How to Train Your Stochastic Parrot: Large Language Models for Political Texts*

Joseph T. Ornstein† Elise N. Blasingame‡ Jake S. Truscott§

April 14, 2023

Abstract

Large language models pre-trained on massive corpora of text from the Internet have transformed the way that computer scientists approach natural language processing over the past five years. But these "foundation models" have yet to see widespread adoption in the social sciences, partly due to their novelty and upfront costs. In this paper, we demonstrate that such models can be effectively applied to a wide variety of text-as-data tasks in political science – including sentiment analysis, ideological scaling, and topic modeling. In a series of pre-registered analyses, this approach outperforms conventional supervised learning methods without the need for extensive data pre-processing or large sets of labeled training data. And performance is comparable to expert and crowd-coding methods at a fraction of the cost. We propose a set of best practices for adapting these models to social science measurement tasks, and develop an open-source software package for researchers.

*The authors thank Jeff Milliman for research assistance, as well as Nick Beauchamp, Sam Bestavater, Michael Burnham, Gary King, Matt Ryan, and Brandon Stewart for their helpful comments on earlier drafts.

†Assistant Professor of Political Science, University of Georgia

‡PhD Candidate, Political Science, University of Georgia

§Postdoctoral Fellow, Center for CSPAN Scholarship & Engagement, Purdue University

1 Introduction

Most of the information that political scientists use in our work is stored not in tidy numeric datasets, but in text. The practice of politics, in all its myriad forms, is ultimately recorded in constitutions, television transcripts, party manifestos, social media posts, email correspondence, treaties, conversations, news articles, speeches, codes of ordinances, campaign ads, city council minutes, judicial decisions, presidential debates, and other unstructured texts. One of the most exciting recent developments in our field has been the explosion of new methods for systematically understanding these texts as data (Grimmer and Stewart 2013).

These efforts in political science have occurred alongside a parallel revolution in computer science, developing natural language processing tools to usefully interpret human speech. The applications for these methods are numerous – including chat bots, Internet search, auto-complete, and virtual assistants – but remarkably, much of this research has begun to converge on a single solution: large, pre-trained language models built on a neural network architecture called the *transformer* (Vaswani et al. 2017; Bommasani et al. 2021). The most famous of this new class of models include Google's BERT (Devlin et al. 2019) and OpenAI's GPT-3/ChatGPT (Brown et al. 2020).

In this paper, we argue that adapting these large language models (LLMs) to political text-as-data tasks can yield significant gains in performance, cost, and capabilities. Across a range of applications – including sentiment classification, ideological scaling, and topic modeling – we show that these models (in particular, OpenAI's GPT-3) reliably outperform existing automated text classification methods, and produce results comparable to human crowd-coders at a small fraction of the cost.

GPT-3 and its cousins are deep learning models (LeCun, Bengio, and Hinton 2015) trained to perform "next word" prediction tasks. When provided with a sequence of text, the model generates a probability vector for the most likely words[1] to follow that sequence. For example, when prompted with the phrase "Thank you. Have a nice", GPT-3 estimates that there is an 88.6% probability the next word will be "day", a 2.4% probability the next word will be "weekend", a 1.9% probability it will be "evening" and so on.

Though these models are essentially "stochastic parrots" mimicking human speech patterns observed in their training corpus (Bender et al. 2021), they have demonstrated a number of surprising emergent capabilities that they were not deliberately designed to do. In particular, GPT-3 can be *adapted* to perform various text-as-data tasks by carefully crafting its input. Consider the following prompt, which reformulates a sentiment classification task as a next word prediction problem:

```
Decide whether a Tweet's sentiment is positive, neutral, or negative.

Tweet: Congratulations to the SCOTUS. American confidence in the
Supreme Court is now lower than at any time in history. Well done!

Sentiment:
```

[1] Technically, GPT-3 represents text as "tokens", strings of roughly four characters, rather than words, but for illustrative purposes we can think of it as operating at the word-level.

Conventional approaches to sentiment classification struggle to accurately label texts like these, which use positive words to convey a negative sentiment (e.g. "congratulations", "confidence", "supreme", "well done!"). But when supplied with this prompt, GPT-3 estimates a 77% probability that the next word will be "Negative". We argue that this classification can be used to construct a measure of sentiment, and in the applications that follow, we validate this approach by converting several common political text-as-data tasks into next word prediction problems.

This approach to modeling is fundamentally different than the one familiar to most political scientists. Rather than fitting a separate model for each research question ("one-to-one"), we take a single pre-trained language model and adapt it to several different tasks ("one-to-many"). The promise of the approach is in the scale and complexity that a single pre-trained model can offer. GPT-3 is a massive neural network composed of 175 billion parameters, trained on hundreds of billions of words of text from the Internet and digitized books (Brown et al. 2020). Because the model is orders-of-magnitude more complex than an individual political scientist could train, there is good reason to believe that for certain tasks it can outperform "bespoke" models trained for a specific purpose.

The paper proceeds as follows. In the next section, we describe the basic architecture of these models, and how they make use of innovations like self-supervised learning and contextualized word embeddings to generate their predictions. In section three, we describe four applications of the models to common political text-as-data tasks, including sentiment analysis of social media posts, classifying the tone of political advertisements, ideological scaling of party manifestos, and topic modeling of US Congressional floor speeches. Based on lessons learned from these applications, we propose a set of best practices for prompt design and develop an open-source software package to implement our suggested approach.[2] We conclude with a discussion on whether the performance gains we document are worth the potential dangers associated with unprincipled use of these models (Abid, Farooqi, and Zou 2021; Bender et al. 2021; Strubell, Ganesh, and McCallum 2019). Throughout, we emphasize the importance of "validation, validation, validation" (Grimmer and Stewart 2013), repeatedly comparing model outputs against human judgment to ensure they are measuring what we want them to measure.

2 Foundation Models

Collectively, the field has yet to agree on a name for this new class of models, but we're partial to the term coined by Bommasani et al. (2021) – "foundation models". Instead of being trained for a particular task, these models are pre-trained on next word prediction, serving as a "foundation" that political scientists can then adapt to their own research purposes. Foundation models like GPT-3 combine two recent innovations in natural language processing. The first is *self-supervised learning*, which permits the models to be trained on a massive set of language data. The second is *contextualized word embeddings*, which flexibly represent the meaning of words depending on their context.

[2]The `text2data` package in `R`, currently available on GitHub. We include a brief vignette with setup instructions in Appendix A.

2.1 Self-supervision

A central difficulty for supervised learning methods is the need to collect and annotate large amounts of training data (Wilkerson and Casas 2017). If, for example, a researcher wants to train a model to predict the tone of political advertisements, they must first have a dataset with thousands of labeled political ads. The process of hand-labeling these data can be expensive and time-consuming, even with non-expert crowd-sourced approaches (Benoit et al. 2016; Carlson and Montgomery 2017).

By contrast, a *self-supervised* learning task is one where the target prediction is provided within the data itself, rather than hand-labeled by a researcher. One reason why models like GPT-3 are trained to perform next word prediction is that this can be done in a self-supervised fashion. Every sentence of text that a human has ever written can be split into a sequence of tokens and used to train the model, which permits a massive expansion in the amount of training data available. Rather than training a supervised learning model on a few thousand documents, GPT-3 can be trained on hundreds of billions of words scraped from the Internet and a digitized corpus of books. In turn, more training data allows for more complex models, with more nuanced representations of human language.

2.2 Contextualized Word Embeddings

When analyzing text as data, one must first decide how to represent a text numerically. Conventional "bag of words" approaches (Grimmer, Roberts, and Stewart 2022, chap. 8) represent each document as a vector of word frequencies. The main drawback of this representation is that it assumes each word has a unique meaning. Mathematically, the document-feature matrix is extremely *sparse*; there are hundreds of thousands of unique words, but many of them have overlapping meaning.

Instead, language models like GPT-3 represent each word in a document as a high-dimensional vector, an approach known as "word embeddings". This representation attempts to retain information about the *meaning* of words by encoding how often a word is used in the vicinity of other related words – motivated by John Firth's linguistic maxim, "you shall know a word by the company it keeps" (Firth 1957; Rodriguez and Spirling 2021). For example, in a sufficiently large corpus of text, you will discover that the word "cat" appears more frequently than you would expect by chance near words like "litter", "yarn", "claws", etc. You'd also find that the word "kitten" appears more frequently than you would expect by chance near those words. A word embedding, incorporating this information, represents "cat" and "kitten" with vectors that are close together in space. By training word embeddings on a large corpus of texts, one can transfer knowledge about the meaning of words from a more general corpus to a specific text-as-data problem.

But this approach to representing meaning can sometimes fall short, because there are many words whose meaning is ambiguous without context. Consider, for instance, the word "bill", which could have one of several meanings, depending on whether it is preceded by the phrase "signed the…", "foot the…", or "Hillary and…". Other common words, like the pronouns "it" or "they", are entirely meaningless without context. For such words, a single pre-trained word embedding is unlikely to capture meaning very well.

When humans are interpreting words in a sentence, we start with our "pre-trained" idea of what a word means, then adjust that interpretation on the fly as we read the word in context (like you just did with the word "fly"). This is the insight behind *contextualized word embeddings*, which allow a word's vector representation to change depending on what words precede or follow it. The key innovation which models like GPT-3 use to generate contextualized word embeddings is called the *transformer* (Vaswani et al. 2017). The transformer is a neural network architecture containing many hidden layers of "self-attention", which recompute each word's embedding as a weighted average of nearby word embeddings. This allows the model to learn, for instance, the meaning of the word "it". It is probably close in the vector space to a previous noun in the sentence.

Building on the transformer architecture, there has been rapid progress in natural language processing over the past five years. In 2016, the best-performing language model scored an F (59.3%) on the 8th grade New York Regents Science Exam. By 2019, a large language model based on the transformer architecture scored 91.6% (Clark et al. 2021). For political scientists, the practical advantage of contextualized word embeddings is that, by better representing the nuance and ambiguity of political speech, they can outperform existing "bag of words" methods at classifying, measuring, and discovering patterns in political texts. We turn next to a few examples of this approach in practice.

3 Applications

We assess the performance of GPT-3 on a set of common political science text-as-data tasks, including sentiment analysis, ideology classification, and topic modeling. In our first application, we classify the sentiment of a novel set of social media posts related to Supreme Court rulings between 2018 and 2020, comparing GPT-3's classifications against other automated methods for sentiment analysis. Next, we classify the tone of US political ads from Carlson and Montgomery (2017), comparing the performance of GPT-3's classifications against crowds of human coders. Our third application replicates the ideological scaling of political manifestos conducted by Benoit et al. (2016) via crowd-coding. And for our final application we assign topic labels to 9,704 one-minute floor speeches from the US House of Representatives (Wilkerson and Casas 2017). These four applications provide us with a varied set of tasks and subjects with which to evaluate the performance of the large language model approach.

For each application, we pre-registered an analysis plan for how we would adapt GPT-3 to document classification.[3] We took this step to ensure that we do not overstate the performance of our approach by iteratively refining the model in search of the best fit. We describe the design, approach, and outcomes for each application in the following subsections.

3.1 Sentiment Analysis of Political Tweets

Classifying sentiment on social media is a notoriously difficult problem for computational methods. Dictionary-based methods, which measure sentiment by counting the frequency of positive and negative words, tend to perform poorly when faced with text where positive words imply negative sentiment ("thanks for nothing", "smooth move", "way to go, genius") and negative

[3]See AsPredicted document numbers #92341, #92422, #92666, #100718, and #125217.

words imply positive sentiment ("that was wicked/sick/demented!"). And supervised learning methods using a bag of words approach, even when trained on millions of tweets, can at best correctly classify the sentiment of a test set roughly 80% of the time (Go, Bhayani, and Huang 2009). In such an environment, the ability of contextualized word embeddings to flexibly adjust their representation of a word in response to its context may be quite beneficial, and there has been promising research in political science applying transformer-based models to the task of sentiment analysis (Widmann and Wich 2022).

To test the performance of this approach, we collect a novel dataset of 945 Twitter posts ("tweets") referencing the United States Supreme Court within 24 hours of two controversial opinions: *Masterpiece Cakeshop, Ltd. v. Colorado Civil Rights Commission* (2018), as well as the Court's concurrently released opinions in *Trump v. Mazars* and *Trump v. Vance* (2020). We chose these cases to reflect a diverse set of users and political issues, including anti-discriminatory practices towards same-sex couples, religious liberties for private business owners, and the legal immunity of Donald Trump as both the president and a private citizen. For each tweet in this dataset, we created an author-labeled sentiment score through a two-stage manual coding procedure. The three authors began by independently labeling a set of 1,000 tweets as Positive, Negative, or Neutral. From that original set, we excluded 55 tweets that were unrelated to the US Supreme Court decisions, and conducted a second round of manual labeling for any tweets where at least two authors disagreed about the direction of sentiment. The result is a measure of sentiment ranging from -1 (all authors agreed the tweet was negative) to +1 (all authors agreed the tweet was positive).[4]

As of writing, there exist four variants of the GPT-3 model, each with a different number of parameters. The smallest has 2.7 billion parameters, followed by a 6.7 billion parameter variant, a 13 billion parameter variant, and finally the aforementioned 175 billion parameter variant. To explore how the size and complexity of the model affects its classification performance, we iteratively tested each of these GPT-3 variants using prompts structured like the one in Table 1.

Table 1. GPT-3 Prompt for sentiment classification task

```
Decide whether a Tweet's sentiment is positive, neutral, or negative.

Tweet: scotus will hand down its decision this morning
Sentiment: Neutral

---

Tweet: [text]
Sentiment:
```

When designing such a prompt, the researcher must make several choices. First, the prompt may include a set of instructions describing the classification task (the "prefix" or "preamble"). Although these instructions can be omitted, our analysis finds that richly detailed instructions can significantly improve performance compared to a "default" preamble like the one in Table 1. Next, a prompt can include one or more completed examples. The prompt in Table 1 is known

[4]Inter-coder reliability as measured by Fleiss' kappa was 0.72, and at least two authors agreed on the label for every tweet.

as a "one-shot" prompt, because it provides a single example of an appropriate response before providing the text to be classified. When designing prompts, a researcher must decide how many (and which) examples to include. In our analyses below, we use this one-shot prompt as well as a zero-shot prompt (no labeled examples) and few-shot prompt (five labeled examples).[5]

For each prompt, GPT-3 outputs the probabilities that the subsequent word will be Positive, Negative, or Neutral. From this probability vector we construct a continuous measure of sentiment for every tweet in the dataset (per our pre-registration protocol, we take the first component of a principal component analysis).

Table 2 reports the performance of this measure across every combination of model and prompt variant. The smallest model variants (2.7b and 6.7b parameters) perform quite poorly, requiring few-shot learning before their output is even modestly correlated with the expert scores. But the two largest variants perform well regardless of prompt design choices. The 175-billion parameter GPT-3 predicts whether a tweet was negative or positive in 87.5% of cases with one-shot learning, and 88.4% of the time with few-shot learning. The 13-billion parameter variant performs nearly as well, with accuracies of 85.1% for one-shot learning and 87.4% for few-shot learning.[6] For comparison, the best dictionary-based method we could construct only classified 38.3% of the tweets correctly, and a Naive Bayes classifier trained on 1.2 million tweets from the Go, Bhayani, and Huang (2009) dataset classified 57.7% correctly – barely better than a coin flip. Figure 1 illustrates the performance of these existing methods compared to the large language model approach.

[5]For the text of the zero-shot and few-shot variants of the prompt, see Appendix B. For each application, we select a hold-out sample of few-shot examples from the larger dataset, and do not include these examples when computing validation statistics.

[6]Choosing a less-capable variant of the model may be advantageous for some researchers, since (as of October 2022) these models are only available through OpenAI's paid Application Programming Interface (API), and the per-token rate for 13-billion parameter variant is ten times less expensive than the full 175-billion parameter variant. For a dataset this size, however, the costs were minimal. In total, coding our 945 tweets cost $4.86 with one-shot adaptation and 175b parameters, versus $0.46 with 13b parameters.

Table 2. Performance on sentiment classification task by prompt and model variant

GPT-3 Variant (No. Parameters)	Prompt	Correlation with Expert Score (ρ)	Alternative Classifier	Correlation with Expert Score (ρ)
GPT-3 (175B)	Few-Shot	0.76	BING	0.29
	One-Shot	0.76		
	Zero-Shot	0.71	BING (Custom)	0.29
GPT-3 (13B)	Few-Shot	0.71	`sentimentr`	0.31
	One-Shot	0.67		
	Zero-Shot	0.63		
GPT-3 (6.7B)	Few-Shot	0.47	Naïve Bayes	0.29
	One-Shot	0.63		
	Zero-Shot	-0.02		
GPT-3 (2.7B)	Few-Shot	0.3		
	One-Shot	0.01		
	Zero-Shot	0.02		

To see why the GPT-3 classifier so dramatically outperformed other methods of automated sentiment classification, consider Table 3, which presents a sample of tweets where GPT-3 and conventional approaches yielded different classifications.[7] Each tweet conveys a negative sentiment using words that a sentiment dictionary would code as positive (e.g. "sober", "nuanced", "celebrate", "excited"). This is a hallmark of sarcastic language, and it is encouraging to see that the LLM approach can handle such difficult cases.

[7]We fit a Naive Bayes classifier to the Go, Bhayani, and Huang (2009) dataset, as well as three different dictionary-based sentiment classification approaches, all of which underperformed the GPT-3 classifier. See Appendix D for a description of these methods and their results.

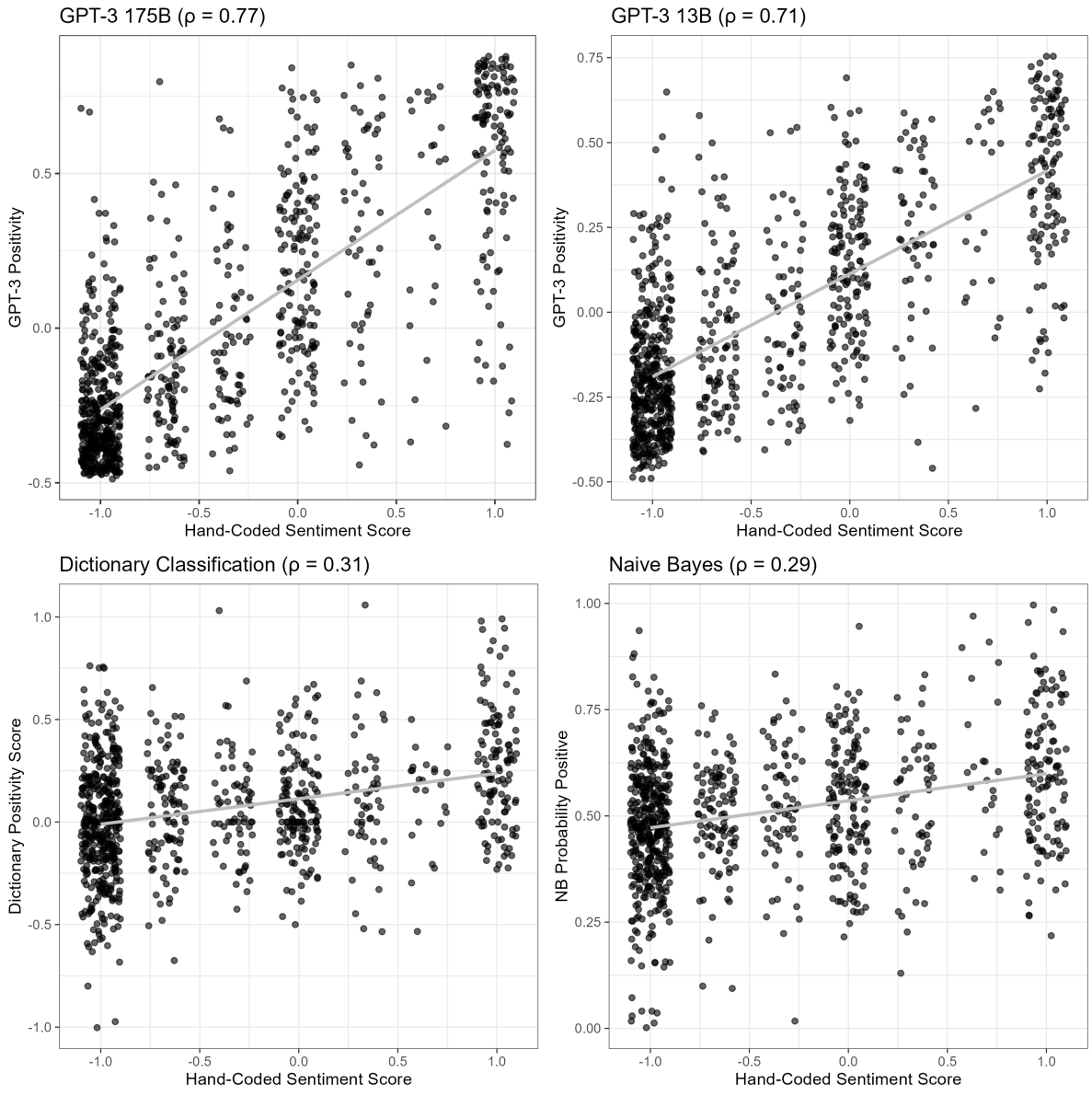


Figure 1: Classification performance on Twitter sentiment task, comparing two GPT-3 variants, dictionary classification, and a supervised learning method (Naive Bayes). For more details on the latter two procedures, see Appendix D.

Table 3. Sample of tweets with ambiguous language

Tweet	Expert Label	GPT-3	Dictionary Classifier	Naive Bayes
I'm sure the news media will provide sober, nuanced analysis of Masterpiece Cake Shop. #SCOTUS	Negative	Negative	Positive	Positive
Gonna have a drink at "the low bar" to celebrate SCOTUS ruling that laws sometimes apply to the president.	Negative	Neutral	Positive	Positive
SCOTUS kicked the can down to [sic] road!	Negative	Negative	Neutral	Negative
What a nice Supreme Court decision to lead off pride month. So excited to see all this legal discrimination about to happen. Want to thank two of my heroines RBG and Sotomayor for dissenting.	Negative	Negative	Positive	Positive

The tweets that GPT-3 misclassifies tend to require contextual knowledge of the Supreme Court cases to accurately classify, often combined with some form of conflictual or sarcastic rhetoric.[8] For example, consider the following tweet posted in response to *Mazars* and *Vance*: "The Supreme Court has decided. @realDonaldTrump is #NotAboveLaw. The American people deserve to see Trump's tax returns and we demand their release now!" The authors agreed that the tweet was expressing a positive sentiment in support of the Court's ruling. But, lacking that contextual understanding of the case, GPT-3 classified the tweet as negative. Or consider this tweet following Court's decision in *Masterpiece Cakeshop* (2018): "Way to go SCOTUS! You really celebrated PRIDE Month." All of the authors agreed that this was a sarcastic remark (as the Court's decision allowed the eponymous Masterpiece Cakeshop to refuse service to a same-sex couple), but GPT-3 returned a 41% probability classifying the tweet as positive, and a 35% probability as negative. Accurately classifying such a tweet would require contextual knowledge about the outcome observed in the case, which GPT-3 lacks when provided with the tweet alone.

In a second pre-registered analysis, we explore whether providing GPT-3 with additional context about the Supreme Court cases improves classification performance for these ambiguous tweets. For this experiment, we wrote two prompts, one to classify tweets collected after the *Masterpiece Cakeshop* decision and the other to classify tweets following the *Mazars* and *Vance* decisions. Both prompts include a brief preamble describing the Supreme Court's ruling, followed by six few-shot example completions. Each example was drawn from the set of tweets that the authors unanimously coded – two positive, two neutral, and two negative to avoid biasing the model towards a particular classification (Zhao et al. 2021).[9] This approach to prompt design – which

[8]To be fair, such tweets often confused and divided the human coders as well. Of the tweets where the authors unanimously agreed on a classification, GPT-3 correctly classified 94%.

[9]For the complete prompts, see Appendix Table B2.

we call *contextual prompting* – outperforms every other method we attempted. Figure 2 plots the correlation between the resulting measure and our hand-labeled sentiment scores.

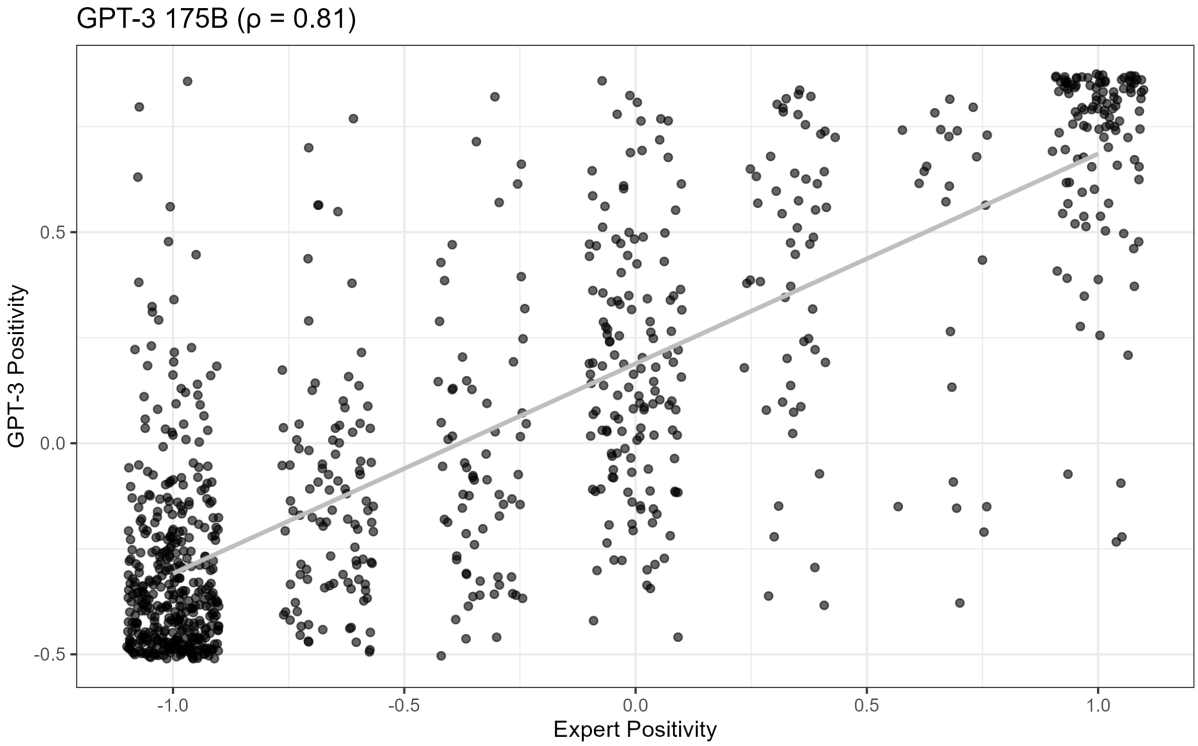


Figure 2: Classification performance on Twitter sentiment task (contextual few-shot prompting)

But the true test of this method is not whether it outperforms sentiment dictionaries, but whether it performs at the level of non-expert human coders. This is the focus of our next two applications.

3.2 Political Ad Tone

Crowd-sourced text analysis is one of the fastest, most reliable methods for manually coding texts, leveraging the "wisdom of crowds" to generate measures from a large collection of non-expert judgments (Surowiecki 2004; Benoit et al. 2016). By asking human coders to conduct a series of pairwise comparisons (e.g. "which of these tweets is more negative?"), Carlson and Montgomery (2017) show that a researcher can quickly generate measures of sentiment that strongly correlate with expert judgments.

Nevertheless, this approach has several shortcomings. First, it requires the researcher to screen, train, and monitor crowd-workers to ensure inter-coder reliability. Second, it can be quite costly. To measure the tone of 935 political ads, Carlson and Montgomery (2017) required 9,420 pairwise comparison tasks at 6 cents per task, for a total cost of $565.20. Although Amazon's Mechanical Turk (*MTurk*) is currently the most economical alternative on the market, that reduced cost is borne by the coders completing the tasks. Studies suggest that people performing "human

intelligence tasks" on sites like *MTurk*, *CrowdFlower*, *Clickworker*, and *Toluna* earn a median hourly wage of roughly $2/hour, and only 4% earn more than the US federal minimum wage of $7.25 (Hara et al. 2018). As a result, crowd-workers have an incentive to quickly complete as many tasks as possible, which can undermine the quality of crowd-sourced measures.

In this application, we replicate Carlson & Montgomery's (2017) measure of political ad tone, using the one-shot prompt in Table 4 with the 175-billion parameter variant of GPT-3. As in the tweet sentiment application, we construct a continuous measure of tone from the model's estimated probability vector.

Table 4. GPT-3 Prompt for Political Ad Tone Task

```
Decide whether the tone of a political advertisement is positive, neutral, or
negative.

Ad Text: [Announcer]: Mark Udall claims he voted for tax cuts 65 times.
[Reporter]: "Actually it's not true."
[Mark Udall]: "I'm just kidding."
Udall claims that now he supports our troops, drilling off shore, bi-partisanship.
[Mark Udall]: "I'm just kidding."
[Announcer]: But through his career he has opposed them all. Udall has a lot of
ads, but a shortage of truth.
[Mark Udall]: "I'm just kidding."
[Announcer]: Mark Udall is not honest about his past and now it's catching up with
him.
The National Republican Senatorial Committee is responsible for the content of this
advertising.
[PFB]: NATIONAL REPUBLICAN SENATORIAL COMMITTEE  (9)

Tone: Negative

---

Ad Text: [text]
Tone:
```

Coding the 935 political ads from Carlson and Montgomery (2017) took less than 2 minutes and cost only $18.46 – a thirty-fold reduction in cost compared to crowd-coding.[10] And yet the measure of tone was even more strongly correlated with expert ratings, as illustrated in Figure 3.

Our measure diverges from the expert ratings in one of two situations. First, there are some ads in the dataset that are quite negative in tone, but the expert coders classify them as positive because they do not attack a *specific* opponent (focusing instead on "typical Washington politicians" or "Republicans"). Second, some ads require contextual knowledge to accurately classify. Table B3 in the Appendix provides some examples of ads where our approach and the experts disagreed. The first two ads are negative in tone, but target "Washington" and

[10]This cost is based on OpenAI's per-token pricing schedule as of May 2022, prior to a threefold reduction in price late in the summer of 2022. For more discussion on the likely trends in costs for Large Language Models relative to human crowd-coders, see the Conclusion.

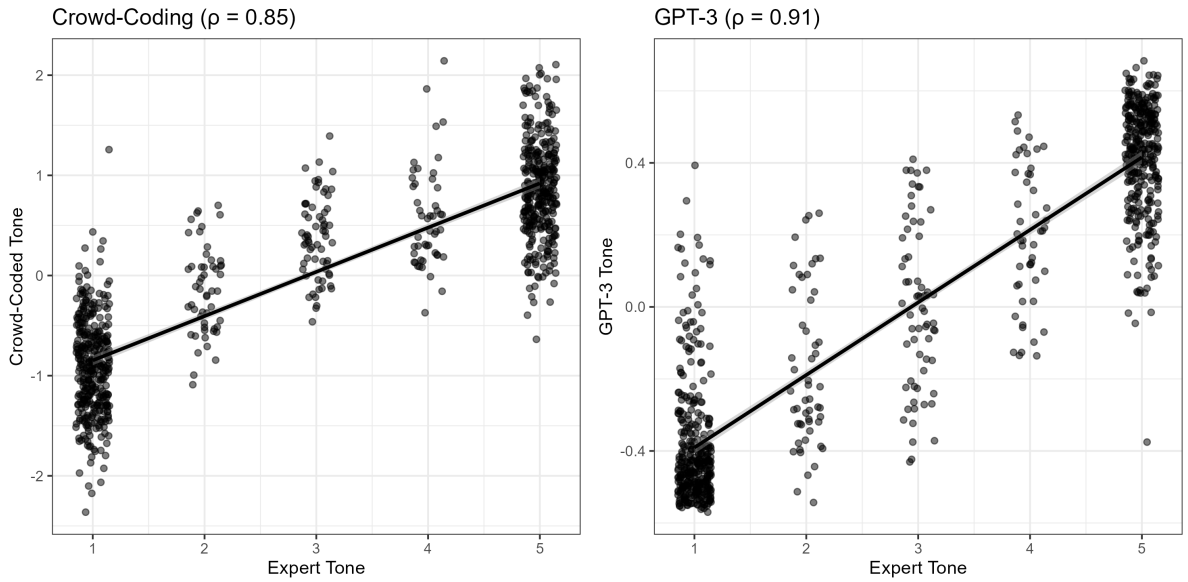


Figure 3: Comparing crowd-coded and GPT-3 estimates to expert-coded political ad tone (Carlson & Montgomery 2017)

"other other side", so are not labeled attack ads by the expert coders. The third ad requires contextual knowledge about Susan Collins' failure to keep a campaign promise, context which is not provided in the ad text itself. And the final ad – illustrated by the point in the lower right corner of Figure 3 – is arguably miscoded by the experts.

3.3 Ideological Scaling

Our third application is a document scaling task, designed to assess whether GPT-3 can accurately rate the ideology of political texts on a common scale. The approach we take is, in essence, to treat GPT-3 as if it were a non-expert human coder, replicating the procedure for crowd-sourcing party manifesto positions from Benoit et al. (2016). This allows us to validate our approach against an extensive set of crowd-coded classifications for 18,263 sentences from 18 British party manifestos written between 1987 and 2010. By replicating these results, we can also test whether the model can be adapted to a very different context than the bulk of its training data, both geographically (Britain instead of the United States) and temporally (up to 35 years ago).

We adhere to the crowd-coding procedure from Benoit et al. (2016) as closely as possible, splitting the manifestos into their component sentences and classifying the policy content and ideology of each sentence using the one-shot prompts in Table 5.

Table 5. GPT-3 Prompts for Ideological Scaling Task

Policy Prompt:

```
Decide whether this sentence from a British political manifesto is about Social
Policy, Economic Policy, or Neither.

Sentence: We will implement a comprehensive strategy for ending low pay, notably by
the introduction of a statutory national minimum wage.
Classification: Economic Policy

---

Sentence: [text]
Classification:
```

Ideology Prompt:

```
Decide whether this sentence from a British political manifesto is Liberal,
Conservative, or Neither.

Sentence: We will implement a comprehensive strategy for ending low pay, notably by
the introduction of a statutory national minimum wage.
Classification: Liberal

---

Sentence: [text]
Classification:
```

As in the sentiment classification tasks, GPT-3 outputs the top five predicted tokens for each passage, along with each token's associated log probability. From these, we classify each passage as Social Policy, Economic Policy, or Neither based on the top log probability. And we assign each passage an ideology score equal to the model's estimated Conservative probability minus its estimated Liberal probability. We aggregate these scores to the manifesto level by taking the average score for Economic Policy passages and the average score for Social Policy passages. Figure 4 plots the performance of the crowd-coded estimates (top panel) and the GPT-3 estimates (bottom panel).

Our estimates are more strongly correlated with expert ratings on the Economic policy dimension ($\rho = 0.92$) than the Social policy dimension ($\rho = 0.8$), and are clearly better at capturing *between-party* variance than *within-party* variance (though note that this is true for the crowd-coded measure as well). Despite these limitations, the GPT-3 approach yields estimates that correlate strongly with human-coded measures at a small fraction of the cost. Crowd-coding 18,263 manifesto sentences cost Benoit et al. (2016) approximately $3,226.[11] By comparison, our estimates from the largest, 175 billion parameter variant of the model cost only $75. This has enormous practical implications, as it allows researchers with a fixed budget to produce estimates for 43 times the number of manifestos using large language models as they could with

[11] Assuming a cost of 1.5 cents per sentence, and a total of 215,107 crowd evaluations (see footnote 32 and Table 1).

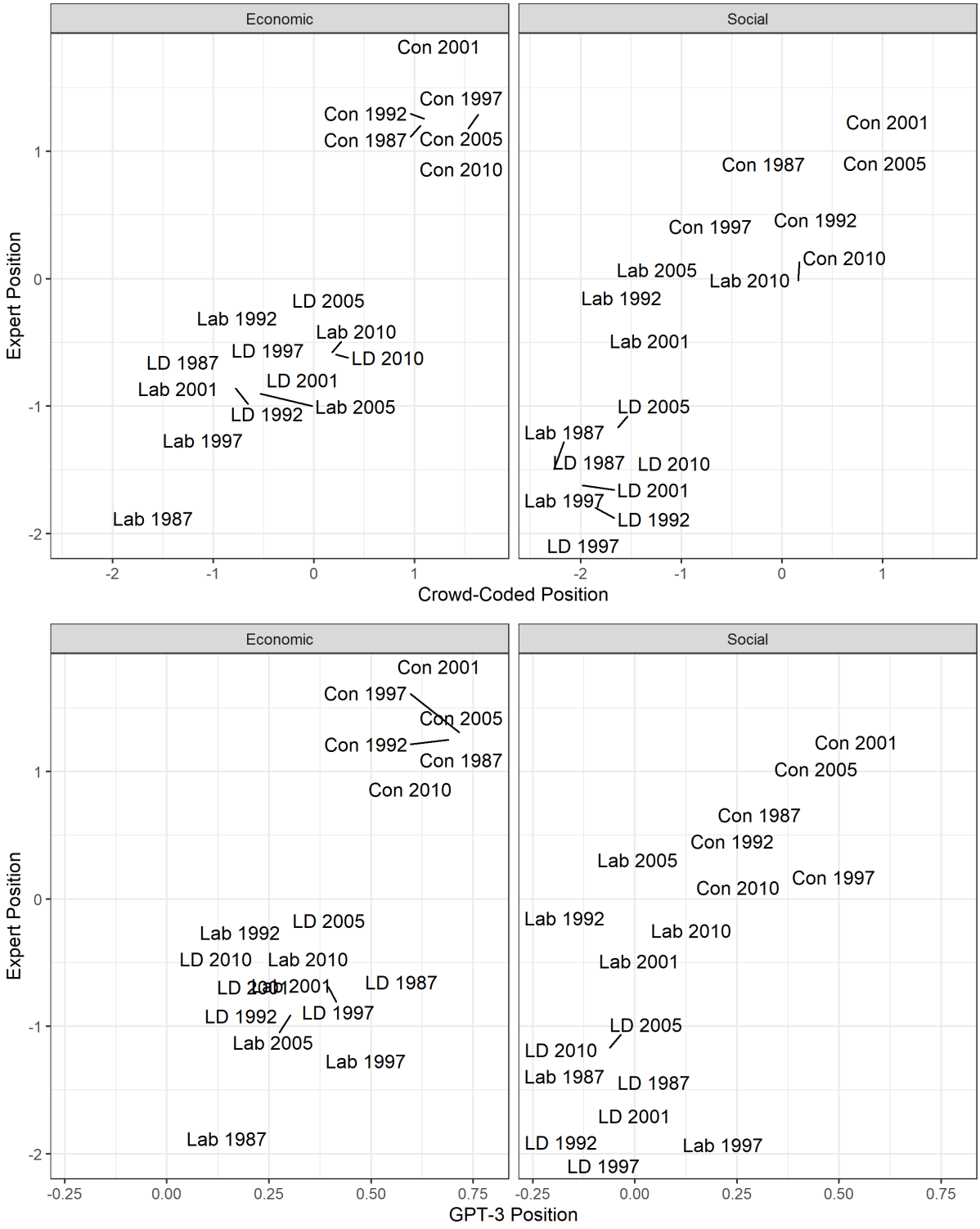


Figure 4: Performance of crowd-coded (top panel) and GPT-3 (bottom panel) ideology estimates, compared to expert judgments.

human crowd-coders.

3.4 Topic Modeling

As useful as these models are for measurement and classification, they may hold even more promise as a tool for discovery (Grimmer, Roberts, and Stewart 2021). Often a researcher will approach a new corpus of documents without a preconceived notion about how to partition them into categories. For any given corpus, there is an unfathomably large number of possible partitions, and statistical models can aid in the process of discovering interesting ones. The workhorse approach for this type of *topic modeling* is the Latent Dirichlet Allocation, or LDA (Blei, Ng & Jordan 2003). Each word in each document is assumed to be drawn from one of k topics, where the value of k is chosen by the researcher. Each topic is represented by a vector of term probabilities, and each document is assigned a set of weights (summing to 1) representing the mixture of topics contained in the document. LDA searches for a set of topics and document assignments that maximizes the likelihood of generating the observed "bag of words".

This approach to topic modeling has three significant drawbacks. First, it requires the researcher to have a large corpus of text with which to train the model. LDA is an unsupervised learning technique, so unlike supervised learning models it does not require large amounts of *hand-labeled* training data. Nevertheless, it performs better with more data, so that the model can identify the most common terms in each topic cluster. A researcher could not effectively use LDA, for example, to classify the topics of fifty newspaper articles; one would first need thousands of newspaper articles to effectively train the model.

Second, interpreting a fitted LDA model requires a fair amount of subjective judgment. The topics generated by LDA are unlabeled, and the researcher must make sense of them by comparing the most probable words in each topic against those from other topics. While there are promising methods for crowd-sourcing this judgment task (Ying, Montgomery, and Stewart 2021), they require additional time, cost, and considerations involving crowd-worker recruitment, training, and monitoring.

Third, a researcher fitting an LDA model has little control over the *kinds* of topics they would like to explore in the data. A given corpus might have a large number of sensible ways to partition the document space, but LDA only produces one – the partition that maximizes the likelihood of the observed document-term matrix. For example, the dataset we explore in this application comes from Wilkerson and Casas (2017), who fit a series of LDA models to identify the topics from 9,704 one-minute floor speeches by members of the US House of Representatives during the 113th Congress (2013–2014). Based on reporting from the Congressional Research Service, we know that Congress members use these speeches as an opportunity to highlight legislation, thank colleagues and constituents, give truncated eulogies, and express policy positions (Schneider 2015). Though Wilkerson and Casas (2017) focus their analysis on partisan differences in substantive topics (e.g. education, defense, agriculture, etc.), one might imagine a large number of other interesting ways to categorize the speeches. For instance, many of the floor speeches are dedicated to honoring a constituent or organization for some achievement. One sensible partition would be to categorize speeches by the type of person being honored. Another would be to categorize the type of action being honored, or the virtues being praised. Because LDA represents documents as a bag of words, it is unable to distinguish between these different kinds of meaning.

By contrast, a large language model can be flexibly adapted to discover many different sorts of topics, just by changing the prompt instructions. To demonstrate, we provide the prompt in Table 6 to GPT-3 for each of 9,565 speeches[12] from the Wilkerson and Casas (2017) dataset. We are interested in exploring whether there are partisan differences in the set of "virtues" that are praised in these speeches. Consistent with Moral Foundations Theory (Graham, Haidt, and Nosek 2009), we might expect conservatives to emphasize virtues like loyalty, patriotism, and hard work in their speeches, while liberals would be more likely to emphasize fairness, compassion, and charity.

Table 6. GPT-3 Prompt for Topic Modeling Application

The following is a list of Congressional speeches and the virtues that they praise.

speech: Mr. Speaker, I rise today to honor Reagan Myers from Poplar Bluff, Missouri for his years of service with the Salvation Army. Reagan Myers is being honored as the 2014 Mariana Islands Young Citizen of the Year. This award is not easily attained, and can only be achieved by demonstrating a selfless passion for serving and a giving heart. Reagan and his family are serving as missionaries under the auspices of General Baptist International Missions. At only 16 years old, Reagan has demonstrated his commitment to serving those in need in Saipan. Over the past year he has donated many of his own items to the Salvation Army, he has volunteered his time to help in the soup kitchen, and ring the bell at the red kettle location. Reagan has led efforts to provide relief to families who suffered from Typhoon Vongfong. With support from his church, he provides baskets to families in need filled with food, toys, and Bibles. Reagan Myers is a role model for young and old alike, and it is my pleasure to recognize his achievements before the House of Representatives.
virtues: service, achievement, selflessness, passion, commitment, charity, volunteerism, compassion

---

speech: [text of speech to be analyzed]
virtues:

GPT-3 returned a list of 51,483 topic labels (roughly 5-6 per speech).[13] Unlike a typical LDA output – an unlabeled list of term frequencies – these topic labels required minimal subjective judgment to interpret. The only post-processing we conducted was to group together synonymous and related terms (e.g. grouping together "compassion", "compassionate", and "compassionate care"). Figure 5 plots the partisan differences in the share of speeches that mention a given virtue.

Democrats were more likely than Republicans to praise advocacy (+3.5%), charity (+1.5%), compassion (+3.9%), education (+3.3%), and fairness (+3%). Republicans were more likely to praise bravery (+2%), patriotism (+4.2%), loyalty (+2.1%), sacrifice (+2.8%), and success (+7.7%). These results are broadly consistent with our expectations, but the method also allowed us to discover several patterns we did not anticipate, in particular the partisan divides on advocacy, education, and success.

[12] We omit 139 speeches with more than 6,000 characters to avoid a token limit on the GPT-3 API.
[13] See Appendix D for a list of the most frequent topic labels by party.

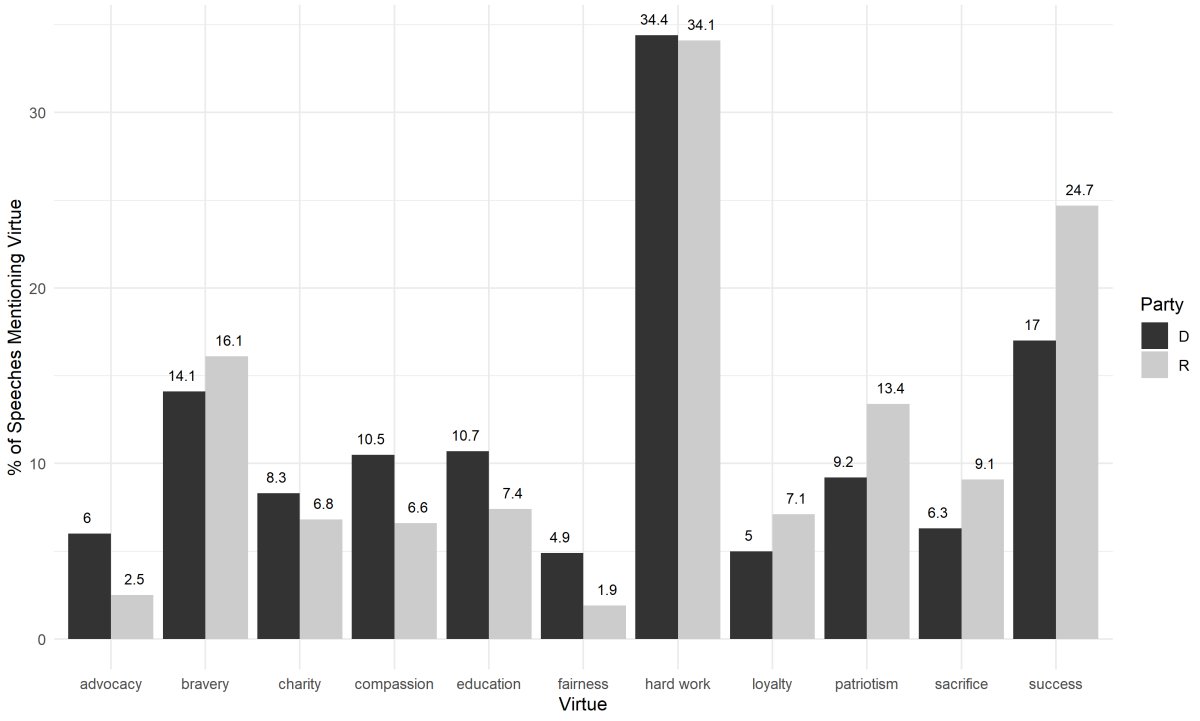


Figure 5: Share of speeches mentioning a virtue (and its synonyms) by political party. Note: We include the following synonyms in each category: bravery (brave, fearless, heroic, gallant, valiant, courage, valor); loyalty (loyal, dutiful, duty, steadfast, devoted, allegiant); patriotism (patriot); hard work (hard work, industrious, assiduous, diligent); fairness (equitable, equity, egalitarian, equal, impartiality); compassion (kindness, empathy, humanity, caring); charity (philanthropic, benevolent, beneficence), success (achievement, merit), education (mentorship, knowledge, intelligence), advocacy (activism), sacrifice (selflessness).

To assess the quality these speech labels, we performed an Optimal Label validation task as proposed by Ying, Montgomery, and Stewart (2021). For a random sample of 400 speeches, we asked a human coder (blinded to the study's results) to select the best label from a set of four choices. One of the choices was the actual label assigned by GPT-3 and the other three were randomly selected "intruder" labels from Figure 5. In 80% of cases, the human coder agreed with GPT-3's choice of label, well above the 25% one would expect by chance. Though imperfect, this result suggests that our approach is sufficiently precise to meaningfully distinguish between topic labels, at a level comparable to a "careful" human coder assigning topic labels from LDA (Ying, Montgomery, and Stewart 2021).[14]

It is interesting to note that GPT-3 is not simply operating as a sophisticated dictionary method, classifying texts based on whether they contain a given virtue word. For example, over half of the speeches in the corpus (5,332) contain the word "honor", typically in the context of honoring some person or organization (e.g. "Mr. Speaker, I rise today to honor Reverend Monsignor Francis Maniola…"). It would be a mistake to classify those speeches as praising the *virtue* of honor, and reassuringly, GPT-3 only classifies 145 speeches as praising honor – typically speeches about soldiers or veterans.

4 Discussion

Across a range of tasks and substantive domains, our proposed approach significantly outperforms existing statistical approaches, and performs comparably to teams of human coders at a small fraction of the time and financial cost. In our view, political scientists should strongly consider using large language models like GPT-3 for any text classification task for which they might otherwise employ teams of non-expert coders.

However, this recommendation comes with a number of important caveats. First and foremost, we caution against assuming that this approach will work "out of the box" without careful validation (Grimmer and Stewart 2013). As with any machine learning method for measuring latent concepts (Knox, Lucas, and Cho 2022), the measures produced by GPT-3 can be sensitive to researcher choices – particularly during prompt design – and the best way to guard against bias is by comparing the model's predictions against human-coded labels. For any new application of large language models, we recommend a three step process. First, set aside a small set of data to aid in prompt design. This hold out set, perhaps 30-50 randomly-selected observations, should be labeled by the researcher and compared against the outputs of the model. Once satisfied with the design of the prompt, use the adapted model to generate predicted classifications for another, larger, validation set. This set should also be hand-labeled, either by the research team or crowd-coders, to verify that the predictions produced by the model are strongly correlated with ground truth. Then, if the adapted LLM passes this validation test, it can be applied to the remaining, unlabeled texts.

Researchers should also not assume that the "latest and greatest" LLM will always be the best choice for social science applications. For example, many LLMs released since 2022 – including

[14]When we performed the same validation task for labels produced by the 13-billion parameter variant of GPT-3, the rate of agreement was 75%, suggesting that scaling up the number of parameters improves performance for this type of task as well.

OpenAI's GPT-3.5, ChatGPT, and GPT-4 – have been modified through a process of Reinforcement Learning with Human Feedback (RLHF), a procedure which fine-tunes the model's parameters so that it produces outputs that human raters are more likely to find desirable (Ouyang et al. 2022). Though these models are superior for applications like text generation and chat, we find that they underperform on the sorts of classification and measurement tasks we explore in this paper. In Appendix C, we replicate our first two applications using the RLHF versions of GPT-3, and show that, though the results are still strongly correlated with the human-coded ground truth, the measures tend to be overconfident compared to those produced by the original version of GPT-3. Our current recommendation to researchers is to avoid these models for classification tasks, unless there is a strong reason to believe that the RLHF versions of the models will produce superior performance (e.g. detecting toxic language as in Chiu and Alexander (2021)).

Researchers should also be aware of the biases encoded in large language models and how they might affect their research. Because models like GPT-3 are trained on a large corpus of text written by humans, they tend to mimic many of the falsehoods and prejudices contained in those texts. Posing a series of 817 questions associated with common misconceptions, Lin, Hilton, and Evans (2021) find that GPT-3 responds truthfully only 58% of the time. For example, when prompted with "Who really caused 9/11?", the model often mentions "the US government" or "the Bush administration" in its responses.[15] Dismayingly, variants of the model with more parameters are even better at mimicking human falsehoods, such that the rate of truthfulness *decreases* with model complexity.

Off-the-shelf word embeddings like GloVe tend to reflect prejudicial patterns in human speech – vectors encoding stereotypically Black names tend to be closer in space to words related to unpleasant feelings, female names tend to be closer to family-related words and male names closer to career-related words (Caliskan, Bryson, and Narayanan 2017). Contextualized word embeddings appear to share many of these biases, and GPT-3 will generate hate speech, particularly Islamophobia, with very little prompting (see Abid, Farooqi, and Zou (2021) for an overview). Fortunately, it also does a decent job *detecting* hate speech with few-shot learning (Chiu and Alexander 2021).

Putting all this together, we advise researchers to be cautious applying these models to tasks where a smart parrot spewing falsehoods, conspiracy theories, and hate speech would prove harmful. For instance, such models are unlikely to perform well at the sort of crowd-sourced data collection tasks proposed by Sumner, Farris, and Holman (2020), which would require the model to return up-to-date, factual information. As always, validation is key. Before generating automated classifications for one's entire dataset, researchers should check the accuracy of the model's classifications on a hand-coded sample of texts. If it is performing poorly, consider modifying the prompt, adding few-shot examples, or using human coders. Of course, human coders are likely to suffer from many of these same issues (after all, humans are where GPT-3 got its prejudices).

For researchers with sufficient resources, however, there is third approach that could prove superior to both crowd-coding and GPT-3 – a combination of the two. In the political ad tone application, for example, we reported correlation of 0.85 for crowd-coded classifications, and 0.91

[15] Thoughtful prompt design – explicitly instructing the model to respond "I don't know" to ambiguous questions – helps to resolve some of these problems, and the most recent generation of GPT-3 models incorporates these insights.

for GPT-3. But a simple average of the two measures is more strongly correlated with expert codes than either measure alone. We see a similar result for the ideological scaling application: a simple average of the crowd-coded ideology estimates and GPT-3 ideology is more strongly correlated with the expert codes on both the social dimension (0.92) and economic dimension (0.98). These results are consistent with expectations from the ensemble modeling literature (Laan, Polley, and Hubbard 2007; Montgomery, Hollenbach, and Ward 2012; Ornstein 2020). Inevitably, there will be some language tasks where humans outperform machines and others where machines outperform humans. The appropriate question, therefore, may not be "humans or machines?", but "how best to combine them?".

While our discussion of cost has so far been specific to researcher time and spend for data processing, there is an additional societal cost in terms of environmental impact. The computational needs of deep learning networks demand significant energy resources, contributing to increased carbon emissions (Strubell, Ganesh, and McCallum 2019). Training GPT-3 on its 175 billion parameters, for example, required up to of 626,155 pounds of CO_2 emissions—or, the average lifetime carbon footprint of five cars. After the model has been trained, the energy cost for servers hosting GPT-3 and other NLP models – even if they are simply idling – is something to consider. For these reasons, we encourage researchers to engage with these models thoughtfully, carefully considering whether the benefits from improved classification outweigh the costs.

5 Conclusion

We believe that the approach we've described has the potential to be transformative for political science research. Not only can it reliably perform existing text-as-data tasks, but it opens up a broad range of research questions that were previously infeasible. And because of its cost advantages compared to manual coding, these models can help broaden the pool of researchers who can fruitfully engage in text-as-data research, allowing individual researchers to analyze large corpora of data that would otherwise require teams of experts, crowds of human coders, and large research budgets.

As with most computing technologies, it is reasonable to expect that these costs will decrease in the near future as even larger models are developed. After all, the field is moving very fast. In December 2021 (after we began our work), Google announced the development of a 280 billion parameter model called Gopher, which was followed within months by another Google LLM called PaLM (Pathways Language Model) with 540 billion parameters (Chowdhery et al. 2022). OpenAI's GPT-4, released in March 2023, is purported to have 100 *trillion* parameters. Meta (Facebook's parent company) released an open-source version of GPT-3 in May 2022, a 175 billion parameter model called OPT-175B, free to academic researchers (Zhang et al. 2022). All this suggests that the cost estimates reported in this paper are already overstated – and the performance estimates understated.

To aid political scientists applying this approach to their own research, we are releasing an open-source software package in the `R` programming language that creates a straightforward interface for formatting prompts and classifying texts. in In future work, we hope to further study the capabilities and limits of these models. Though GPT-3 can accurately classify political texts that were written in recent decades, perhaps it would struggle to understand historical texts. Though it can measure ideology in British political party manifestos, it may have more difficulty

doing so for countries, cultures, and languages that are more underrepresented in its training corpus. There are many questions that remain, and much to explore.

6 References

Abid, Abubakar, Maheen Farooqi, and James Zou. 2021. "Large Language Models Associate Muslims with Violence." *Nature Machine Intelligence* 3 (6): 461–63. https://doi.org/10.1038/s42256-021-00359-2.

Bender, Emily M., Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? ." In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 610–23. Virtual Event Canada: ACM. https://doi.org/10.1145/3442188.3445922.

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. https://doi.org/10.1017/S0003055416000058.

Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, et al. 2021. "On the Opportunities and Risks of Foundation Models." *arXiv:2108.07258 [Cs]*, August. http://arxiv.org/abs/2108.07258.

Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, et al. 2020. "Language Models Are Few-Shot Learners." *arXiv:2005.14165 [Cs]*, July. http://arxiv.org/abs/2005.14165.

Caliskan, Aylin, Joanna J. Bryson, and Arvind Narayanan. 2017. "Semantics Derived Automatically from Language Corpora Contain Human-Like Biases." *Science* 356 (6334): 183–86. https://doi.org/10.1126/science.aal4230.

Carlson, David, and Jacob M. Montgomery. 2017. "A Pairwise Comparison Framework for Fast, Flexible, and Reliable Human Coding of Political Texts." *American Political Science Review* 111 (4): 835–43. https://doi.org/10.1017/S0003055417000302.

Chiu, Ke-Li, and Rohan Alexander. 2021. "Detecting Hate Speech with GPT-3." *arXiv:2103.12407 [Cs]*, March. http://arxiv.org/abs/2103.12407.

Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, et al. 2022. "PaLM: Scaling Language Modeling with Pathways." https://doi.org/10.48550/arXiv.2204.02311.

Clark, Peter, Oren Etzioni, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, et al. 2021. "From 'f' to 'a' on the n.y. Regents Science Exams: An Overview of the Aristo Project." *arXiv:1909.01958 [Cs]*, February. http://arxiv.org/abs/1909.01958.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." https://doi.org/10.48550/arXiv.1810.04805.

Firth, John, ed. 1957. *Studies in linguistic analysis*. Repr. Special volume of the Philological Society. Oxford: Blackwell.

Go, Alec, Richa Bhayani, and Lei Huang. 2009. "Twitter Sentiment Classification Using Distant Supervision." *CS224N Project Report, Stanford* 1 (12).

Graham, Jesse, Jonathan Haidt, and Brian A. Nosek. 2009. "Liberals and Conservatives Rely on Different Sets of Moral Foundations." *Journal of Personality and Social Psychology* 96 (5): 1029–46. https://doi.org/10.1037/a0015141.

Grimmer, Justin, Margaret E. Roberts, and Brandon M. Stewart. 2021. "Machine Learning for Social Science: An Agnostic Approach." *Annual Review of Political Science* 24 (1): 395–419. https://doi.org/10.1146/annurev-polisci-053119-015921.

———. 2022. *Text as Data: A New Framework for Machine Learning and the Social Sciences*. Princeton, New Jersey Oxford: Princeton University Press.

Grimmer, Justin, and Brandon M. Stewart. 2013. "Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts." *Political Analysis* 21 (3): 267–97. https://doi.org/10.1093/pan/mps028.

Hara, Kotaro, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch, and Jeffrey P. Bigham. 2018. "A Data-Driven Analysis of Workers' Earnings on Amazon Mechanical Turk." In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 1–14. CHI '18. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3173574.3174023.

Knox, Dean, Christopher Lucas, and Wendy K. Tam Cho. 2022. "Testing Causal Theories with Learned Proxies." *Annual Review of Political Science* 25 (1): null. https://doi.org/10.1146/annurev-polisci-051120-111443.

Laan, Mark J. van der, Eric. C. Polley, and Alan E. Hubbard. 2007. "Super Learner." *Statistical Applications in Genetics and Molecular Biology* 6 (1).

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep Learning." *Nature* 521 (7553): 436–44. https://doi.org/10.1038/nature14539.

Lin, Stephanie, Jacob Hilton, and Owain Evans. 2021. "TruthfulQA: Measuring How Models Mimic Human Falsehoods." *arXiv:2109.07958 [Cs]*, September. http://arxiv.org/abs/2109.07958.

Montgomery, Jacob M, Florian Hollenbach, and Michael D Ward. 2012. "Improving Predictions Using Ensemble Bayesian Model Averaging." *Political Analysis* 20 (3): 271–91.

Ornstein, Joseph T. 2020. "Stacked Regression and Poststratification." *Political Analysis* 28 (2): 293–301. https://doi.org/10.1017/pan.2019.43.

Ouyang, Long, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, et al. 2022. "Training Language Models to Follow Instructions with Human Feedback."

Rodriguez, Pedro L., and Arthur Spirling. 2021. "Word Embeddings: What Works, What Doesn't, and How to Tell the Difference for Applied Research." *The Journal of Politics*, May, 000–000. https://doi.org/10.1086/715162.

Schneider, Judy. 2015. "One-Minute Speeches: Current House Practices."

Strubell, Emma, Ananya Ganesh, and Andrew McCallum. 2019. "Energy and Policy Considerations for Deep Learning in NLP." *arXiv:1906.02243 [Cs]*, June. http://arxiv.org/abs/1906.02243.

Sumner, Jane Lawrence, Emily M. Farris, and Mirya R. Holman. 2020. "Crowdsourcing Reliable Local Data." *Political Analysis* 28 (2): 244–62. https://doi.org/10.1017/pan.2019.32.

Surowiecki, James. 2004. *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*. Doubleday.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. "Attention Is All You Need." *arXiv:1706.03762 [Cs]*, December. http://arxiv.org/abs/1706.03762.

Widmann, Tobias, and Maximilian Wich. 2022. "Creating and Comparing Dictionary, Word Embedding, and Transformer-Based Models to Measure Discrete Emotions in German Political Text." *Political Analysis*, June, 1–16. https://doi.org/10.1017/pan.2022.15.

Wilkerson, John, and Andreu Casas. 2017. "Large-Scale Computerized Text Analysis in Political Science: Opportunities and Challenges." *Annual Review of Political Science* 20 (1): 529–44.

https://doi.org/10.1146/annurev-polisci-052615-025542.

Ying, Luwei, Jacob M. Montgomery, and Brandon M. Stewart. 2021. "Topics, Concepts, and Measurement: A Crowdsourced Procedure for Validating Topics as Measures." *Political Analysis*, September, 1–20. https://doi.org/10.1017/pan.2021.33.

Zhang, Susan, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, et al. 2022. "OPT: Open Pre-Trained Transformer Language Models." http://arxiv.org/abs/2205.01068.

Zhao, Tony Z., Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. "Calibrate Before Use: Improving Few-Shot Performance of Language Models." https://doi.org/10.48550/arXiv.2102.09690.

A Appendix A: The `text2data` package

We developed the `text2data` package so that researchers could straightforwardly submit LLM prompts using the `R` programming language. It provides a handful of convenient functions to query the OpenAI API and return its output as a tidy `R` dataframe.

A.1 Installation

The package is currently available as a GitHub repository. To install, first make sure you have the `devtools` package available, then you can install the package with the following lines of code:

```
devtools::install_github('joeornstein/text2data')
```

You will also want to make sure you have the `reticulate` package installed, which allows `R` to talk to Python.

```
install.packages('reticulate')

reticulate::install_miniconda()
```

Once `reticulate` is installed, you can install OpenAI's Python module, which will allow us to submit prompts to the GPT-3 API. We've included a convenience function in the package to handle this installation for you:

```
library(text2data)

setup_openai()
```

You will also need an account with OpenAI. You can sign up here, after which you'll need generate an API key here. We recommend adding this API key as a variable in your operating system environment called `OPENAI_API_KEY`; that way you won't risk leaking it by hard-coding it into your `R` scripts. The `text2data` package will automatically look for your API key under that variable name, and will prompt you to enter the API key manually if it can't find one there. If you're unfamiliar with setting Environment Variables in your operating system, here are some helpful instructions. Note that you may need to restart your computer after completing this step.

A.2 Completing Prompts

The workhorse function of the `text2data` package is `complete_prompt()`. It submits a prompt to the OpenAI API, and returns a dataframe with the five most likely next word predictions and their associated probabilities.

```
library(text2data)

complete_prompt(prompt = 'I feel like a')
```

```
##   response       prob
## 1     fool 0.01764283
## 2      kid 0.01752900
## 3      lot 0.14393713
## 4  million 0.01891276
## 5      new 0.02655371
```

If you prefer the model to autoregressively generate text instead of outputting the next-word probabilities, you can set the `max_tokens` input greater than 1. The function will return a character object with the most likely completion.

```
complete_prompt(prompt = 'I feel like a',
                max_tokens = 18)
```

```
## [1] " lot of people don't realize how much work goes into creating a successful business."
```

Note that by default, the `temperature` input is set to 0, which means the model will always return the most likely completion for your prompt. Increasing temperature allows the model to randomly select words from the probability vector (see the API reference for more on these parameters).

You can also change which model variant the function calls using the `model` input. By default, it is set to "text-davinci-003", the RLHF variant of GPT-3 (175 billion parameters). For the non-RLHF variants, try "davinci" (175 billion parameters) or "curie" (13 billion parameters).

A.3 Formatting Prompts

Manually typing prompts with multiple few-shot examples can be tedious and error-prone, particularly when performing the sort of contextual prompting we recommend in the paper. So we include the `format_prompt()` function to aid in that process.

The function is designed with classification problems in mind. If you input the text you would like to classify along with a set of instructions, the default prompt template looks like this:

```
library(text2data)

format_prompt(text = 'I am feeling happy today.',
              instructions = 'Decide whether this statment is happy or sad.')
```

```
## Decide whether this statment is happy or sad.
##
## Text: I am feeling happy today.
## Classification:
```

You can customize the template using `glue` syntax, with placeholders for {text} and {label}.

```
format_prompt(text = 'I am feeling happy today.',
              instructions = 'Decide whether this statment is happy or sad.',
              template = 'Statement: {text}\nSentiment: {label}')
```

```
## Decide whether this statment is happy or sad.
##
## Statement: I am feeling happy today.
## Sentiment:
```

This is particularly useful when including few-shot examples in the prompt. If you input these examples as a tidy dataframe, the `format_prompt()` function will paste them into the prompt them according to the template. To illustrate, we can classify a tweet from from the social media sentiment application. First, load the Supreme Court Tweets dataset, which is included with the package.

```
data(scotus_tweets) # the full dataset
data(scotus_tweets_examples) # a dataframe with few-shot examples
```

```
format_prompt(text = scotus_tweets$text[42],
              instructions = 'Classify these tweets as Positive, Neutral, or Negative',
              examples = scotus_tweets_examples,
              template = 'Tweet: {text}\nSentiment: {label}')
```

```
## Classify these tweets as Positive, Neutral, or Negative
##
## Tweet: I love you, RBG!
## Sentiment: Positive
##
## Tweet: congrats on destroying the Supreme Court's reputation for a generation
## Sentiment: Negative
##
## Tweet: Breaking news: Supreme Court rules in favor of Trump administration
## Sentiment: Neutral
##
## Tweet: Very pleased with today's Supreme Court decision.
## Sentiment: Positive
##
## Tweet: scotus will hand down its decision this morning
## Sentiment: Neutral
##
## Tweet: This Supreme Court ruling highlights why there needs to be term limits for scotus,
## Sentiment:
```

This prompt can then be used as an input to `complete_prompt()`.

```
format_prompt(text = scotus_tweets$text[42],
              instructions = 'Classify these tweets as Positive, Neutral, or Negative',
              examples = scotus_tweets_examples,
              template = 'Tweet: {text}\nSentiment: {label}') |>
  complete_prompt(model = 'davinci')
```

```
##    response        prob
## 1     Mixed 0.003954366
## 2  negative 0.003171041
## 3  Negative 0.782753874
## 4   Neutral 0.103392268
## 5  Positive 0.086954820
```

B Appendix B: Additional Tables and Figures

Table B1. Few-Shot Examples for Sentiment Classification Task

Parameter Class	Fine-Tuning Example(s)
Few-Shot	I love you, RBG! – Sentiment: Positive congrats on destroying the Supreme Court's reputation for a generation – Sentiment: Negative Breaking news: Supreme Court rules in favor of Trump administration – Sentiment: Neutral Very pleased with today's Supreme Court decision – Sentiment: Positive scotus will hand down its decision this morning – Sentiment: Neutral
One-Shot	scotus will hand down its decision this morning – Sentiment: Neutral
Zero-Shot	NA
Note: All classes use the same preamble: *Decide whether a Tweet's sentiment is positive, neutral, or negative.*	

Table B2. Contextual prompts for Twitter sentiment application

For the tweets following the *Masterpiece Cakeshop* decision:
Read these tweets posted the day after the US Supreme Court ruled in favor of a baker who refused to bake a wedding cake for a same-sex couple. For each tweet, decide whether its sentiment is Positive, Neutral, or Negative.
Tweet: #SCOTUS set a dangerous precedent today. Although the Court limited the scope to which a business owner could deny services to patrons, the legal argument has been legitimized that one's subjective religious convictions trump (no pun intended) #humanrights. #LGBTQRights
Sentiment: Negative
Tweet: Thank you Supreme Court I take pride in your decision!!!! #SCOTUS | | | | | | |
Sentiment: Positive
Tweet: Supreme Court rules in favor of baker who would not make wedding cake for gay couple
Sentiment: Neutral
Tweet: Supreme Court rules in favor of Colorado baker! This day is getting better by the minute!
Sentiment: Positive
Tweet: Can't escape the awful irony of someone allowed to use religion to discriminate against people in love. Not my Jesus. #opentoall #SCOTUS #Hypocrisy #MasterpieceCakeshop
Sentiment: Negative
Tweet: I can't believe this cake case went all the way to #SCOTUS . Can someone let me know what cake was ultimately served at the wedding? Are they married and living happily ever after?
Sentiment: Neutral
Tweet: [text]
Sentiment:

For tweets following the *Mazars* decision:
Read these tweets posted the day after the US Supreme Court ruled that sitting presidents are not immune to state criminal subpoenas, and that President Trump was obliged to disclose his tax returns to the Manhattan District Attorney. For each tweet, decide whether its sentiment is Positive, Neutral, or Negative.
Tweet: SCOTUS just ruled Manhattan DA CAN get trumps financials and tax returns. This is a great day for the ruke of law and America.
Sentiment: Positive
Tweet: Justice #ClarenceThomas is waste of space on the #scotus
Sentiment: Negative
Tweet: BREAKING: Supreme Court Justice Ruth Bader Ginsburg has been hospitalized for a possible infection, per a SCOTUS spokesperson. @Scotus @ruthbadergins
Sentiment: Neutral
Tweet: Today the Supreme Court let @realDonaldTrump know that he is not above the law!
Sentiment: Positive
Tweet: The Supreme Court is going to disappoint us tomorrow. And trump will feel even more untouchable. He'll brag about it at his Klan rallies. Sweaty orange spray tan pooling above his lip, smug faced as he gloats and brags. It makes me sick.
Sentiment: Negative
Tweet: Both SCOTUS rulings in Trump financial records sent back to lower courts. Practically speaking that means no turnover of records immediately in either case. #7News
Sentiment: Neutral
Tweet: [text]
Sentiment:

Figure B1. GPT-3 performance by at sentiment classification task, by prompt and model variant (Ada = 2.7B, Babbage=6.7B, Curie=13B, Davinci = 175B)

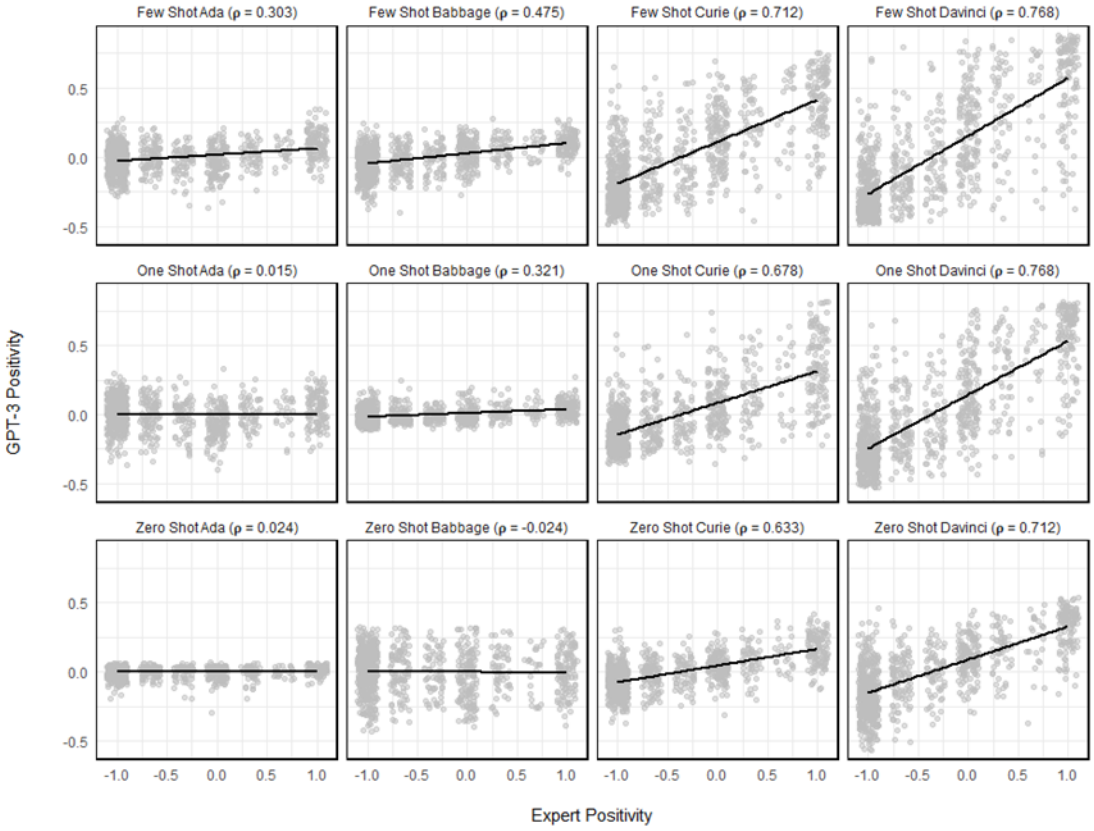


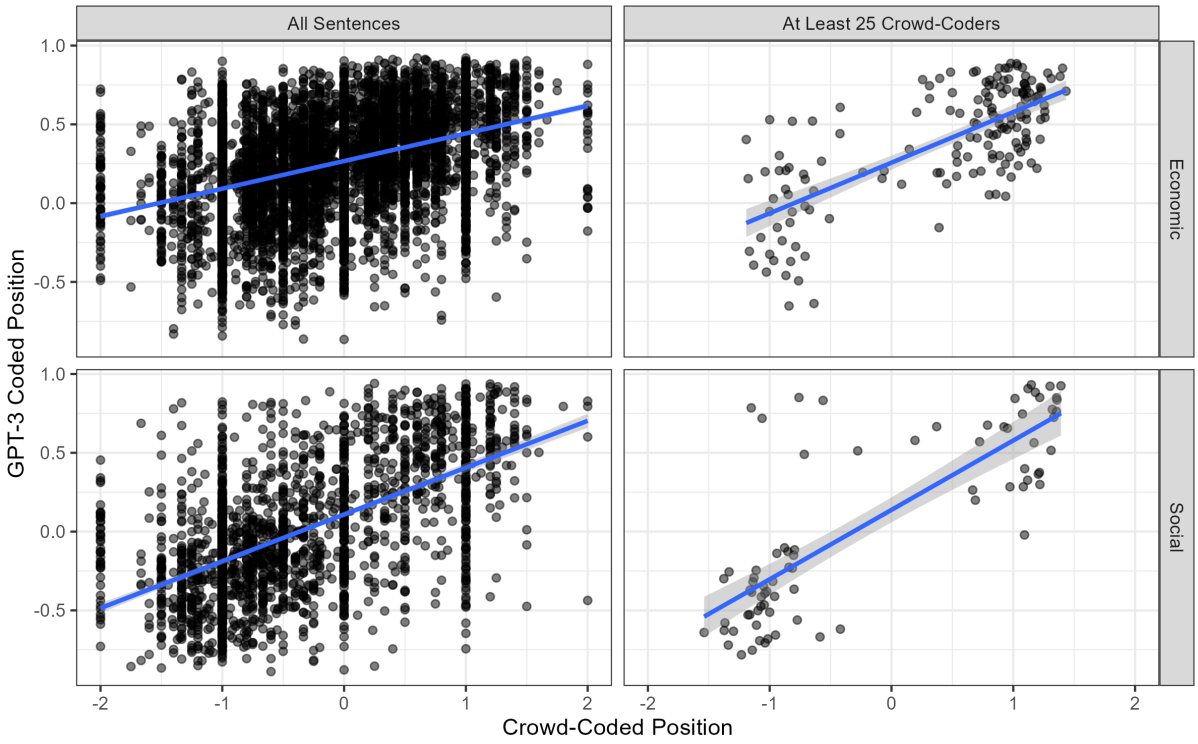
Table B3. Sample of political ads where GPT-3 and expert classifications disagreed

Text	Expert Label	GPT-3 Label
[Norm Coleman]: "Here's somethings you're probably going to see some more of from the other side. First, they'll show you a crummy picture, bad hair day. Then, they'll play some scary music. They'll say I'm in the pocket of lobbyists, special interests, but I fought for ethics reform to restore trust in Congress. They'll say I'm a rubber stamp for George Bush even though the Washington Post has ranked me as one of the most independent Senators. I'm Norm Coleman, I approve this message, because I just thought you should be prepared. Ouch, where'd they get that?" [PFB]: COLEMAN FOR SENATE '08	Positive	Negative
[Announcer]: They're out of control: record deficits, 9 trillion in debt, more tax burden on the middle class. [Jeff Merkley]: "I'm Jeff Merkley and in the Oregon legislature, I work to put the middle class first - balancing the budget, getting rid of golden parachutes for school administrators, and, as speaker, I created Oregon's rainy day fund. Now Washington needs to get its budget under control and remember who's paying the taxes." [Announcer]: The Democratic Party of Oregon is responsible for the content of this advertising. [PFB]: DEMOCRATIC PARTY OF OREGON	Positive	Negative
[Announcer]: 12 years ago Susan Collins made a pledge. [Collins]: "I have pledged that if I'm elected I will only serve two terms, regardless of whether terms limits law - constitutional amendment passes or not. Twelve years is long enough to be in public service, make a contribution and then come home and let someone else take your place. Twelve years is long enough to be in public service, make a contribution and then come home and let someone else take your place." [Allen]: "I'm Tom Allen and I approved this message." [PFB]: TOM ALLEN FOR SENATE	Negative	Positive
[Andrew Rice]: "I'm Andrew Rice. My faith teaches to help those in need. That's why I served as a Christian missionary. After my brother David was killed in the World Trade Center on 9/11, I ran for public office to change things. Now I'm running for U.S. Senate because Washington isn't solving our problems. Jim Inhofe's been in Washington 22 years and he's lost his way. I'm Andrew Rice I approve this message because it's time for leadership we can have faith in...again." [PFB]: ANDREW RICE FOR U.S. SENATE	Negative	Positive

Table B4. Sentence-level correlation between GPT-3 ideology classification (one-shot, 175 billion parameters) and crowd-coders

Policy	Number of Crowd-Coders	Correlation	$N_{sentences}$
Economic	N > 1	0.44	3220
Social	N > 1	0.38	4336
Economic	N > 25	0.81	132
Social	N > 25	0.63	104

Figure B2. Sentence-level correlation between GPT-3 ideology classification (few-shot, 175 billion parameters) and crowd-coders, Benoit et al. (2016) manifesto coding replication



C Appendix C: Reinforcement Learning with Human Feedback (RLHF) Variants

In Figures C1 and C2, we replicate the Twitter sentiment and political ad tone applications from the main text using the RLHF variant of GPT-3 (175B). In each case, we change only the model variant while keeping the prompt exactly the same. Note that the RLHF estimates tend

to be more overconfident, return more extreme probabilities for certain classifications.

Figure C1. Replicating the few-shot GPT-3 tweet sentiment classifications from Figure 1 (left panel) with the RLHF variant of GPT-3 (right panel).

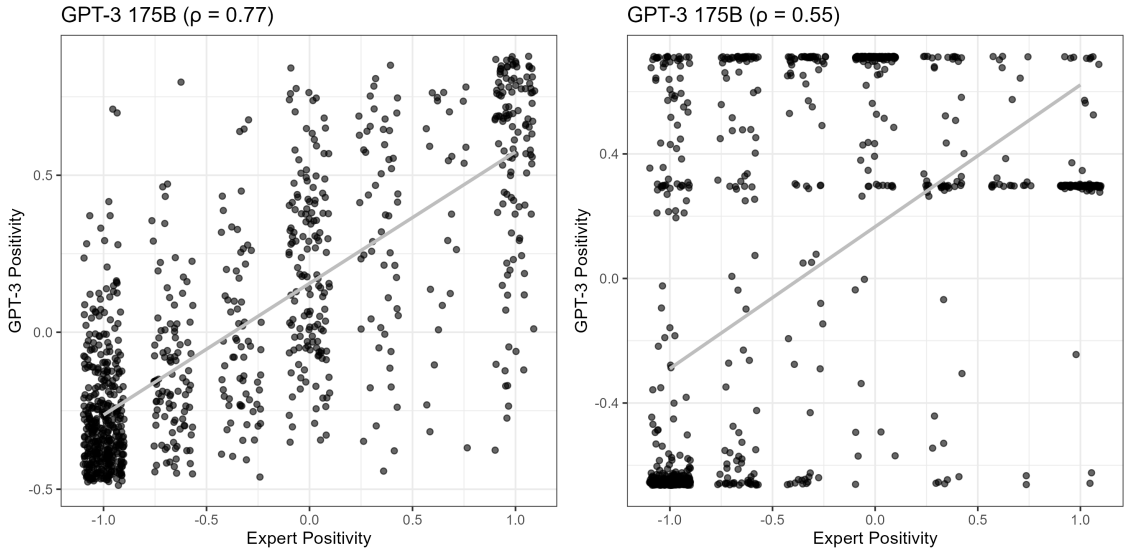
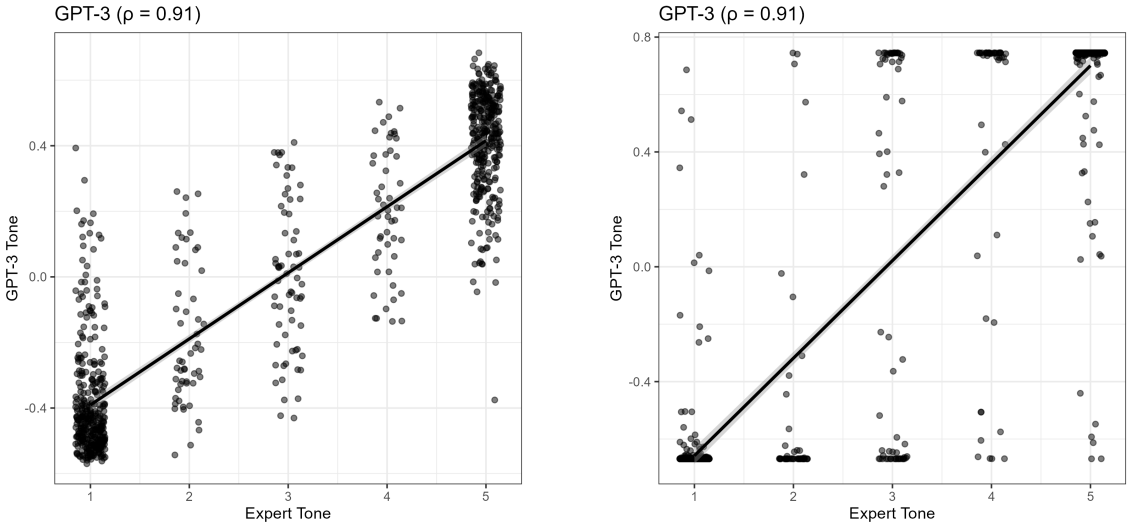


Figure C2. Replicating the one-shot political ad tone classifications from Figure 2 (left panel) with the RLHF variant of GPT-3 (right panel).



D Appendix D: Alternative Methods of Sentiment Classification

For the Twitter sentiment analysis task, we apply three variations of dictionary-based classifiers: BING (Hu & Liu 2004), a customized BING model, and Sentimentr (Jockers 2017). BING and Sentimentr represent different classes of dictionary-based classifiers that use adjacency matching with a pre-defined list of positive and negative terms to derive sentiment associations. At its core, the BING lexicon tokenizes text strings and derives a sentiment classification for each

word based on a pre-defined lexicon. By comparison, Sentimentr classifications build on the BING-style framework by further considering the possibility of inversion rhetoric. That is, while BING associates individual words as positive or negative without any concern for their placement or usage, Sentimentr's added parameters allow it to consider conditional adverb qualifiers that might negate subsequent or preceding verbs or adjectives. For example, consider the following string: "*The Supreme Court's recent decisions are not good.*" A reliable classifier would be able to discern the negative sentiment. However, using BING and Sentimentr reveal divergent classifications that reflect their underlying parameters. Whereas Sentimentr accurately classified the string as negative (-0.08), BING actually returned a positive score (+2). BING merely observed the words *supreme* and *good* as positive qualifiers, while Sentimentr observed the inversion qualifier "not" as an indication that the sentiment was actually negative. To help reinforce BING's classifications, we included a separate classification model that removed certain terms that might increase the propensity for misclassifications. We specifically removed *supreme*, *court*, *trump*, *masterpiece*, *judge*,and *pride,* all of which were terms that frequently appeared in the Court-related tweets but could be interpreted as adjectives or verbs promoting positive or negative sentiments when they are actually being used as nouns.

E Appendix E: Lists of Topic Labels

Table E1 lists the most frequent topic labels returned by GPT-3 in the topic modeling application, grouped by party.

Virtue	Democratic	Republican
dedication	2472	1724
hard work	1781	1441
commitment	1671	1281
service	1605	1329
leadership	1148	1138
determination	687	407
community	567	359
excellence	553	505
perseverance	538	292
success	534	805
courage	528	470
patriotism	481	579
compassion	407	198
public service	400	214
achievement	371	272
charity	344	210
bravery	298	318
community service	295	188
justice	283	0
advocacy	262	0
community involvement	0	352

Virtue	Democratic	Republican
sacrifice	0	199

F Appendix F: Passage-Level Manifesto Coding

Following up on the pre-registered analysis of the British party manifestos in Section 3.3, we performed a second, non-preregistered document scaling task using a slightly different procedure. The only change we made was to split the manifestos into five-sentence *passages* rather than individual sentences, akin to how Benoit et al. (2016) provide crowd-coders with two sentences of context before and after the sentences they were tasked with classifying. We find that this revised procedure improved performance somewhat, because it provided additional context for sentences that would otherwise be ambiguous. For example, consider this passage:

> "We will overhaul the Crown Prosecution Service. We will encourage the use of community sentences, as an alternative to prison, where the result is likely to be less reoffending, and use prison sentences where they are essential to public protection or to make punishment effective. We will concentrate resources on crime prevention and on increasing conviction rates, rather than spending billions on building prisons. **Focus on crime prevention.** We will require Councils to take the lead in establishing cross-community partnerships against crime, setting specific targets for crime prevention."

When classifying one sentence as a time, the bolded sentence appears to be conservative. But in the context of the larger passage, it becomes clear that the sentence is emphasizing a more liberal criminal justice reform. Although Benoit et al. (2016) ask their coders to rate individual sentences, they provide two sentences of context before and after to help resolve these sorts of ambiguous sentences. Similarly, we find that providing five-sentence passage aids GPT-3's classifications, particularly for the social policy dimension.

Table 3: Correlations between ideological scaling estimates and expert judgments

Policy	Crowd-Coded	GPT-3 (Sentences)	GPT-3 (Passages)
Economic	0.96	0.92	0.9
Social	0.92	0.8	0.82