Word Embeddings: What Works, What Doesn't, and How to Tell the Difference for Applied Research

Pedro L. Rodriguez, Vanderbilt University
Arthur Spirling, New York University

Word embeddings are becoming popular for political science research, yet we know little about their properties and performance. To help scholars seeking to use these techniques, we explore the effects of key parameter choices—including context window length, embedding vector dimensions, and pretrained versus locally fit variants—on the efficiency and quality of inferences possible with these models. Reassuringly we show that results are generally robust to such choices for political corpora of various sizes and in various languages. Beyond reporting extensive technical findings, we provide a novel crowdsourced "Turing test"–style method for examining the relative performance of any two models that produce substantive, text-based outputs. Our results are encouraging: popular, easily available pretrained embeddings perform at a level close to—or surpassing—both human coders and more complicated locally fit models. For completeness, we provide best practice advice for cases where local fitting is required.

The idea that words and documents can be usefully expressed as numerical objects is at the core of modern political methodology. The exact method one uses to model "text as data" has been debated. But in recent times, "word embeddings" have exploded in popularity. The premise of these techniques is beguilingly simple: a token of interest ("welfare" or "washington" or "fear") is represented as a dense, real-valued vector of numbers. The length of this vector corresponds to the nature and complexity of the multidimensional space in which we are seeking to "embed" the word. And the promise of these techniques is also simple: distances between such vectors are informative about the semantic similarity of the underlying concepts they connote for the corpus on which they were built. Applications abound. Prosaically, they may be helpful for a "downstream" modeling task where we need better inputs to some supervised problem. More excitingly, the similarities may be substantively informative per se: if the distance between "immigrants" and "hardworking" is smaller for liberals than for conservatives, we learn something about their relative worldviews.

Exploiting the basic principles behind these examples, word embeddings have seen tremendous success as feature representations in well-known natural language processing problems. These include parts-of-speech tagging, named-entity recognition, sentiment analysis, and document retrieval. Given the generality of those tasks, it is unsurprising that word embeddings are rapidly making their way into the social sciences (e.g., Kozlowski, Taddy, and Evans 2018), political science being no exception (e.g., Rheault and Cochrane 2019; Rodman 2019). But as is often the case with the transfer of technology, there is a danger that adoption will outpace understanding. Specifically, we mean comprehension of how well the technology performs—technically and substantively—on specific problems of interest in the domain area of concern. The goal of this paper is to provide that understanding for political science, enabling practitioners to make informed choices when using these approaches.

As conveyed in our examples above, word embeddings serve two purposes. First they have an instrumental function, as feature representations for some other learning task. Second,

Pedro L. Rodriguez (pedro.rodriguez@Vanderbilt.Edu) is a postdoctoral fellow at Vanderbilt University's Data Science Institute, Nashville, TN 37212. Arthur Spirling (arthur.spirling@nyu.edu) is professor of politics and data science at New York University, New York, NY 10012.

Spirling was supported in part by National Science Foundation grant no. 1922658. Data and supporting materials necessary to reproduce the numerical results in the article are available in the *JOP* Dataverse (https://dataverse.harvard.edu/dataverse/jop). An online appendix with supplementary material is available at https://doi.org/10.1086/715162.

Published online November 24, 2021.

  Published by The University of Chicago Press for the Southern Political Science Association. https://doi.org/10.1086/715162

embeddings are a direct object of interest for studying word usage and meaning—that is, human semantics. Good performance in the former need not correlate with good performance in the latter (Chiu, Korhonen, and Pyysalo 2016). In this paper we focus on this second purpose: embeddings as measures of meaning. The reasoning is simple. First, we cannot pretend to foresee all the downstream use cases to which political scientists will apply embeddings. Moreover, given a well-defined downstream task, how to think about performance is trivial—these are usually supervised tasks with attendance metrics measuring accuracy, precision, and recall. Second, word usage, including differences between groups and changes over time, is of direct and profound interest to political scientists, but validating such quantities automatically is challenging (Faruqui et al. 2016).

With this in mind, our specific contribution goes beyond a useful series of results. We propose the framework used to generate them, which will guide researchers through the maze of choices that accompany word embeddings. These include whether to use cheap pretrained or (more) expensive "local" corpus trained embeddings. And, within models, we demonstrate the effects of altering core hyperparameters such as context window size and embedding dimensions. In addition to standard predictive performance and computational cost metrics though, we present two novel approaches to model comparison and validation. First, framing the task as an information retrieval one, we show how models may be mechanically compared in terms of the words they place close to others—including politics-specific tokens. As a second "gold-standard" approach, we propose a new take on the classical Turing test wherein human judges must choose between computer-generated nearest neighbors and human-generated nearest neighbors. While we necessarily make certain choices in terms of embedding architecture and which hyperparameters to focus on, we stress that the framework developed is completely general and not beholden to these choices. It is easily adaptable to evaluate new models—including nonembedding models of human semantics—and other parameter variations.

Our findings are reassuring. In particular, (cheap, readily available) pretrained embeddings perform extremely well on a multitude of metrics relative to human coding and (more expensive) locally trained models for political science problems. This is true beyond our focus *Congressional Record* corpus and extends even to non-English collections. Separate to our intellectual contribution, we also provide the full set of all local models we fit—250 in all—so practitioners can use them "off the shelf" in their own work along with two novel corpora of parliamentary speeches in Spanish and German.

In the next section we clarify terms and provide some background on origins of word embeddings. We then lay out the choices practitioners face, before discussing evaluation methods and how we implement them. Subsequently, we extend our work to a variety of corpora, and we then summarize the main takeaways for researchers.

WORD EMBEDDINGS IN CONTEXT

To fix ideas, consider two popular problems—and solutions—in political science:

1. Estimating political attention; specifically, the focus that legislators place on different aspects of policy at different times.
2. Characterizing the nature of political conflict; specifically, the way in which politicians of different parties use tellingly different words to discuss similar issues

In the case of problem 1, topic models have proved extremely popular (e.g., Quinn et al. 2010). In the case of problem 2, various supervised machine-learning efforts have emerged (e.g., Diermeier et al. 2012; Monroe, Colaresi, and Quinn 2008). With some exceptions, analysts have not attacked these problems with embeddings approaches, which differ from current practice both conceptually and practically.

At a high level, these differences boil down to a particular implementation of the distributional hypothesis—the belief that we may "know a word by the company it keeps" (Firth 1957, 11), that is, that a word's meaning may be learned from its context. In the case of topic models fit via latent Dirichlet allocation (LDA; Blei, Ng, and Jordan 2003), this insight is obtained via the co-occurrence of terms with the word of interest in a document. In the case of embedding models, it is operationalized as the other words that appear near it in text—literally in a narrow window around the word of interest, looking over the corpus as a whole.

An immediate practical consequence of this difference is that the way text data are represented for the models differs fundamentally for embeddings approaches. In the case of both problems 1 and 2 above, the standard arrangement would be to represent each document as a vector, with each element of that vector a (possibly reweighted) set of frequencies of the terms it contains (see, e.g., Denny and Spirling [2018] on "preprocessing"). But this "bag of words" approach is not how we proceed in the case of embeddings. Specifically, our texts are now represented as ordered sequences: the occurrence of a given word(s) is used to predict the occurrence of the next one(s). There are many approaches and "architectures," and they define the windows (the "context") around the word in different ways. But they generally all proceed by using an (artificial) neural network that maps words to real-valued vectors:

indeed, the embeddings are the coefficients of that neural network model. Intuitively, each element of those vectors represents some hypothetical characteristic of the word(s). The vector for a given word can be called a "word embedding." Scholars (e.g., Collobert and Weston 2008) have shown that embeddings capture substantive syntactic and semantic information about language per se that can be explored using basic algebraic operations. A textbook example is using embeddings for analogical reasoning like king − man + woman ≈ queen, where each word in the equation is the vector that represents it. Clearly, what emerges from an embedding model—a real valued vector for every word in the corpus—is very different from the outputs of the traditional approaches above. There, respectively, we obtain a distribution over terms in the vocabulary of the corpus (a "topic"), or a set of words that maximally discriminate between the corpora produced by one party versus another. In neither case do we obtain estimates of semantic relationships between words at the word level.[1]

Given that the inputs, the model, and the outputs are so very different in the case of embeddings, it is unsurprising that the decisions one needs to make when using an embeddings approach are different. Everything that follows is about those decisions: both how they should be made generally and what lessons may be learned when they are applied to representative political science corpora specifically. Broadly these decisions concern:

1. How large a window size we want the model to use. This is a decision about what we mean by "context": one word either side of our target? six words? 12 words? and so forth.
2. How large an embedding to use to represent the words. This is a decision about how complex a model we wish to fit—literally, the number of dimensions (from say, 50 to 450) our word vectors should ultimately have. Too many dimensions, and we may be modeling noise. Too few dimensions, and we risk not capturing important subtleties in meaning.
3. Whether to fit the embedding models locally or to use pretrained embeddings fit to some other (ideally related) corpus. This is a decision about whether our texts are sufficiently similar to other corpora for which embeddings have already been learned. In particular, pretrained embeddings are typically based on the relationships between words in Wikipedia and other large-scale collections. They are computationally cheap (and can be downloaded in minutes) but may be inaccurate for our specific problem, in which case, we may wish to locally train: that is, fit an embeddings model from scratch to our corpus.
4. The specific modeling approach to use—where the choice for most practitioners is GloVe (Pennington, Socher, and Manning 2014) or Word2Vec (Mikolov et al. 2013). We will give more details below, but these are similar mathematically, differing primarily in the way they use co-occurrences of terms in the documents as a whole.

To guide readers more familiar with traditional approaches in political science, table 1 provides an overview of how embeddings differ from those methods—in terms of structure, models, inputs, and so on. In the final row of the table, we note that the various approaches have issues with "stability." In the case of embeddings, the issue is that vectors learned on the same corpus, even with the same technique for the same word, need not be identical for every run of the model. This is partly about inevitable stochastic behavior of algorithms in this field, but it is also connected with fundamental sampling uncertainty in (small) finite corpora (Antoniak and Mimno 2018). Our maintained assumption in what follows is that researchers would prefer more stable to less stable outputs but would always like to know how much stability they could expect in practice.

What substantive light would embeddings, perhaps fit to the *Congressional Record*, shed on the problems we alluded to above? In either case, embeddings tell us how a word is associated with others. For the first problem, of "attention" estimation, embeddings would help us locate sets of documents/speeches that are about similar themes (e.g., Hamilton, Leskovec, and Jurafsky 2016) but also to understand how those themes and the words they contain evolve over time (e.g., Rodman 2019). In terms of the second "conflict" problem, we could imagine learning embeddings for both Republican and Democrat Senators for particular words—say, abortion or welfare. The way that those embeddings differ—literally, the words those words are closest to for the partisans in each group—would presumably be informative about the nature of debate in the contemporary US Congress (see Mebane et al. [2018], Rheault and Cochrane [2019], and Rudolph et al. [2017] for related applications on partisanship). Beyond being of interest per se, the embeddings could obviously be used as inputs to other downstream tasks (see, e.g., Zhang and Pan [2019] for an example in sociology): for example, we might find that knowing the embedding representation of a bill is helpful for predicting whether parliamentarians are likely to vote for it or not.

1. While these may be retrieved from the output of topic models, this is not their intended use case.

Table 1. Comparing Traditional Approaches with Embeddings

	Traditional Unsupervised	Traditional Supervised	Embeddings
Bag of words	Yes	Yes	No
Example models	Latent Dirichlet allocation; structural topic model	Support vector machines; random forest (RF)	GloVe, Word2Vec
Citations/applications	Quinn et al. (2010), Roberts et al. (2014)	Diermeier et al. (2012), Montgomery and Olivella (2018)	Rheault and Cochrane (2019), Rodman (2019)
Inputs	Document-term matrix	Document-term matrix; labeled y	Term-co-occurence matrix
Outputs	Document distribution over topics; topic distribution over words	Term importance matrix (for class prediction)	Word vectors
Example user decisions	Weighting of tokens; number of topics	Training/test split; weighting of tokens; number of trees (RF); number of variables at each split (RF); prior class probabilities	Pretrained or local fit; window size; embedding dimensions; weighting of tokens
Stability concerns	Multiple modes	Sensitivity to training/test set; labeling errors	Algorithmic; corpus characteristics

To make matters concrete below, we will use the "partisan conflict" case as a running example. We will assume that the analyst wants to learn embeddings (vectors) for a set of political terms but does not know a priori which modeling approach is optimal—in terms of run time, quality of fit to their data, and so on. Importantly, we also assume that the researcher does not know the "accuracy" (in an intrinsic, substantive sense) of the representations they might learn. To clarify, suppose the model informs the researcher that, for Republicans, the terms closest to welfare are dependency and reform. Crudely, is this a "good" embedding? Does it reflect judgements that human researchers (an assumed gold standard) would make about the context for this word? Assessing this in a general way will be one of our primary contributions. Before we get there, we must deal with the series of choices we outlined above. As a road map, in figure 1, we give a flowchart of decisions that a typical user (including our "partisan conflict" researcher) would need to make in learning embeddings for their corpus. The terms in bold will be our focus.

Keeping with the spirit of our running example, we will be interested in modeling the *Congressional Record*, specifically, the set of transcripts for the 102nd–111th Congresses (Gentzkow, Shapiro, and Taddy 2018)—a medium sized corpus of around 1.4 million documents. These contain all text spoken on the floor of both chambers of Congress. We restrict our corpus to the set of speeches for which party information is available. We do minimal preprocessing: remove all nontext characters and lower case. Next we subset the vocabulary. We follow standard practice, which is to include all words with a minimum count above a given threshold—we choose 10. This yields a vocabulary of 91,856 words.[2] And we focus on GloVe embeddings simply because it seems more popular with social scientists.[3] We will nonetheless also provide commentary on taking embeddings to other languages and using Word2Vec—the other popular model in this literature.

EVALUATING EMBEDDING MODELS FOR SOCIAL SCIENCE

To evaluate the effects of the choices, we need tasks. In the word embeddings literature (Schnabel et al. 2015), they fall into two categories: extrinsic and intrinsic. Extrinsic tasks are those in which researchers use embeddings as input variables to perform some other (typically, supervised learning) task that has prespecified "correct" answers. One chooses the optimal embedding specification by varying the hyperparameters of the model and then examining the changes to accuracy and other performance statistics that occur when doing this task. Our position is that these tasks are rare and highly specific in political science (Denny and Spirling 2018): certainly we could not think of an obvious one for the *Congressional Record*. An intrinsic task is one in which we assess the worth of particular embeddings by how well they represent "true" relationships between words. In computer science, scholars proceed by, for example, comparing the analogies implied by embeddings (using the rough calculus we mentioned

2. The pretrained GloVe vocabulary consists of 400,000 tokens.

3. In particular, the GloVe pretrained with window size 6 and embedding dimensions 300, available on February 2, 2019, from https://nlp.stanford.edu/projects/glove/, for which the training corpus is Wikipedia 2014 and Gigaword 5.

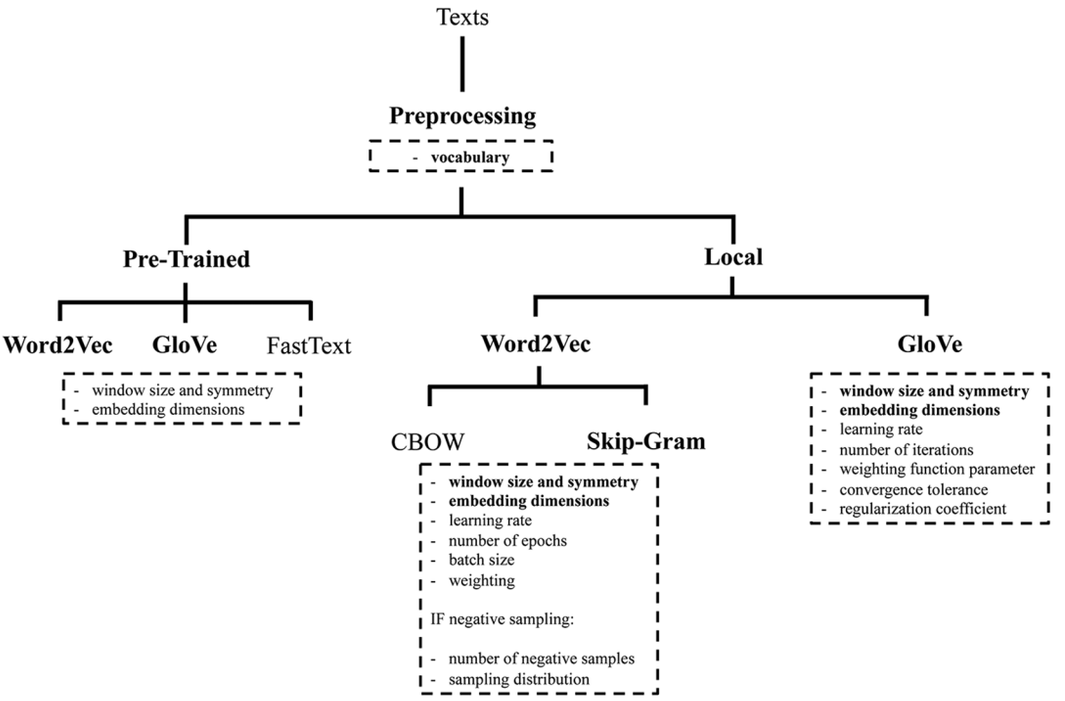

Figure 1. Decision flowchart for embeddings. Branches are either/or approaches. Boxes list decisions to be made under those approaches. Decisions in bold are ones we evaluate below.

above) to a preexisting (and commonly known) bank of such relationships between words (e.g., Mikolov et al. 2013). In principle, a politics researcher could create such a database for congressional speeches. In line with the vector calculus mentioned above for contemporary times, we would want the following embedding equation for leadership to "work" (with closer approximations being better): Pelosi − House + Senate ≈ McConnell. We cannot think of many such examples, however. Moreover, these obviously do not travel: these relationships do not even make sense for the House of Commons/Lords in the United Kingdom (or equivalent in Canada), let alone for other areas of the discipline. While we will still focus on intrinsic tasks—that is, "real life" uses of words—we will take a more "human" approach to evaluate embedding quality.

Turing assessment

To evaluate semantic coherence we draw inspiration from the principles laid out by Turing (1950) in his article on computer intelligence. There, a machine showed human-like abilities if a person engaging in conversation with both a computer and a human could not tell which was which. We use that basic intuition. In particular, an embedding model achieves "human" performance if human judges—crowd workers—cannot distinguish between the output produced by such a model from that produced by independent human coders. In our case, the idea is not to "fool" the humans but, rather, to have them assert a preference for one set of outputs over another. If a set of human judges are on average indifferent between the human responses to a prompt and the model's responses, we say we have achieved human performance with the model. By extension, a model can achieve better than human performance by being on average preferred by coders. Naturally, models may be worse than human if the judges like the human output better.

Notice that while the traditional Turing test connotes a human versus machine contest, the approach here is more general. Indeed, any output can be compared to any other—including where both sets are produced by a model or both by humans—and conclusions drawn about their relative performance as judged by humans. In contrast with other intrinsic evaluation metrics found in the literature, our proposed assessment can incorporate both absolute and comparative evaluations.

The steps we take to assess the relative Turing performance of the models are:

1. *Human-generated nearest neighbors*: For each of 10 political prompt words (described below), have humans—crowd workers on Amazon MTurk—produce a set of nearest 10 neighbors—we have 100 humans perform this task. Subsequently rank "human" nearest neighbors for each prompt in terms of the number of mentions and choose the top 10 for each prompt.

2. *Machine-generated nearest neighbors*: For the embedding model under consideration—pretrained or some variant of the locally fit set up—produce a list of 10 nearest neighbors for each of the 10 given prompt words above.[4]
3. *Human rating*: Have a separate group of humans perform a triad task—135 subjects on average for each model comparison—wherein they are given a prompt word along with two nearest neighbors—a computer- and a human-generated nearest neighbor—and are asked to choose which nearest neighbor they consider better fits the definition of a context word. See appendix A (apps. A–G are available online) for task appearance/wording.
4. *Compute metric*: For each prompt compute the expected probability of the machine-generated nearest neighbor—our candidate model—being selected vis-à-vis a baseline model—humans in our gold standard. We divide this number by 0.5, such that the index will range between 0 and 2. A value of 1 implies that the machine is on par with human performance (i.e., a human rater is equally likely to choose a nearest neighbor generated by the embedding model as one generated by another human) while a value larger (smaller) than 1 implies that the machine performs better (worse) than humans.

We give specific details on how we handle the crowd workers themselves, and the comparisons they produce, in appendix B.

Our approach is predicated on our human coders being able to make reasonable judgments about contexts in the way we described, which seems plausible based on other work (Benoit et al. 2016). Notice that for our comparisons to be useful, we do not require that humans are error free in their judgments: any applied coding problem will involve some mistakes. Rather, we require that, absent time or budget constraints, human coding would be the best possible approach. And thus, to the extent an embedding model can replicate it, that model is of value.

If one believes that the crowd workers differ systematically in their understanding of terms from the "population" underlying the corpus (however defined), one might use covariate information to stratify. That is, one might reweight responses in a way that better reflects the population of interest. Related is the idea that the crowd workers might differ in their responses to embeddings as a function of their own covariate profiles (e.g., whether they are male or female, Democrat or Republican, etc.). However, in what follows, we will not use covariate information on our crowd workers. The reason for this is connected to both research design and ethics. On research design, note that our crowd workers are being treated as de facto research assistants: they are helping us study embeddings; embeddings are not helping us study our research assistants. Thus, regarding ethics, this is not human subjects research, and we have not obtained permissions for the same.

In reference to our running example, we could imagine a traditional approach using research assistants to read Democrat and Republican speeches to identify the different ways they talk of the same issue. These assistants would be required to find the context for terms—literally, what they connote. Here, at a high level, our embedding model is suggesting possible contexts. If humans "like" those suggestions (relative to those suggested by crowd equivalents of research assistants), we have evidence that the models work for this purpose. But to reiterate, we as researchers do not believe model performance—on human or other metrics (below)—will vary by party, so we will focus on aggregate performance in what follows.

In practice, we push our comparison beyond mere binary preferences. Specifically, we also investigate how similar (via log rank deviation) the human output versus embedding output is in terms of the set of nearest neighbors they produce. This gives us a finer-grained sense of how inferences from humans or models would differ in practice.

Other metrics

The foregoing section set out how to assess whether embeddings "make sense." But we also consider other practical issues that applied researchers might have. In particular, we will report results related to:

1. technical criteria—model loss and computation time;
2. query search ranking correlation—across-model (varying hyperparameters) Pearson and rank correlations of nearest neighbor rankings; and
3. model variance (stability)—within-model (holding hyperparameters constant) Pearson correlation of nearest neighbor rankings across multiple initializations.

For 1, we mean how well different models fit our congressional corpus according to their internal optimization procedures (Pennington et al. 2014) and then literally how long it takes each model to fit (in minutes). We assume that researchers would like to know how time consuming such procedures are for an example problem. For 2, we mean the correlation between the outputs of one embedding model specification and another. If these are high across many different choice combinations, there is a sense in which it does

4. It is common in the literature to focus on the top 10 nearest neighbors.

not much matter what the applied researcher chooses. Finally, for 3, we measure how different the results are for the same model run multiple times. Intuitively, if this is high for a given specification (i.e., the correlation is low) there is a danger that researchers will update from an "unusual" run, the general output of which would not be replicated by others using the same code and data. We privilege this concern over what we believe to be a conceivable but limited benefit of different runs revealing different relationships in the data—some of which may be particularly useful. In any case, we suspect users will want to know how much variance they can expect in practice.

ESTIMATION SETUP

We focus our analysis on two hyperparameter choices—five values of each for 25 combinations in all—though of course the framework we lay out is not specific to these parameter pairs:

1. window size—1, 6, 12, 24, and 48 and
2. embedding dimension—50, 100, 200, 300, 450.

To account for estimation-related instability we estimate 10 sets of embeddings for each hyperparameter pair, each with a different randomly drawn set of initial word vectors. In total we estimate 250 different sets of embeddings. The only other hyperparameter choices we make and leave fixed are the number of iterations and convergence threshold. For each model we set the maximum number of iterations to 100 and a use a convergence threshold of 0.001 such that training stops if either the maximum number of iterations is reached or the change in model loss between the current and preceding iterations is below the convergence threshold. None of our models reached the maximum number of iterations before meeting the convergence threshold. We set all remaining hyperparameter values at their default or suggested values in the GloVe software.[5] The set of locally trained models—available on the project's GitHub—represents a contribution in and of itself that will hopefully save researchers time and computational resources.

QUERY SELECTION

We explained that a natural quantity of interest is the set of nearest neighbors of a given word—a query—in the embeddings space. These form the core of our comparison metric in the sense that we will want to know how similar one set of nearest neighbors from one model specification is to another. And, by extension, how "good" one set of nearest neighbors is relative to another in terms of a quality evaluation by human judges. We use two sets of queries: a random sample of 100 words from the common vocabulary and a set of 10 curated political terms.

First among our curated political terms, there are a series of concept words that we suspected would be both easily understood but also exhibit different meanings depending on who is asked: democracy, freedom, equality, justice. Second, there are words pertaining to policy issues that are debated by political parties and motivate voting: immigration, abortion, welfare, taxes. Finally, we used the names of the major parties, which we anticipated would produce very different responses depending on partisan identification: republican, democrat (see Halpern and Rodriguez 2018). Obviously, we could have made other choices. And indeed, we would encourage other researchers to do exactly that. Our queries are intended to be indicative of what we expect broader findings to look like and to demonstrate the utility of our generic approach.

RESULTS: PERFORMANCE COMPARED

Now we report the results for the evaluation metrics outlined in "Evaluating Embedding Models for Social Science." We begin with the "human" performance of embeddings. To reiterate, our presumption is that we would like embedding models to produce nearest neighbors that accord with human priors—and that thus might otherwise have been suggested by research assistants. As we will see, the good news is that they come close, and sometimes do better.

Turing assessment

Figure 2 measures performance of a "candidate" model relative to a "baseline" model. Here, values above (below) 1 mean nearest neighbors from the "candidate" model were more (less) likely to be chosen by human raters. A value of 1 means human raters were on average indifferent between the two models. Figure 2*A* compares two local models: 48–300 (candidate) and 6–300 (baseline). There is no unqualified winner and, in fact, these models have a 0.92 correlation (see fig. 5*B*). If nothing else, this suggests that—from a human perspective—different hyperparameter choices, at least around common practice (window of 6, vector length 300) do not matter much.

How do local models fare directly against human-generated nearest neighbors? Except for one query (immigration), the local model of choice—6–300—shows below-human performance for all but two of the queries. On average, for the set of 10 political queries, the local model achieves 69% (SD = 0.20) of human performance. Turning to pretrained GloVe embeddings, we observe that they are generally preferred to locally trained embeddings (see fig. 2*C*). Moreover,

5. We use the text2vec R package to run all our models.

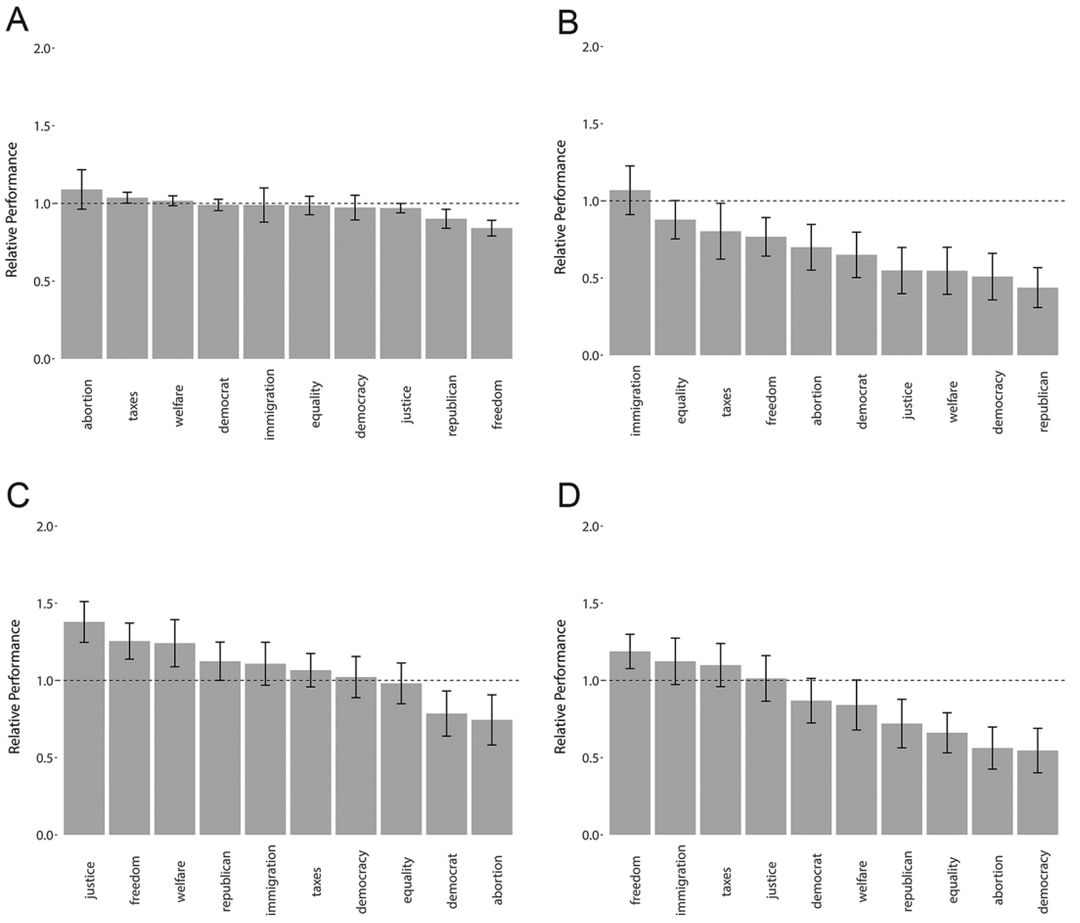

Figure 2. Human preferences: Turing assessment. *A*, Candidate: local 48–300; baseline: local 6–300. *B*, Candidate: local 6–300; baseline: human. *C*, Candidate: GloVe; baseline: local 6–300. *D*, Candidate: GloVe; baseline: human.

pretrained embeddings are more competitive against humans—albeit with greater variance—achieving an average of 86% (SD = 0.23) of human performance (see fig. 2*D*). That is, a researcher using pretrained embeddings for this corpus (available in minutes, at no cost) is able to produce results very similar to those that would have been produced by (relatively slow, costly) research assistants. This is comforting news.

Using the log rank deviation measure, we can compare all models given our set of human-generated lists (see fig. 3). We find that models with larger windows and more dimensions show lower log rank deviations, indicating better performance but with decreasing returns. Put simply, this means that while "bigger" models will get the researcher results closer to those that human coders would produce, so long as users do not opt for the very simplest models (very small windows, short vectors), they can obtain relatively excellent performance with moderately complex ones.

These aggregate results are encouraging, but our experiments can also shed light on exactly where—that is, on what types of queries—researchers might expect to do better or worse in terms of their model selections. First, we note that human coders respond (relatively) poorly to the pretrained outputs for abortion and Democrat as compared to those from the locally trained model (6–300). We give those nearest neighbors in table 2, in the first two groups (first four columns). Digging deeper, we found that human coders particularly liked "lateterm" and "partial" when they were offered them—context words that were not even selected by the pretrained model. In the case, of "Democrat", the human coders liked "liberal" when it was offered. Exactly why human coders preferred these is debatable, but our observation would be that these accord with more "everyday" uses of these terms.

Our investigation on where the local model does worse bears out this hunch. The final column of table 2 reports the nearest neighbors for "justice." As can be readily seen, the local model mostly returns a list of Supreme Court Justices (unsurprisingly, given that this is based on discussions in the *Congressional Record*). By contrast, crowd workers liked "law" and "court" when it was offered. One lesson here, and one we will comment on below, is that crowd workers prefer terms in general use over and above those that require very specific domain knowledge. We suspect this is true of research assistants too.

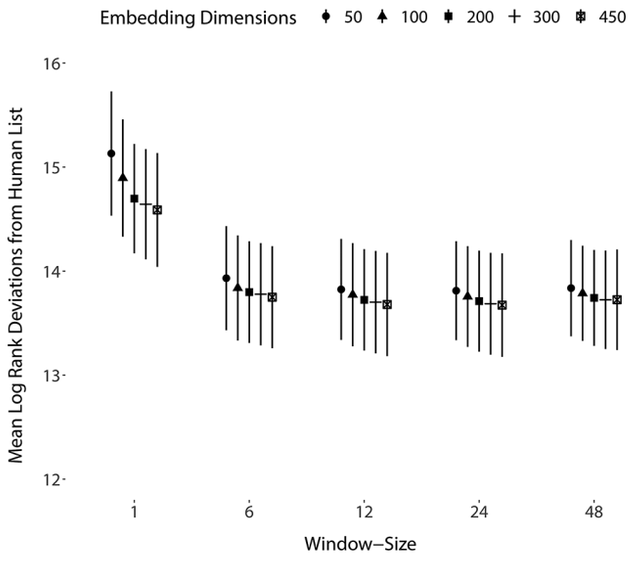

Figure 3. Human preferences: log rank deviations: complex models come closer to "human" assessments, but medium-size models are almost as good as very large ones.

Other metrics

Figure 4*A* displays the mean—over all 10 initializations—minimum loss achieved for 16 (of the 25) parameter pairs we considered.[6] Unsurprisingly, more dimensions and larger window sizes both unconditionally improve model fit albeit with decreasing returns in both parameter choices. Except for very small window sizes (<6), improvements become marginal after around 300 dimensions. If we take loss seriously, then researchers ought to avoid combining few dimensions (<100) with small window sizes (<6). There are two caveats, however. First, models with different window sizes represent qualitatively different notions of context, and presumably the match between that and the substantive problem at hand is more important than comparing relative fit. With reference to our running example, what we mean by "context" for partisan difference should presumably be assessed by the qualitative meaningfulness of the comparisons, not simply an agnostic model loss.

Second, unsurprisingly, bigger models take longer to fit (see fig. 4*B*). The largest here (48–450) took over three hours to compute parallelizing over eight cores. This seems reasonable if only computing once but can become prohibitive when computing over several initializations as we suggest. In this light, the popular parameter setting 6–300 (window size 6, embedding dimensions 300) provides a reasonable balance between performance and computation time.

6. We plotted 16 of the 25 parameter pairs to avoid clutter. The left-out parameter pairs follow the same trend.

Different parameter choices produce different results in terms of performance, but what do these differences mean substantively? To answer this, we compare models with respect to how they rank query searches. Figure 5*A* displays a heat map of pairwise correlations for all models, including GloVe pretrained embeddings, for the set of random queries. We observe high positive correlations (>0.5) between all local models. Correlations are generally higher between models of the same window size, an intuitive result, as they share the underlying co-occurrence statistics. Somewhat less intuitive, comparing models with different window sizes, correlations are higher the larger the window size of the models being compared (e.g., 6 and 48 vis-à-vis 1 and 6). Correlations are larger across the board for the set of political queries (see fig. 5*B*). These results suggest that the embeddings for the *Congressional Record* are sensitive to window size, but this decreases quickly as we go beyond very small window sizes (i.e., models with window sizes of 6 and 48 show much higher correlation than models with window sizes of 1 and 6).

The last column of figures 5*A* and 5*B* compares GloVe pretrained embeddings with the set of local models. For this comparison we subsetted the respective vocabularies to include only terms common to both the local models and the pretrained embeddings. As would be expected, correlations are lower than those between local models, yet they are still surprisingly large—especially for local models with larger window sizes and for the set of political queries (all above 0.5). Our reading is that GloVe pretrained embeddings, even without any modifications (Khodak et al. 2018), may be a suitable alternative to estimating locally trained embeddings on present-day political corpora. This is good news for political scientists who have already relied on pretrained embeddings in their work.

As a final check, and in keeping with the nature of our running example, we looked at whether pretrained embeddings might do a "worse" job of reflecting highly specific local embeddings for our focus corpus. In this case, we mean party: it could in principle be the case that while pretrained embeddings do well in aggregate for the *Congressional Record* they do poorly for Democrats or Republicans specifically. To evaluate this we estimate a set of additional local models (again, 10 for each group and using 6–300 as parameter settings) for subsets—by party—of the aggregate corpus. We find no statistically significant differences in correlations (see app. C).

Finally, we report our results on stability. Figure 6*A* plots the distribution of Pearson correlations for the 100 random queries. Correlations are high—above 0.85—across the board, suggesting that GloVe is overall rather stable—that is, the organization of the embeddings space does not vary dramatically over different initializations. Still, models with larger

Table 2. Where Pretrained Does Better or Worse Than Local

"abortion"		"democrat"		"justice"	
pretrained	local	pretrained	local	pretrained	local
abortions	abortions	senator	republican	supreme	rehnquist
contraception	partialbirth	democratic	democratic	justices	scalia
euthanasia	lateterm	republican	democrats	judicial	owens
antiabortion	procedure	democrats	republicans	court	ginsburg
legalized	ban	sen.	party	judge	court
homosexuality	partial	rep.	liberal	criminal	souter
opposes	clincs	congressman	majority	law	oconnor
prolife	sterilization	senate	conservative	attorney	brennan
pregnancy	clinic	incumbent	side	appeals	department
advocates	birth	gop	ranking	scalia	supreme

Note. First two groups: words ("abortion," "democrat") for which crowd workers preferred local model nearest neighbors (right columns) to pretrained model ones (left columns) for GloVe. Last group ("justice"): crowd workers prefer the pretrained model.

window sizes produce, on average, more stable estimates. As the number of dimensions increases, the difference in stability between different window sizes decreases and eventually flips—larger window sizes result in greater instability. This parabolic relationship between window size, number of dimensions, and stability is likely a function of corpus size—larger more generic corpora will require a greater number of dimensions to allow for multiple word senses—and token frequency—infrequent tokens are likely to be more unstable.[7] For the set of 10 politics queries we observe the same trends, although they do not reach the point at which the relationship reverses (see fig. 6*B*).

Our running example: Partisan conflict results

To complete our main body of results, we return to our "partisan conflict" example. Do we have evidence that Republicans and Democrats differ in their understanding of various political terms? The answer to this is a qualified yes. Given a query and any two models, we can identify the set of nearest neighbors among the top 10 that are unique to each model. Following our setup above we estimate 10 models for each party—10 different initializations—giving us 100 pairwise comparisons that we use to identify the nearest neighbors most common to each party. Figure 7 captures the aggregate of this comparison for two of our prompts, abortion—showing the least overlap—and taxes —showing the most overlap. A value of 100 indicates that nearest neighbor was unique to one party across all 100 pairwise comparisons. Moreover, the fewer the number of terms on the *x*-axis, the greater the overlap across parties—that is, few terms were unique in any pairwise comparison.

Our conclusion is that embeddings can tell us which words partisans "disagree" on. And thus embeddings would indeed be informative about conflict. But this is not our central focus here; instead we now move to understanding whether the general lessons we take from the *Congressional Record* apply to other corpora that are similarly based on (mostly preprepared) elite-level speech.

OTHER CORPORA, OTHER LANGUAGES

We extend our evaluation to four other corpora, varying in size and language. The corpora are:

1. the full set of speeches from the UK Parliament for the period 1935–2016 obtained from Rheault et al. (2016);
2. all State of the Union (SOTU) speeches between 1790 and 2018;
3. the full set of speeches from both chambers of the Spanish Legislature—*Cortes Generales*— for the V–XII legislatures.[8] As political queries we use democracia, libertad, igualdad, equidad, justicia, inmigracion, aborto, impuestos, monarquia, parlamento.
4. the full set of speeches from the German Legislature—*Deutscher Bundestag*—for the election periods 14–19.[9] The political queries in this case are demokratie, freiheit, gleichberechtigung, gerechtigkeit, einwanderung, abtreibung, steuern, cdu, and spd.

7. For the State of the Union corpus, a much smaller corpus, we find that the flip occurs after 100 dimensions (see apps. D and E).

8. As the XII was ongoing at the time of writing we used all speeches available up until October 18, 2018.

9. As the nineteenth *Wahlperiode* was ongoing at the time of writing we used all speeches available up until October 18, 2018.

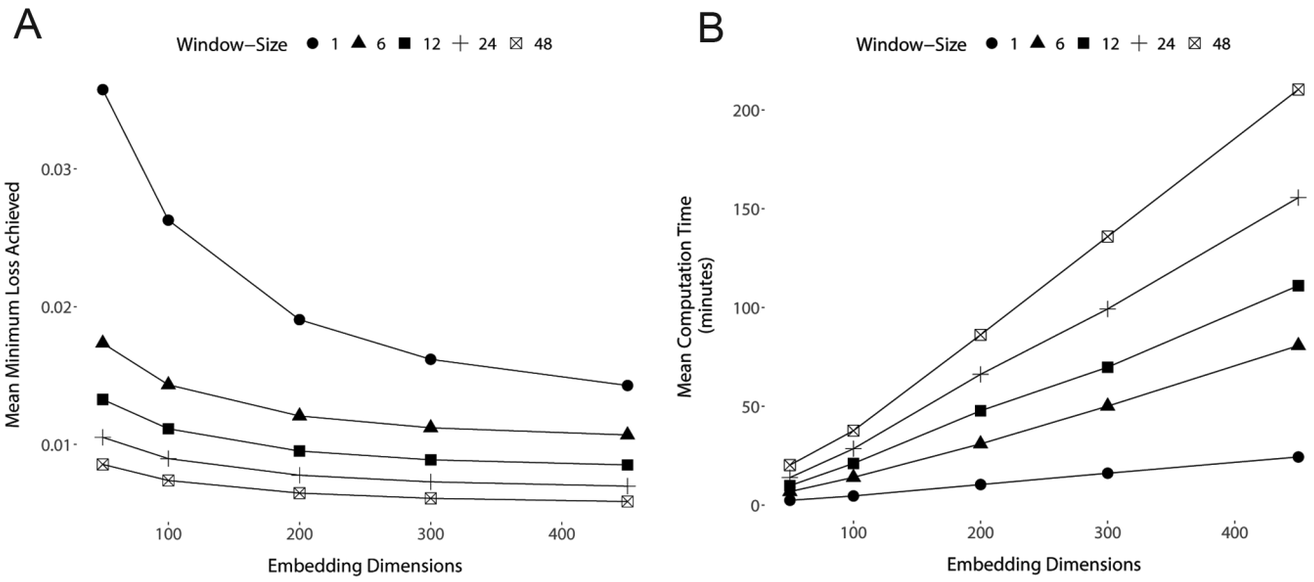

Figure 4. Technical criteria: larger models fit better but take longer to compute. *A*, Mean minimum loss achieved. *B*, Computation time (minutes)

We did not find readily available GloVe pretrained embeddings in German; as such all our comparisons in this case are between locally trained embeddings. Both the Spanish and German corpora are original data sets collected for the purposes of this paper.[10] Summary statistics for these corpora may be found in appendix D, but suffice it to say that the SOTU corpus is substantially smaller than all the other corpora and also encompasses a much longer time period.

In appendix E, we provide the same results plots for each of the above corpora as we gave for our *Congressional Record* corpus. Perhaps surprisingly, but no doubt reassuringly, these are almost identical to the ones above. That is, when we look at the embedding models we fit to these very different corpora, the lessons we learn in terms of hyperparameter choices, stability, and correlations across search queries (i.e., on the issue of whether to fit local embeddings, or to use pretrained ones) are the same as before. Of course, there are some exceptions: for example, we do find that models of window size equal to one perform well in the case of the SOTU corpus and for the German corpus—though to a lesser extent.

GLOVE VERSUS WORD2VEC: SOME DIFFERENCES

In contrast to GloVe, which approximates global co-occurrence counts, Word2Vec follows an online learning approach—the model is progressively trained as we move the context window along the corpus. Word2Vec at no point sees the global co-occurrence counts. Despite this difference, Pennington et al. (2014), the authors of GloVe, show that GloVe and Word2Vec's skip-gram architectures are mathematically similar. We might then conclude that they produce similar embeddings when trained on the same corpus. We find that this is not the case, though human raters do not seem to judge one or other as being better.

In appendix F we explain how we compared the two algorithms and give extensive results discussion. The main points are these: first, for Word2Vec (unlike with GloVe) pretrained embeddings exhibit much lower correlations with the set of local models.

Second, the correlation between both algorithms is never particularly high (for our set of parameter values). Our explanation for this difference is rooted in the way the algorithms are implemented. Whereas GloVe explicitly underweights relatively rare terms, Word2Vec explicitly underweights high frequency terms. Consequently, Word2Vec often picks out relatively rare terms (including misspellings) as nearest neighbors. In practice this means Word2Vec is likely to be less "robust," that is, embeddings will tend to be more corpus specific, than GloVe.

Third, for our set of politics queries (and some vocabulary subsetting to improve the quality of the Word2Vec nearest neighbors), the Turing test implied that our human raters are on average indifferent between the two models.

ADVICE TO PRACTITIONERS

Here we report the main takeaways for practitioners looking to use word embeddings in their research. First, in terms of choice hyperparameters in applied work:

10. We have made these publicly available, and these may be downloaded via the project's GitHub page.

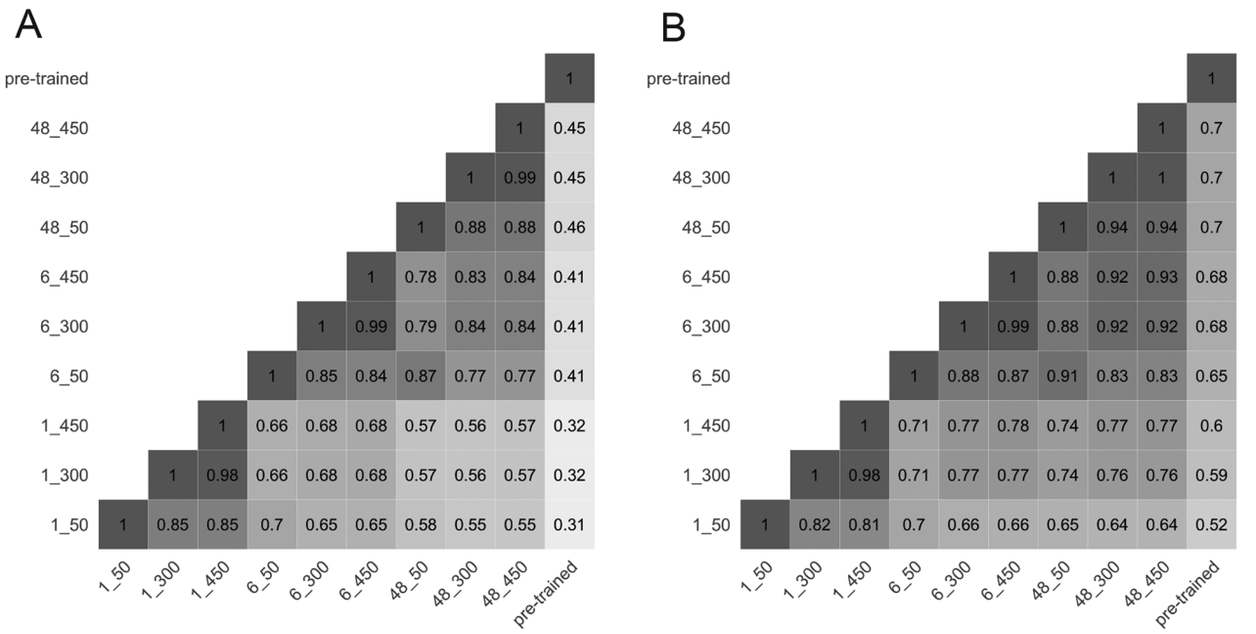

Figure 5. Query search ranking criteria: different models produce similar embeddings, especially for substantive "politics" queries. *A*, Random queries. *B*, Politics queries.

- *Window size and embedding dimensions*: With the possible exception of small corpora like the State of the Union speeches, one should avoid using very few dimensions (below 100) and small window sizes (<5), especially if interested in capturing topical semantics. If one cares about syntactic relationships, then the model choice should be based on that criterion first (i.e., small windows may be desirable). While performance improves with larger window sizes and more dimensions, both exhibit decreasing returns—improvements are marginal beyond 300 dimensions and window size of 6. Given the trade-off between more dimension/larger window size and computation time, the popular choice of 6 (window size) and 300 (dimensions) seems reasonable. This particular specification is also fairly stable, meaning one need not estimate multiple runs to account for possible instability.

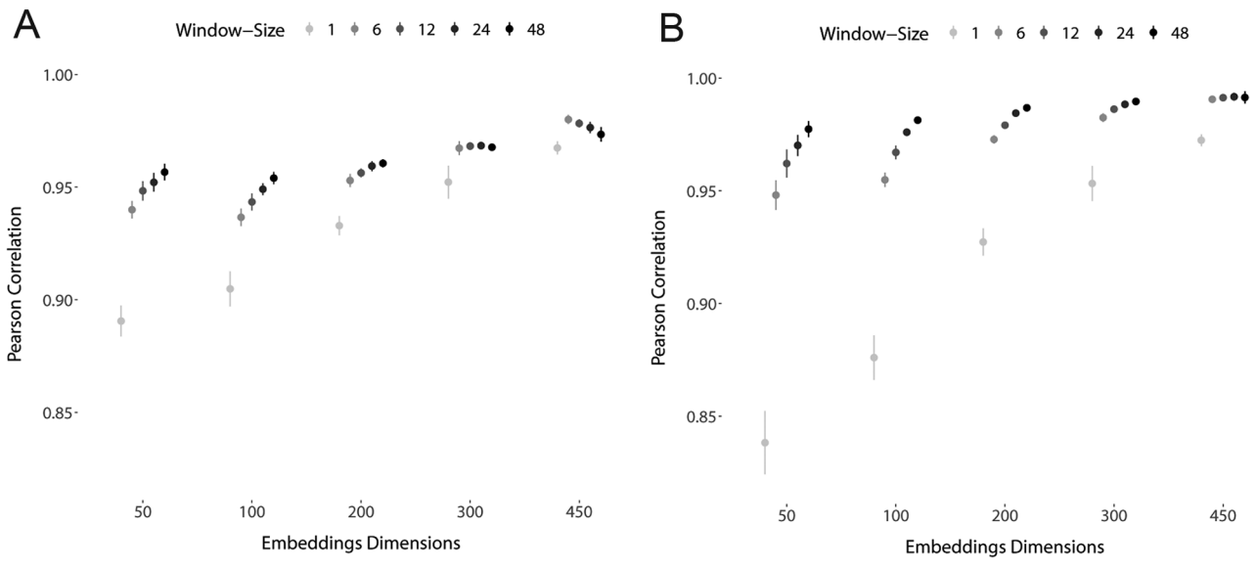

Figure 6. Stability criteria: larger models are more stable but with decreasing returns. *A*, Random queries. *B*, Politics queries

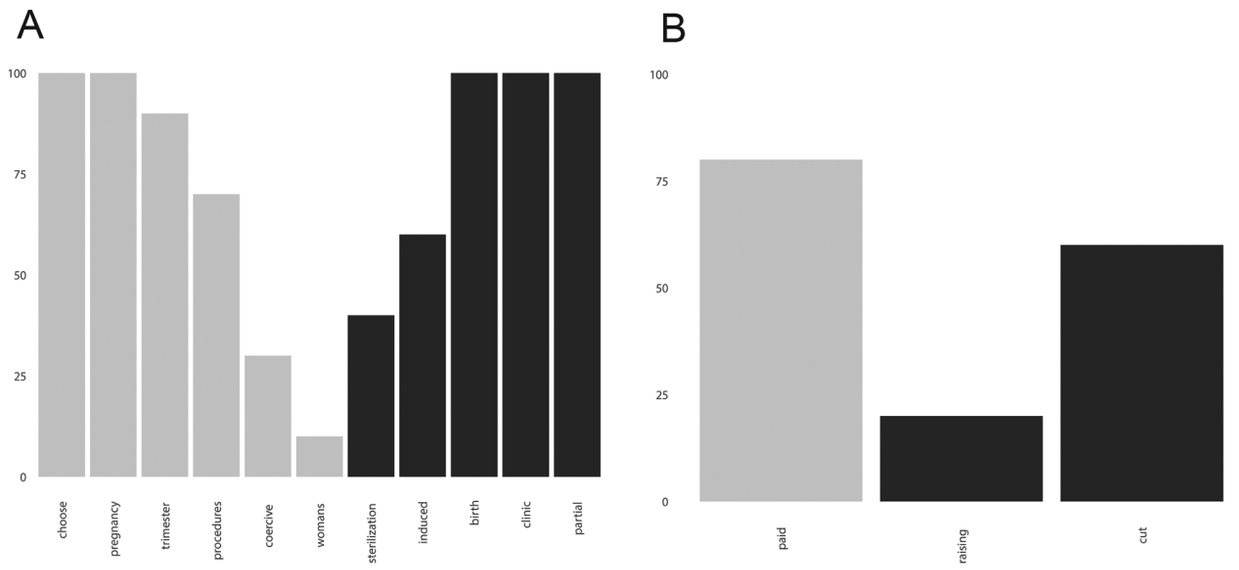

Figure 7. Nearest neighbors most common to each party. Democrat = light gray; Republican = dark gray. *A*, Abortion. *B*, Taxes

- *Pretrained versus local embeddings*: GloVe pretrained embeddings generally exhibit high correlations (>0.4 for the set of random queries and >0.65 for the set of curated queries) with embeddings trained on our selection of political corpora.[11] At least for our focus *Congressional Record* corpus, there is little evidence that using pretrained embeddings is problematic for subdivisions of the corpus by party—Republican versus Democrat speech. Human coders generally prefer pretrained representations, but not for every term, and it is quite close for many prompts. Specifically, GloVe pretrained word embeddings achieve on average—for the set of political queries—80% of human performance and are generally preferred to locally trained embeddings. Where they do prefer pretrained embeddings, it is apparently due to (some) local ones simply requiring too much specialized domain knowledge. Thus, if a researcher has a particularly technical or specialized understanding of a term (beyond something that could be easily communicated to a human coder), they should use locally trained embeddings.These results suggest that embeddings estimated on large online corpora (e.g., Wikipedia and Google data dumps) can reasonably be used for the analysis of contemporaneous political texts. Moreover, the output of our local models is also reasonably liked by our human raters though they prefer the pretrained models.

11. This is lower in the case of small corpora like the State of the Union and in the case of random queries for the Spanish corpus.

Second, in terms of methodology lessons on how to evaluate models:

- *Query search*: In the absence of a clearly defined evaluation metric—a downstream task with labeled data—embeddings can be compared in terms of how they "organize" the embedding space. We propose doing so using query search ranking correlations for a set of randomly selected queries and—given a specific domain of interest—a set of representative domain-specific queries. To discriminate between models resulting in very different embedding spaces, both can be compared to a baseline, either a model known to perform well or, as we do, a human baseline.
- *Crowdsourcing*: Crowdsourcing provides a relatively cheap alternative to evaluate how well word embedding models capture human semantics. We had success with a triad task format, a choice task with an established track record and solid theoretical grounding in psychology.
- *Human Turing test*: A given embeddings model—or any model of human semantics for that matter—can be said to approximate human semantics well if, on average, for any given cue, the model generates associations (nearest neighbors) that a human cannot systematically distinguish from human-generated associations. Specifically, we define human performance as the point at which a human rater is on average indifferent between a computer and a human-generated association.

Third, in terms of instability:

- *Stability*: Word embedding models have nonconvex objective functions—can have multiple locally optimal solutions. This produces additional variability beyond sampling error that, if unaccounted for, can lead to mistaken and nonreplicable inferences. To account for estimation-related instability we endorse estimating the same model several times, each with different randomly drawn initial word vectors and use an average of the distance metric of choice. The good news, from our results at least, is that embeddings that perform well on the human and other metrics also tend to be the most stable.

Fourth, in terms of algorithm (GloVe or Word2Vec [skip-gram]):

- *GloVe versus Word2Vec (skip-gram)*: Although GloVe is mathematically very similar to Word2Vec's skip-gram architecture, in practice they will diverge, often quite substantially, in their mapping of the semantic space. Word2Vec benefits from a more careful filtering of the vocabulary (e.g., increasing the minimum count or setting a lower maximum number of words in the vocabulary) as it tends to overweight relatively rare terms (often misspellings). Once Word2Vec has an appropriately filtered vocabulary, it performs as well as GloVe with human raters.

DISCUSSION OF RESULTS

Why do we get the results we do? That is, why are pretrained embeddings sometimes preferred to locally fit ones given that the latter are domain specific? And why do humans sometimes prefer human-created neighbors but sometimes prefer those generated by a statistical model?

On the issue of poor local fits, one possibility is simply a lack of data. That is, corpora being used for such fits are too small to exhibit the helpful smoothing that a very large corpus (like Wikipedia) would allow. Thus, even with weighting down rare terms, small corpora have idiosyncratic co-occurrences (perhaps even typos) that are unappealing to our human coders. It is also possible that our crowd workers have strong preexisting beliefs or opinions about our political prompts. Suppose, for example, that the majority of crowd workers are (strong) Democrat partisans and that therefore their priors regarding the nearest neighbors of Republican are systematically different from those of a more removed expert coder. We expect these crowd workers—relative to the researcher—to be skeptical of the nearest neighbors from a model trained on the relatively abstruse language found in the *Congressional Record*. Related, above we noted that on some particular cues, it was the case that crowd workers did not much like model suggestions that required highly specific domain knowledge. To reiterate a point we made earlier, though, we assume that such effects are small; in any case, our present focus is not on human-subjects research (in which our crowd workers become subjects of an experiment).

As to the core Turing issue—that humans sometimes prefer model output rather than that of other humans—we suspect that this is fundamentally connected to issues of sampling. Even though we remove outlier human suggestions, it may be the case that a model aggregating over millions of words is more reasonable, on average. Meanwhile, one pathology of embeddings is that they can quickly become out of date (e.g., until recently "Trump" would be a word with nearest neighbors pertaining to real estate or casinos, rather than the presidency).

Finally, is there any evidence that an end user would suffer in terms of the merits of their study should they choose the wrong model specification? This is beyond the scope of the current paper, but in appendix G we give an example of "negative" consequences.

CONCLUSIONS

Word embeddings in their modern scalable form have captured the attention of industry and academia—including social science. As with all methodological advances, it is vital that we understand what they can do for us and what they cannot.

For our domain, we have good news: by all the technical and substantive criteria we used, off-the-shelf pretrained embeddings work very well relative to—and sometimes better than—both human coders and more involved locally trained models. Furthermore, locally trained embeddings perform similarly—with exceptions—across specifications. This should reduce end-user angst about their parameter choices. The general form of these findings extends to historical and non-English texts. Finally and with caveats, GloVe and Word2Vec both enjoy similar performance as rated by human coders.

We dealt with a broad but necessarily limited number of possible options. This includes the nature of our text data, which are elite-level (mostly preprepared) speeches—not letters, tweets, or free-form communication. Of course, other researchers will care about different concepts and specifications. Irrespective of those particularities however, our work flow will be useful. Finally, of course, we have focused on relative performance: we have not studied whether embeddings are interesting or useful per se for understanding behavior, events, and so on. We leave such questions for future work.

ACKNOWLEDGMENTS

We would like to thank audiences at the following conferences and seminars for helpful feedback: Polmeth (2019), APSA (2019), Comptext (2020), Atlantic Canada (2020), Binghamton University, Harvard University, University of Michigan, University of Montreal, University of Notre Dame, Princeton University, and Northeastern University. Jacob Montgomery and Brandon Stewart provided valuable written comments. We also thank our anonymous reviewers for their feedback.

REFERENCES

Antoniak, Maria, and David Mimno. 2018. "Evaluating the Stability of Embedding-Based Word Similarities." *Transactions of the Association for Computational Linguistics* 6:107–19.

Benoit, Kenneth, Drew Conway, Benjamin E. Lauderdale, Michael Laver, and Slava Mikhaylov. 2016. "Crowd-Sourced Text Analysis: Reproducible and Agile Production of Political Data." *American Political Science Review* 110 (2): 278–95.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. "Latent Dirichlet Allocation." *Journal of Machine Learning Research* 3 (January): 993–1022.

Chiu, Billy, Anna Korhonen, and Sampo Pyysalo. 2016. "Intrinsic Evaluation of Word Vectors Fails to Predict Extrinsic Performance." In Omer Levy, Felix Hill, Anna Korhonen, Kyunghyun Cho, Roi Reichart, Yoav Goldberg, and Antoine Bordes, eds., *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*. Berlin: Association for Computational Linguistics, 1–6.

Collobert, Ronan, and Jason Weston. 2008. "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning." In William Cohen, Andrew McCallum, and Sam Roweis, eds., *Proceedings of the 25th International Conference on Machine Learning*, ACM. New York: Association for Computing Machinery. 160–67.

Denny, Matthew J., and Arthur Spirling. 2018. "Text Preprocessing for Unsupervised Learning: Why It Matters, When It Misleads, and What to Do about It." *Political Analysis* 26 (2): 168–89.

Diermeier, Daniel, Jean-François Godbout, Bei Yu, and Stefan Kaufmann. 2012. "Language and Ideology in Congress." *British Journal of Political Science* 42 (1): 31–55.

Faruqui, Manaal, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. "Problems with Evaluation of Word Embeddings Using Word Similarity Tasks." arXiv preprint arXiv:1605.02276.

Firth, John Rupert. 1957. *Studies in Linguistic Analysis*. Oxford: Wiley-Blackwell.

Gentzkow, Matthew, J. M. Shapiro, and Matt Taddy. 2018. "Congressional Record for the 43rd–114th Congresses: Parsed Speeches and Phrase Counts." https://data.stanford.edu/congress_text.

Halpern, David, and Pedro Rodriguez. 2018. "Partisan Representations: Partisan Differences in Semantic Representations and Their Role in Attitude Judgments." In Charles Kalish, Martina Rau, Jerry Zhu, and Timothy T. Rogers, eds., *Proceedings of the 40th Annual Conference of the Cognitive Science Society*. Madison, WI: Cognitive Science Society. 445–50.

Hamilton, William L., Jure Leskovec, and Dan Jurafsky. 2016. "Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change." arXiv preprint arXiv:1605.09096.

Khodak, Mikhail, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 2018. "A la Carte Embedding: Cheap but Effective Induction of Semantic Feature Vectors." arXiv preprint arXiv:1805.05388.

Kozlowski, Austin C., Matt Taddy, and James A. Evans. 2018. "The Geometry of Culture: Analyzing Meaning through Word Embeddings." arXiv preprint arXiv:1803.09288.

Mebane, 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?" Working paper.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. "Distributed Representations of Words and Phrases and Their Compositionality." In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds., *Advances in Neural Information Processing Systems*. Lake Tahoe, NV: Neural Information Processing Systems, 3111–19.

Monroe, Burt L., Michael P. Colaresi, and Kevin M. Quinn. 2008. "Fightin' Words: Lexical Feature Selection and Evaluation for Identifying the Content of Political Conflict." *Political Analysis* 16 (4): 372–403.

Montgomery, Jacob M., and Santiago Olivella. 2018. "Tree-Based Models for Political Science Data." *American Journal of Political Science* 62 (3): 729–44.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. "Glove: Global Vectors for Word Representation." In Alessandro Moschitti, Bo Pang, and Walter Daelemans, eds., *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha: Association for Computational Linguistics, 1532–43.

Quinn, Kevin M., Burt L. Monroe, Michael Colaresi, Michael H. Crespin, and Dragomir R. Radev. 2010. "How to Analyze Political Attention with Minimal Assumptions and Costs." *American Journal of Political Science* 54 (1): 209–28.

Rheault, L., K. Beelen, C. Cochrane, and G. Hirst. 2016. "Measuring Emotion in Parliamentary Debates with Automated Textual Analysis." *PLOS One* 11 (12).

Rheault, Ludovic, and Christopher Cochrane. 2019. "Word Embeddings for the Analysis of Ideological Placement in Parliamentary Corpora." *Political Analysis* 28 (1): 112–33.

Roberts, Margaret E., Brandon M. Stewart, Dustin Tingley, Christopher Lucas, Jetson Leder-Luis, Shana Kushner Gadarian, Bethany Albertson, and David G. Rand. 2014. "Structural Topic Models for Open-Ended Survey Responses." *American Journal of Political Science* 58 (4): 1064–82.

Rodman, Emma. 2019. "A Timely Intervention: Tracking the Changing Meanings of Political Concepts with Word Vectors." *Political Analysis* 28 (1): 87–111.

Rudolph, Maja, Francisco Ruiz, Susan Athey, and David Blei. 2017. "Structured Embedding Models for Grouped Data." In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., *Advances in Neural Information Processing Systems*. Long Beach, CA: Neural Information Processing Systems. 251–61.

Schnabel, Tobias, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. "Evaluation Methods for Unsupervised Word Embeddings." In Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims, eds., *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon: Association for Computational Linguistics. 298–307.

Turing, Alan. 1950. "Computing Machinery and Intelligence." *Mind* 59 (236): 433.

Zhang, Han, and Jennifer Pan. 2019. "CASM: A Deep-Learning Approach for Identifying Collective Action Events with Text and Image Data from Social Media." *Sociological Methodology* 49 (1): 1–57.