Mining Meaning: Semantic Similarity and the Analysis of Political Text

Supplementary Materials

Replication Code

All replication code will for the figures in the main text and all figures and tables in the online appendix will be deposited in accoradance with the journal's guidelines.

Encoding Models

A review of encoding models is beyond the scope of this paper. Nevertheless, I clarify a few points here to provide context for the sentence-encoder presented in the main text.

Word-Encoders

Initially, encoding models operated on words, transforming each word or token into high-dimensional vectors in such a way that their proximity in the embedding space reflected some set of shared semantic properties, e.g., Word2Vec (Mikolov et al. 2013). Political scientists have used word-level embeddings to estimate the political stances of Twitter users (Mebane Jr et al. 2018); to infer ideological placement of parliamentary members (Rheault and Cochrane 2020); to assess the negativity in parliamentary speeches (Rudkowsky et al. 2018); and to understand the influence of partisanship on word usage (Rodriguez, Spirling, and Stewart 2021).

Word-level embeddings have their limitations, however. For example, word-level models have difficulty handling negation, an essential aspect of deriving meaning from a sentence (X. Zhu, Li, and de Melo 2018). Word-encoding models also generate static embeddings that are a function of the data on which they are trained. Yet human language is flexible and word meaning is not static. Human judgments of semantic similarity accommodate this fluidity; word-level embeddings do not. Rodriguez, Spirling, and Stewart (2021) illustrate the challenge of static embeddings in interpreting the meaning of the word "society" without sufficient context.

Sentence-Encoders

Sentence-encoders try to close the gap between a machine's representation of language meaning and a human's. In particular, sentence-encoders try to capture enough of the semantic properties of text *strings* that their representational space encodes meaning in a way that humans recognize. While the models themselves are opaque, probes have shown that they can encode syntactic and semantic information, as well as information about entity types (Rogers, Kovaleva, and Rumshisky 2021). These models are evaluated by how well they perform against the benchmark of human judgment across a range of tasks and, secondarily, in their computational efficiency (e.g., Conneau and Kiela 2018). There have been several step-changes in sentence-encoder performance on both of these dimensions (for a review, see Luitse and Denkena 2021). For the purposes of this article, I simply note that the state-of-the-art in sentence-encoder models share a particular design feature in that they are Transformer-based[1], and that the latest step-change in computational efficiency came when Reimers and Gurevych (2019) introduced the Sentence-Bidirectional Encoder Representations from Transformers (S-BERT). Reimers and Gurevych (2019)'s innovation has generated an entire family of language models implemented in Python that are freely available from https://huggingface.co/sentence-transformers.

The challenge for sentence-encoders is to learn how to represent (snippets of) human language in a representational space that produces relationships that align well with human estimates of those same relationships. The advantage of these models for social scientists – and those who are new to computational tools in particular – is that all of this training is done *before* the researcher interacts with the model. Nevertheless, it is worth noting *why* these models can be used out-of-the-box.

Building a model of human language is an immense task requiring vast amounts of training

[1]The Transformer architecture was introduced in Vaswani et al. (2017) and demonstrated both a performance and computational resource advantage over the prior generation of deep learning models, such as Recurrent Neural Networks (RNNs) and Long-Term Short-Term Memory (LSTM) models.

input, as well as a task or tasks against which to evaluate the model's performance (e.g., Song et al. 2020). While the details vary, training input is often billions of text snippets, either from structured corpora (e.g., Y. Zhu et al. (2015)'s BookCorpus) or from unstructured corpora (e.g., the Common Crawl). The tasks on which they are trained also vary, but include predicting masked tokens or determining the probability of a subset of permuted tokens (Tan 2020). At this stage, these "base" language models can represent text in ways that retain semantic structure and word-use context, but the representational space is not optimized to reflect any particular kind of relationship *between* the texts.

These representations generated by these models can be further refined, however, by training on specific tasks (Hill, Cho, and Korhonen 2016). Reimers and Gurevych (2019) note that these tasks can include assessments of semantic similarity, multilingual translations, or even returning answers to questions. The flexibility of these "base" models demonstrates that they are good solutions to the transfer learning problem, which refers to the difficulty many models have when performing outside the scope of their training task(s) (Azunre 2021). Thus, although these "base" language models were only trained to predict missing or permuted tokens, the representations they generate still contain rich semantic information that is more generally useful.

The next stage of pre-training that a language model can undergo might be thought of as the point at which the model learns "what goes with what." This is the stage at which the model's representational space is optimized to reflect one (or more) types of relationships between texts. In their initial formulation, Reimers and Gurevych (2019) noted that these models could capture a variety of different meaning-based linguistic relationships. For example, a model could be optimized for taking in a question and returning an answer. In that case, the base model would be augmented by a last layer trained on corpora containing pairs of questions and answers from WikiAnswers, Stack Exchange, or a number of other sources.[2] Alternatively, a model could be optimized for identifying most-similar sentences

[2]See Hugging Face's Q&A models for examples.

(i.e., semantic textual similarity). In that case, the last layer of the base model is trained on a corpus of sentences annotated for an indicator of similarity, such as Stanford's Natural Language Inference dataset (Bowman et al. 2015). It is also possible to train models on multiple tasks, generating representations that can be used for different objectives (semantic search and semantic similarity, for example).

Additional layers can also be added to fine-tune a model's representational space. Often these layers are added based on benchmark tasks, such as the Semantic Textual Similarity Benchmark dataset (Cer et al. 2017). On the one hand, fine-tuning reduces the generality of the model's representations. On the other, it optimizes those representations for a clearly defined task.

Model Specifications

Language models are under constant development. Throughout this paper, I use the sentence-encoder `stsb-mpnet-base-v2`, which is optimized for semantic textual similarity (Reimers and Gurevych 2019). The base language model is Microsoft's MPNet (Song et al. 2020) and it was trained on semantic similarity using the Stanford Natural Language Inference dataset (Bowman et al. 2015), then fine-tuned using the STS benchmark dataset (Cer et al. 2017).

`stsb-mpnet-base-v2` embeds sentences into a 768-dimensional dense vector space and encodes up to 75 tokens as its default. While the model can be adjusted to accommodate longer strings, my own testing suggests that doing so reduces the model's performance. I suspect that forcing the model to vectorize texts that are much more complex than those on which it was trained creates degrades the model's representations somewhat.

Benchmarking

The most commonly used measure of the quality of a model's assessments is the correlation between the model's similarity ratings and human similarity ratings. The Short Text

Semantic Similarity benchmark dataset (O'Shea, Bandar, and Crockett 2014) provides a set of human-coded similarity judgments that is ideally suited to evaluate the STS-trained model's performance. The dataset includes 64 sentence-pairs developed specifically for the purpose of assessing NLP model performance in semantic similarity judgments.[3] Each sentence-pair was rated by 64 human subjects for similarity in meaning on a scale from 0 to 4. O'Shea, Bandar, and Crockett (2014) reported the mean similarity rating for every sentence-pair, as well as the performance of several NLP models.

I entered the 64 sentence-pairs into each of three sentence-encoders and retrieved their embeddings. The three encoders were the STS-trained model (`stsb-mpnet-base-v2`), an all-purpose model (`all-mpnet-base-v2`) using the same base language model (MPNet), and an averaged word-level embeddings model (`average-word-embeddings-glove6B300d`). For each sentence-pair, I then calculated the cosine similarity between the two embedding vectors produced by a given model.

Reimers, Beyer, and Gurevych (2016) show that Spearman's rank correlation is more appropriate than Pearson's correlation for comparing semantic similarity measures. However, O'Shea, Bandar, and Crockett (2014) only reported results for their LSA model using Pearson's r. Thus, Table A1 reports Pearson's correlation for the sake of comparability. For the three encoding models shown in Table A1, the choice of Spearman's ρ does not meaningfully change the results: STS-trained model $\rho = 0.915$; all-purpose model $\rho = 0.899$; GloVe model $\rho = 0.821$.

To put these correlations in perspective, it is worth considering the range of human inter-rater correlations reported by the Comparative Manifestos Project and collected in Mikhaylov, Laver, and Benoit (2012) (Table 1). These correlations range from 0.70 (within a group of nine coders on their second CMP contract) to 0.88 (9 training coders versus the master coding answers, on their second attempt). Both sentence-encoders perform well against these reference points, though CMP rater tasks are for categorization, not semantic similarity

[3]None of these sentence-pairs were used in training or fine-tuning `stsb-mpnet-base-v2`.

judgments per se.

Table A 1: Benchmarking Task for Similarity Estimates

Model	Type	Task	r vs. Human Ratings
stsb-mpnet-base-v2 (Hugging Face)	Sentence encoder	STS-trained	0.912
all-mpnet-base-v2 (Hugging Face)	Sentence encoder	All-purpose	0.907
Average GloVe embeddings	Word encoder	None	0.802
O'Shea et al. (2014)'s LSA model	Latent Semantic Analysis	None	0.693

A second type of benchmarking task asks whether the STS-trained model can solve three recognized "hard problems" in NLP. First, the model must be able to account for negation (i.e., "X" and "not X" must be far apart in the embedding space). Second, the model must be able to capture shared meaning between two sentences even if they do not share words. Third, the model should pass the test of superficial similarity and detect a significant difference when only one word has been altered, drastically changing the sentence's meaning.

Despite their similar scores in the human rater correlation task, Figure A1 shows why an STS-trained model (middle bar) is more promising than the all-purpose model (left bar) and far better than a word-level encoder (right bar). The values in the figure result from running the reference sentence and four target sentences through each encoding model, retrieving the vectors of embeddings, and then calculating the cosine similarity for each sentence-pair.

As the figure shows, all models retain extremely high similarity scores for the first panel, though the word-level model stumbles by identifying the sentences as identical. The STS-trained model handles negation best (second panel) and the word-level model fails. The

STS-trained model also correctly assigns a relatively high degree of similarity between two sentences that share no words, but some meaning (third panel), and passes the test of superficial similarity (fourth panel).

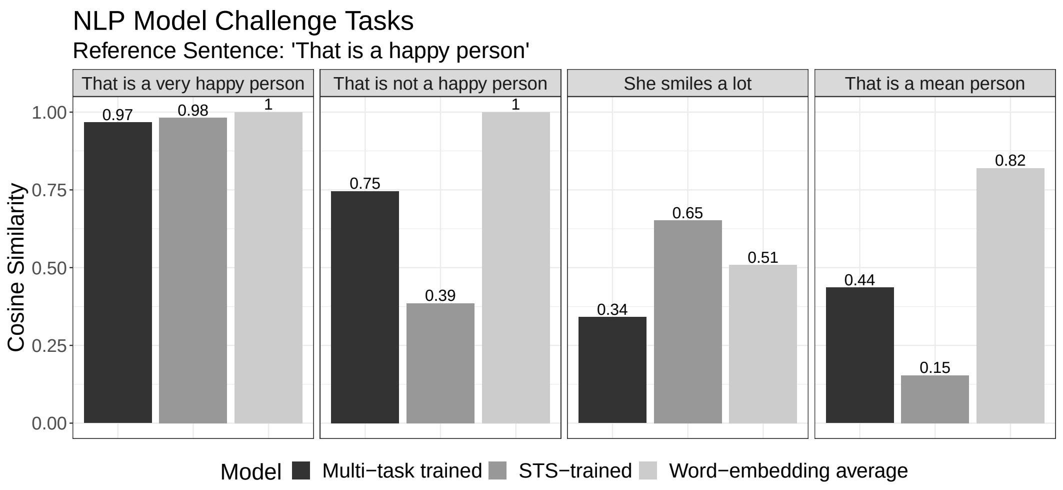


Figure A 1: Sentence-Embedding Model Performance

In sum, the STS-trained model avoids established NLP pitfalls, which gives us greater confidence in its similarity judgments.

Relative Semantic Similarity **Derivation**

If Category $\mathbf{X}$ is defined as a set of texts (sentences, paragraphs, or documents) whose embeddings are represented by vectors $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}$, and the cosine similarity of any two vectors $\mathbf{x_1}, \mathbf{x_2}$ is denoted by $S_{x1,x2}$, then the pairwise similarities of all texts within Category $\mathbf{X}$, i.e., Sim_X, can be represented in a square, symmetric matrix of cosine similarities:

$$Sim_X = \begin{bmatrix} S_{x1,x1} & S_{x1,x2} & \cdots & S_{x1,xn} \\ S_{x2,x1} & S_{x2,x2} & \cdots & S_{x2,xn} \\ \vdots & \vdots & \ddots & \vdots \\ S_{xn,x1} & S_{xn,x2} & \cdots & S_{xn,xn} \end{bmatrix} \tag{1}$$

Within-category similarity, W_X, is given by the mean of the row-wise averages of Sim_X, excluding the identity values. Thus, where i and j index rows and columns, respectively, so that s_{ij} denotes the elements of Sim_X:

$$W_X = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} s_{ij} \tag{2}$$

If Category $\mathbf{Y}$ is defined as a set of sentence-length texts whose embeddings are represented by vectors $\mathbf{y_1}, \mathbf{y_2}, ..., \mathbf{y_n}$, then the pairwise similarity for all texts in Categories $\mathbf{X}$ and $\mathbf{Y}$, Sim_{XY}, can be represented in a symmetric matrix of cosine similarities as well:

$$Sim_{XY} = \begin{bmatrix} S_{x1,y1} & S_{x1,y2} & \cdots & S_{x1,ym} \\ S_{x2,y1} & S_{x2,y2} & \cdots & S_{x2,ym} \\ \vdots & \vdots & \ddots & \vdots \\ S_{xn,y1} & S_{xn,y2} & \cdots & S_{xn,ym} \end{bmatrix} \tag{3}$$

Between-category similarity, B_{XY}, is given by the mean of the row averages of Sim_{XY}, which is an n row by m column matrix.

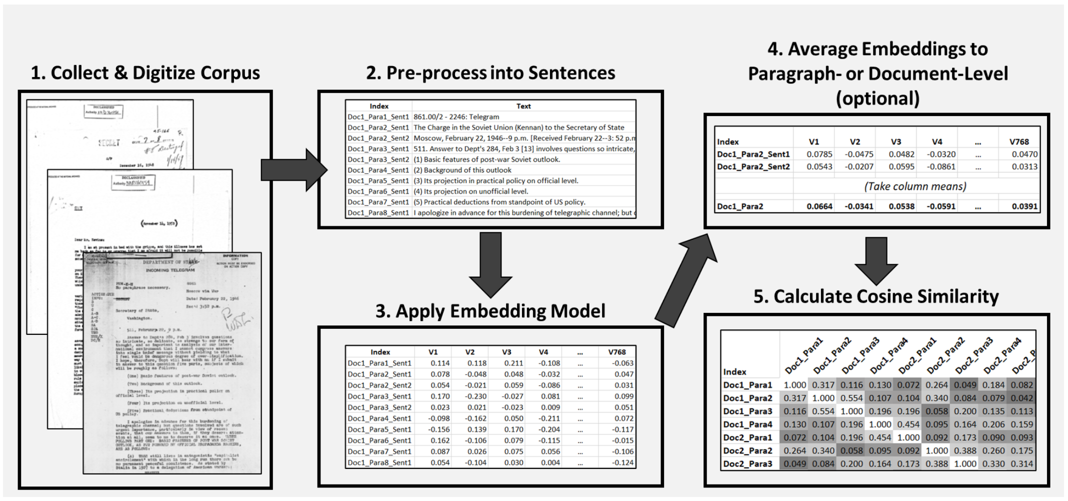
$$B_{XY} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{m} \sum_{j=1}^{m} s_{ij} \tag{4}$$

The *relative semantic similarity* of Category $\mathbf{X}$ with respect to Category $\mathbf{Y}$ is thus the difference score:

$$WB_{XY} = W_X - B_{XY} \tag{5}$$

Workflow and Example

Figure A2 illustrates the workflow for calculating semantic similarity across a corpus. Steps 2-5 can be carried out entirely in `R` (R Core Team 2016) with an interface to a Python installation (Van Rossum and Drake 2009), such as the `reticulate` package (Ushey, Allaire, and Tang 2022).


1. Collect & Digitize Corpus
2. Pre-process into Sentences
3. Apply Embedding Model
4. Average Embeddings to Paragraph- or Document-Level (optional)
5. Calculate Cosine Similarity

Index	Text
Doc1_Para1_Sent1	861.00/2 - 2246: Telegram
Doc1_Para2_Sent1	The Charge in the Soviet Union (Kennan) to the Secretary of State
Doc1_Para2_Sent2	Moscow, February 22, 1946--9 p.m. [Received February 22--3: 52 p.m
Doc1_Para3_Sent1	511. Answer to Dept's 284, Feb 3 [13] involves questions so intricate,
Doc1_Para3_Sent2	(1) Basic features of post-war Soviet outlook.
Doc1_Para4_Sent1	(2) Background of this outlook
Doc1_Para5_Sent1	(3) Its projection in practical policy on official level.
Doc1_Para6_Sent1	(4) Its projection on unofficial level.
Doc1_Para7_Sent1	(5) Practical deductions from standpoint of US policy.
Doc1_Para8_Sent1	I apologize in advance for this burdening of telegraphic channel; but

Index	V1	V2	V3	V4	...	V768
Doc1_Para1_Sent1	0.114	0.118	0.211	-0.108	...	-0.063
Doc1_Para2_Sent1	0.078	-0.048	0.048	-0.032	...	0.047
Doc1_Para2_Sent2	0.054	-0.021	0.059	-0.086	...	0.031
Doc1_Para3_Sent1	0.170	-0.230	-0.027	0.081	...	0.099
Doc1_Para3_Sent2	0.023	0.021	-0.023	0.009	...	0.051
Doc1_Para4_Sent1	-0.098	-0.162	0.050	-0.211	...	0.072
Doc1_Para5_Sent1	-0.156	0.139	0.170	-0.204	...	-0.117
Doc1_Para6_Sent1	0.162	-0.106	0.079	-0.115	...	-0.015
Doc1_Para7_Sent1	0.087	0.026	0.075	0.056	...	-0.106
Doc1_Para8_Sent1	0.054	-0.104	0.030	0.004	...	-0.124

Index	V1	V2	V3	V4	...	V768
Doc1_Para2_Sent1	0.0785	-0.0475	0.0482	-0.0320	...	0.0470
Doc1_Para2_Sent2	0.0543	-0.0207	0.0595	-0.0861	...	0.0313
	(Take column means)					
Doc1_Para2	**0.0664**	**-0.0341**	**0.0538**	**-0.0591**	**...**	**0.0391**

Index	Doc1_Para1	Doc1_Para2	Doc1_Para3	Doc1_Para4	Doc2_Para1	Doc2_Para2	Doc2_Para3	Doc2_Para4	Doc2_P
Doc1_Para1	1.000	0.317	0.116	0.130	0.072	0.264	0.049	0.184	0.082
Doc1_Para2	0.317	1.000	0.554	0.107	0.104	0.340	0.084	0.079	0.042
Doc1_Para3	0.116	0.554	1.000	0.196	0.196	0.058	0.200	0.135	0.113
Doc1_Para4	0.130	0.107	0.196	1.000	0.454	0.095	0.164	0.206	0.159
Doc2_Para1	0.072	0.104	0.196	0.454	1.000	0.092	0.173	0.090	0.093
Doc2_Para2	0.264	0.340	0.058	0.095	0.092	1.000	0.388	0.260	0.175
Doc2_Para3	0.049	0.084	0.200	0.164	0.173	0.388	1.000	0.330	0.314

Figure A 2: Raw Text to Similarity Workflow

Example R code for a minimal working example using this workflow will be available with the replication materials. The MWE code includes guidance for Python requirements (version 3 or higher) and necessary modules.

The Combatting Communism Corpus: Coding Examples

The CC Corpus contains 12,263 hand-coded paragraphs. Table A2 shows the distribution of category assignments across these paragraphs.

Table A 2: CC Corpus Category Codings (Paragraph Level)

Category	N	Share (%)
0	11,020	89.86
1	122	0.99
2	470	3.83
3	368	3
Mixed	283	2.31

Table A3 shows the results of the permutation inference procedure used in Figure 1 of the main text. The implied p-value refers to the probability of observing a value for WB_{XY} that is at least as large as the true value calculated from the original data when there is no relationship between category assignments and semantic similarity. In the case of the hand-coding scheme, there is less than a 1 in 500 chance that the true WB_{XY} values are false positives.

Table A 3: CC Corpus Category Permutation Results

WB_XY	Permutations	N Above Observed	Implied P-Value
Category 1 v. 2	500	0	<0.002
Category 1 v. 3	500	0	<0.002
Category 2 v. 1	500	0	<0.002
Category 2 v. 3	500	0	<0.002
Category 3 v. 1	500	0	<0.002
Category 3 v. 2	500	0	<0.002

Table A4 shows the results of the permutation inference procedure used in Figure 2 of the main text. In the case of secrecy status, there is a relatively high probability ($p > 0.05$) that – in all cases – the differences observed between documents with different audiences are false positives.

Table A 4: CC Corpus Secrecy Status Permutation Results

WB_XY	Permutations	N Above Observed	Implied P-Value
Never Class. v. Top Secret	500	40	0.08
Never Class. v. OTR	500	120	0.24
OTR v. Top Secret	500	169	0.34
OTR v. Never Class.	500	222	0.44
Top Secret v. Never Class.	500	227	0.45
Top Secret v. OTR	500	261	0.52

Table A5 provides example texts for each category.

Table A 5: CC Corpus Category Examples

Category	Example	Author
0	Second, in the economic field, not only must the United States remain strong itself, but it must realize that it is a pressing matter - again in self-preservation - to do its utmost to make and keep the entire free world strong. This means that our economic policies must be realistic and vigorous. We cannot afford outmoded slogans. We must produce goods and we must ship goods abroad, and that means granting credits and receiving imports. This we must face squarely and act upon it.	Dean Acheson, speech, 4/18/47, paragraph 56
1	We in the United States follow our tradition in continually seeking the truth, not only about ourselves, but also about possible enemies – particularly with regard to relative military strength. This knowledge is essential if we are to assess our chances of survival, because the Communist leaders reiterate they cannot exist on the same planet with the freedom loving Democratic nations.	W. Stuart Symington, speech, 4/12/50, paragraph 45

Category	Example	Author
2	In foreign countries Communists will, as a rule, work toward destruction of all forms of personal independence, economic, political or moral. Their system can handle only individuals who have been brought into complete dependence on higher power. Thus, persons who are financially independent–such as individual businessmen, estate owners, successful farmers, artisans and all those who exercise local leadership or have local prestige, such as popular local clergymen or political figures, are anathema. It is not by chance that even in USSR local officials are kept constantly on move from one job to another, to prevent their taking root in the local community.	George F. Kennan, memo, 2/22/1946, paragraph 62
3	But the strength of Communism is also its weakness and worst enemy. The Communists, by fanatically following the Marxist-Leninist philosophy, have reversed life - they are attempting to create through destruction, to gain victories by glorifying defeat. To build a bright new world, they are degrading man, taking from him, idea by idea, thought by thought, attitude by attitude, the values of independent reasoning and truth seeking. The very ingredients of eventual success, intelligence, Judgment and moral reserve, are being systematically and ruthlessly denied him. In return, they infuse into him, idea by idea, thought by thought, attitude by attitude, the dialectics of materialism and secularism. The end result of this alien reblooding of thousands of men and women, is to create a Communist man - a creature intellectually sterile, spiritually void and oblivious to the realities of life. This creation, Communist man, on whom the Communists depend to conquer their future new world, is democracy's chief hope. This robot - thoughtless, lifeless and senseless, eventually will be the shoal on which Communism will flounder and die.	J. Edgar Hoover, speech, 5/2/1950, paragraph 12

References

Azunre, Paul. 2021. *Transfer Learning for Natural Language Processing.* Simon and Schuster. https://books.google.com?id=bGI7EAAAQBAJ.

Bowman, Samuel R., 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 (EMNLP).* Association for Computational Linguistics.

Cer, Daniel, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. "SemEval-2017 Task 1: Semantic Textual Similarity - Multilingual and Cross-lingual Focused Evaluation." *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 1–14. http://arxiv.org/abs/1708.00055.

Conneau, Alexis, and Douwe Kiela. 2018. "SentEval: An Evaluation Toolkit for Universal Sentence Representations." http://arxiv.org/abs/1803.05449.

Hill, Felix, Kyunghyun Cho, and Anna Korhonen. 2016. "Learning Distributed Representations of Sentences from Unlabelled Data." In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1367–77. San Diego, California: Association for Computational Linguistics. https://aclanthology.org/N16-1162.

Luitse, Dieuwertje, and Wiebke Denkena. 2021. "The Great Transformer: Examining the Role of Large Language Models in the Political Economy of AI." *Big Data & Society* 8 (2): 20539517211047734. https://doi.org/10.1177/20539517211047734.

Mebane Jr, Walter R, Patrick Wu, Logan Woods, Joseph Klaver, Alejandro Pineda, and Blake Miller. 2018. "Observing Election Incidents in the United States via Twitter: Does Who Observes Matter?" In *Annual Meeting of the Midwest Political Science Association, Chicago.*

Mikhaylov, Slava, Michael Laver, and Kenneth Benoit. 2012. "Coder Reliability and Misclassification in the Human Coding of Party Manifestos." *Political Analysis* 20 (1): 78–91. https://www.cambridge.org/core/journals/political-analysis/article/coder-reliability-and-misclassification-in-the-human-coding-of-party-manifestos/145AC6C39 0225AB29DA0BBA99038E796.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. "Efficient Estimation of Word Representations in Vector Space." http://arxiv.org/abs/1301.3781.

O'Shea, James, Zuhair Bandar, and Keeley Crockett. 2014. "A New Benchmark Dataset with Production Methodology for Short Text Semantic Similarity Algorithms." *ACM Transactions on Speech and Language Processing* 10 (4): 19:1–63. https://doi.org/10.114 5/2537046.

R Core Team. 2016. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Reimers, Nils, Philip Beyer, and Iryna Gurevych. 2016. "Task-Oriented Intrinsic Evaluation of Semantic Textual Similarity." In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 87–96. Osaka, Japan: The COLING 2016 Organizing Committee. https://aclanthology.org/C16-1009.

Reimers, Nils, and Iryna Gurevych. 2019. "Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks." http://arxiv.org/abs/1908.10084.

Rheault, Ludovic, and Christopher Cochrane. 2020. "Word Embeddings for the Analysis of Ideological Placement in Parliamentary Corpora." *Political Analysis* 28 (1): 112–33. https://www.cambridge.org/core/journals/political-analysis/article/word-embeddings-for-the-analysis-of-ideological-placement-in-parliamentary-corpora/017F0CEA9B3DB6E1B94AC36A509A8A7B.

Rodriguez, Pedro L., Arthur Spirling, and Brandon M. Stewart. 2021. "Embedding Regression: Models for Context-Specific Description and Inference." *Working Paper*, June.

Rogers, Anna, Olga Kovaleva, and Anna Rumshisky. 2021. "A Primer in BERTology: What We Know About How BERT Works." *Transactions of the Association for Computational Linguistics* 8 (January): 842–66. https://doi.org/10.1162/tacl_a_00349.

Rudkowsky, Elena, Martin Haselmayer, Matthias Wastian, Marcelo Jenny, Štefan Emrich, and Michael Sedlmair. 2018. "More Than Bags of Words: Sentiment Analysis with Word Embeddings." *Communication Methods and Measures* 12 (2-3): 140–57. https://doi.org/10.1080/19312458.2018.1455817.

Song, Kaitao, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. "MPNet: Masked and Permuted Pre-training for Language Understanding." In *Advances in Neural Information Processing Systems*, 33:16857–67. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2020/file/c3a690be93aa602ee2dc0ccab5b7b67e-Paper.pdf.

Tan, Xu. 2020. "MPNet Combines Strengths of Masked and Permuted Language Modeling for Language Understanding." Microsoft Research. December 9, 2020. https://www.microsoft.com/en-us/research/blog/mpnet-combines-strengths-of-masked-and-permuted-language-modeling-for-language-understanding/.

Ushey, Kevin, JJ Allaire, and Yuan Tang. 2022. *Reticulate: Interface to 'Python'*. Manual.

Van Rossum, Guido, and Fred L. Drake. 2009. *Python 3 Reference Manual* (version 3.0).

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. "Attention Is All You Need." In *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc. https://papers.nips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Zhu, Xunjie, Tingfeng Li, and Gerard de Melo. 2018. "Exploring Semantic Properties of Sentence Embeddings." In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 632–37. Melbourne, Australia: Association for Computational Linguistics. https://aclanthology.org/P18-2100.

Zhu, Yukun, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. "Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books." In, 19–27. https://www.cv-foundation.org/openaccess/content_iccv_2015/html/Zhu_Aligning_Books_and_ICCV_2015_paper.html.