in those layers. Overall, this suggests that the contextually sensitive distributional semantic embeddings of LMs capture subtle changes in semantic interpretation of different related-constructions.

4 Conclusion

Our hope is that both the complete `semantic-features` library for projecting CWEs into semantic spaces and the online demo will facilitate running linguistically informative experiments using contextual word embeddings.

Acknowledgments

Thanks to the three anonymous reviewers for their feedback. We also thank Katrin Erk and Gabriella Chronis for their comments on an earlier draft and the research project in general. $(\text{KM})^2$ acknowledge funding from NSF Grant 2104995 awarded to Kyle Mahowald.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework.

John Beavers. 2011. An aspectual analysis of ditransitive verbs of caused possession in english. *Journal of semantics*, 28(1):1–54.

Jeffrey R. Binder, Lisa L. Conant, Colin J. Humphries, Leonardo Fernandino, Stephen B. Simons, Mario Aguilar, and Rutvik H. Desai. 2016. Toward a brain-based componential semantic representation. *Cognitive Neuropsychology*, 33(3):130–174.

Joan Bresnan. 2007. Predicting the dative alternation. *Cognitive foundations of interpretation/Royal Netherlands Academy of Science*.

Erin M. Buchanan, K. D. Valentine, and Nicholas P. Maxwell. 2019. English semantic feature production norms: An extended database of 4436 concepts. *Behavior Research Methods*, 51:1849–1863.

Gabriella Chronis, Kyle Mahowald, and Katrin Erk. 2023. A method for studying semantic construal in grammatical constructions with interpretable contextual embedding spaces. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 242–261, Toronto, Canada. Association for Computational Linguistics.

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, Minnesota. Association for Computational Linguistics.

Katrin Erk. 2009. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 57–65, Boulder, Colorado. Association for Computational Linguistics.

Adele E Goldberg. 1995. *Constructions: A Construction Grammar Approach to Argument Structure*. University of Chicago Press.

Robert Hawkins, Takateru Yamakoshi, Thomas Griffiths, and Adele Goldberg. 2020. Investigating representations of verb bias in neural language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4653–4663, Online. Association for Computational Linguistics.

Malka Rappaport Hovav and Beth Levin. 2008. The english dative alternation: The case for verb sensitivity. *Journal of linguistics*, 44(1):129–167.

Jaap Jumelet, Willem Zuidema, and Arabella Sinclair. 2024. Do language models exhibit human-like structural priming effects? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14727–14742, Bangkok, Thailand. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*.

Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019.

RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*. ArXiv: 1907.11692.

Zoey Liu and Stefanie Wulff. 2023. The development of dependency length minimization in early child language: A case study of the dative alternation. In *Proceedings of the Seventh International Conference on Dependency Linguistics (Depling, GURT/SyntaxFest 2023)*, pages 1–8, Washington, D.C. Association for Computational Linguistics.

Ken McRae, George S Cree, Mark S Seidenberg, and Chris McNorgan. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior research methods*, 37(4):547–559.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv e-prints*, page arXiv:1301.3781.

Kanishka Misra. 2022. minicons: Enabling flexible behavioral and representational analyses of transformer language models. *arXiv:2203.13112*.

Kanishka Misra and Najoung Kim. 2024. Generating novel experimental hypotheses from language models: A case study on cross-dative generalization. *arXiv preprint arXiv:2408.05086*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP 2014*, pages 1532–1543.

Jackson Petty, Michael Wilson, and Robert Frank. 2022. Do language models learn position-role mappings? In *Proceedings of the 46th annual Boston University Conference on Language Development*.

Qing Yao, Kanishka Misra, Leonie Weissweiler, and Kyle Mahowald. 2025. Both direct and indirect evidence contribute to dative alternation preferences in language models. *arXiv preprint arXiv:2503.20850*.

A Hyperparameters

The following search ranges are used by `optuna` for hidden size, batch size, and learning rate when enabled.

Hyperparameter	**Lower Limit**	**Upper Limit**
Hidden Size	*min*	min(2**min*, *max*)
Batch size	16	128
Learning Rate	10^{-6}	1

Table 3: Search ranges for optimization, where *min* denotes the minimum between the length of the embedding and length of the feature vector and *max* denotes the maximum between the two values

B Demo Models

All 117 models available through the Gradio app have 2 layers with 50% dropout, and early stopping after 6 epochs of non-decreasing validation loss. The maximum epoch limit was set to 100, though in reality, the best performing models finished training after 40-60 epochs. Hyperparameter tuning was used for hidden size, batch size, and learning rate, and pruning was not enabled. For the Buchanan models, the raw feature labels were not used, and the normalized feature values were used. In total, training all 117 models took 25 GPU hours, including those which were discarded in the process of optimization. Models were trained using an NVIDIA A40 GPU.