LARGE LANGUAGE MODELS FOR TEXT CLASSIFICATION: FROM ZERO-SHOT LEARNING TO FINE-TUNING

Youngjin Chae[*]
Department of Sociology
Rutgers University
New Brunswick, NJ
yj.chae@rutgers.edu

Thomas Davidson
Department of Sociology
Rutgers University
New Brunswick, NJ
thomas.davidson@rutgers.edu

August 23, 2023

ABSTRACT

This study analyzes large language models (LLMs) as a methodology for computational sociology, focusing on applications to supervised text classification. We consider how the latest generation of text-to-text transformer models can make predictions using prompts and minimal training examples and assess the sensitivity of these approaches to wording and composition. Through a comprehensive case study on identifying opinions expressed about politicians on Twitter and Facebook, we evaluate four different LLM architectures, varying in size, training data, and architecture. We compare the performance across different training regimes, from prompt-based zero-shot learning to fine-tuning using thousands of annotated examples. Our findings demonstrate how LLMs can perform complex text classification tasks with high accuracy, substantially outperforming conventional baselines. We use these results to provide practical recommendations for sociologists interested in employing LLMs for text classification tasks. Fine-tuning smaller models offers an optimal solution for most researchers due to their relatively high accuracy and low cost. We discuss the trade-offs between proprietary and open-source models, the importance of evaluating models for bias, and concerns related to transparency and reproducibility. This study contributes to understanding the capabilities and limitations of these models in a sociological context, providing a foundation for future research and applications in the field.

Keywords large language models · stance detection · computational sociology

Acknowledgments. We thank members of the American Sociological Association Collective Behavior and Social Movements Section Small-Group on Computational Methods and attendees at the session on Text as Data at the 2023 Annual Meeting in Philadelphia for helpful comments and suggestions. We also thank the Office of Advanced Research Computing (OARC) at Rutgers, The State University of New Jersey for providing access to the Amarel cluster that contributed to the results reported here.

[*]This pre-print is currently under review. Please contact the authors via email with any feedback or questions.

1 Introduction

Large language models (LLMs) have recently led to substantial breakthroughs in many areas of natural language processing, and attention to the topic of artificial intelligence among academics and the general public has exploded with the release of OpenAI's ChatGPT in late 2022. LLMs promise to open up new opportunities for social scientific research, as state-of-the-art models can easily be adapted to new tasks and show impressive performance with relatively limited training (Do et al., 2022, Wankmüller, 2022). The generative capacity of these models also enables new kinds of social scientific inquiry based on machine-generated texts (Argyle et al., 2023). We anticipate that there will be growing interest in the study of the impact of these technologies as they become ubiquitous across many domains of social life (Joyce et al., 2021). In this paper, we evaluate the use of LLMs as a methodology for supervised text classification. We highlight the capabilities of these models for a sociological audience, providing an overview of recent developments in computational linguistics and machine learning and demonstrating how various methodological decisions involved in the implementation of LLMs impact predictive performance.

As a case study, we consider the application of LLM classifiers to the task of detecting stances in social media posts (Somasundaran and Wiebe, 2010, Aldayel and Magdy, 2021). One of the key objectives of sociological research is to infer attitudes, opinions, and beliefs, whether via surveys, interviews, or ethnographic study. Stance detection represents an important computational approach that can achieve this goal using observational texts that sociologists are increasingly collecting and analyzing, and it is more suitable for such measurement than the more popular technique of sentiment analysis (Bestvater and Monroe, 2022). In particular, we consider stances expressed toward leading candidates in the 2016 US Presidential election. We use two datasets containing social media posts annotated with stances expressed toward Donald Trump and Hillary Clinton in the 2016 election, a widely used benchmark dataset from Twitter (Mohammad et al., 2016), and an original set of Facebook comments. While our case study is directly relevant to political sociology and analyses of online speech, the same techniques can be applied to any text classification task.

To assess the performance of large language models for text classification, we compare four different models, varying in terms of size, architecture, and the manner in which we interact with them. We selected two models based on the popular BERT framework (Devlin et al., 2019, Lewis et al., 2019, Yin et al., 2019) and two variants of OpenAI's GPT-3 (Brown et al., 2020, Ouyang et al., 2022). Across these models, we vary the complexity of the prediction task and explore how different

training regimes impact performance, from zero-shot approaches that rely purely on instructions contained in prompts without any training data to models fine-tuned on thousands of annotated examples. Stance detection is also conducive to experimentation because we can vary the complexity of the task by manipulating whether it involves simple stance prediction, joint prediction of both a stance and the target, or identification of stances expressed towards multiple targets. Additionally, we consider how textual prompts and the composition of training examples affect performance in zero- and few-shot settings.

We develop recommendations for sociologists and other social scientists interested in using LLMs for text classification based on the results of these experiments. We find that LLMs perform just as well when provided with complex prediction tasks as simpler alternatives, highlighting how these models can generalize to multifaceted kinds of inferences that are often of interest to sociologists. Most models evaluated substantially outperform baseline models trained using conventional machine learning approaches. While the largest, most powerful model considered achieves the best performance in nearly all tests, smaller LLMs fine-tuned on annotated data perform almost as accurately as their larger siblings at a fraction of the cost, making them the optimal solution for most research projects. We also discuss how document length, corpus size, and resource constraints impact model selection. Finally, we consider potential biases in these models and how they can be identified and addressed and examine the trade-offs faced when deciding between proprietary models and open-source alternatives. We anticipate that these experiments and recommendations will serve as a useful resource for sociologists interested in deploying these cutting-edge tools in their research.

2 Background

2.1 Text classification and large language models

Text classification is an application of supervised machine learning (SML) used to categorize texts into pre-determined classes, in contrast to unsupervised approaches such as topic modeling, which are used to inductively summarize and group texts (Evans and Aceves, 2016, Nelson, 2017, Molina and Garip, 2019). Text classification is widely used in computer science for tasks such as sentiment analysis (Pang and Lee, 2008), spam filtering (Méndez et al., 2006), and hate speech detection (Davidson et al., 2017). Over the past decade, the technique has been adopted by sociologists for a variety of tasks, including the analysis of protest discussion on Twitter (Hanna, 2013), coverage of

inequality in newspaper articles (Nelson et al., 2018), and rhetoric in political speeches (Bonikowski et al., 2022).

Large language models are the most recent innovation in supervised text classification. Several recent studies show how these models outperform conventional machine learning algorithms at various sociologically relevant classification tasks such as the detection of emotional language (Widmann and Wich, 2022), nationalist, populist, and authoritarian rhetoric (Bonikowski et al., 2022), and offensive language (Wankmüller, 2022). To understand these new techniques, it is important to contextualize them by discussing earlier developments in language modeling. Interest in using computers to model and generate texts dates back to the beginning of the 20th century (Martin and Jurafsky, 2009, 122-123), but probabilistic language models were first developed in the 1990s. The goal of a language model is to predict the next word or set of words in a sequence. For example, what is the most likely word to end this sentence: "The cat caught a ..."? Each possible word in a vocabulary can be assigned a probability of appearing. A good model should assign a high probability to the word "mouse" and a much lower probability to words like "excavator" or "whale." The simplest approach is the *bi-gram* model, where the previous word is used to predict the next word. Predicted probabilities can be obtained by counting pairs of co-occurring words in large corpora of text (Martin and Jurafsky, 2009). *n-gram* language models extend this logic by using the previous n words in a sequence to predict the next word by modeling text sequences as Markov processes (Brown et al., 1992). Returning to the example above, we should expect to make better predictions if the model has more context (e.g. "cat caught a" compared to "caught a" or "a"). However, these models become intractable as n increases beyond a trivial length and can be extremely sensitive to the composition of the training corpus, making them difficult to generalize to new data. Nonetheless, relatively simple n-gram language models have useful applications for social scientific analyses of texts (Danescu-Niculescu-Mizil et al., 2013, Jensen et al., 2022).

More sophisticated language models based on neural network architectures were proposed in the early 2000s (Bengio et al., 2003), but computational constraints made them difficult to operationalize. By the early 2010s, neural network architectures optimized to run on high-performance graphics processing units (GPUs) and the availability of large corpora of online text made it possible to train more powerful language models. Notably, Mikolov et al. (2013a,b) proposed the Word2Vec architecture, using variations of word prediction tasks to train neural networks as probabilistic language models. The language model itself was largely incidental, receiving less attention than rich semantic associations contained in internal weights, or embeddings, learned during training.

These embeddings can be used to represent texts in low-dimensional space and have opened up new possibilities for the sociological study of language (Kozlowski et al., 2019, Stoltz and Taylor, 2021, Zhou, 2022, Rodriguez and Spirling, 2022).

A type of neural network architecture known as the *transformer* (Vaswani et al., 2017) has enabled the most recent advances in language modeling. A *transformer* consists of an *encoder* and a *decoder*. The encoder takes in an input sequence of text tokens $x = x_1, ..., x_n$ (a token is a word or shorter substring) and converts them into a sequence of continuous values $z = z_1, ..., z_n$, or "hidden states" in the neural network. The process is autoregressive, as previously transformed values $z_1, ..., z_{n-1}$ are used as additional input when the encoding of x_n is generated. The decoder then takes these values and generates an output sequence, $y = (y_1, ..., y_m)$, such as a translation into another language, effectively reversing the operation performed by the encoder. The key innovation proposed by Vaswani et al. (2017) is the *self-attention* mechanism. The model computes a weight for each token in the input sequence based on its relevance to every other token and these weights capture relationships between tokens in the input regardless of the distance between them (see Wankmüller (2022) for further detail on the transformer architecture). Self-attention improves efficiency and enables the transformer-based language models to capture long-range dependencies in texts, effectively solving the tractability issues faced by earlier n-gram language models. This allows these models to capture more nuanced semantic associations, including "contextualized" embedding representations that allow the meaning of a word to vary depending upon its context (Smith, 2020).

Contemporary transformer-based language models consist of neural networks containing multiple stacked transformer layers. These models are known as large language models, or LLMs, due to the scale of the training data and the number of parameters they contain. For example, BERT (Bidirectional Encoder Representations from Transformers) — developed by researchers at Google (Devlin et al., 2019) and one of the most widely used transformer models[1] — was trained on a dataset containing full text from over seven thousand books and the entirety of English Wikipedia, constituting approximately 2.5 billion words. More recent models use even more data, relying on sources like Google's C4 dataset, which contains text from over 15 million web domains, effectively encompassing much of the text available on the Internet.[2] In tandem with the increasing scale of the data, the models have been increasing in size: OpenAI's GPT-3 has 175 billion parameters (Brown et al., 2020) and Google's PaLM model has 540 billion parameters (Chowdhery et al., 2022). However, this trend may be abating due to more efficient training procedures. Google's PaLM-2

is more powerful than its predecessor but has 200 billion fewer parameters [3] and Meta's LLaMA outperforms models ten times larger by training on more text (LLaMA was trained on 1.4 trillion *tokens*, where a token is a word or word fragment) (Touvron et al., 2023).

There are several other important differences between LLMs and conventional approaches to supervised machine learning. First, rather than training a model from scratch, existing LLMs can be adapted to new tasks, known as *transfer learning*. LLMs are initially trained using variations of next-word prediction, similar to earlier language models. After the models have learned a probabilistic representation of a large corpus of text, they can be adapted to perform tasks such as translation, summarization, and question-answering using a process known as *fine-tuning*. The fine-tuning process is integral to the training of many contemporary LLMs and can substantially improve performance across multiple tasks (Radford et al., 2018, Wei et al., 2022). The capacity to perform well across many tasks and generalize easily to new tasks has led some to label LLMs "foundation models" (Bommasani et al., 2022). Most recently, substantial improvements have been achieved through a process known as reinforcement learning with human feedback (RLHF) (Ziegler et al., 2020). Humans rank output returned by a model in response to an input and the scores are fed back into the model. Algorithms are trained to scale this process using *reinforcement learning*, allowing a model to optimize the output automatically. RLHF was used to develop the InstructGPT (Ouyang et al., 2022), the version of GPT-3 used in the analyses below, as well as ChatGPT.

The generalizability of LLMs is particularly desirable for social scientists, as it enables us to use cutting-edge methods and extend them for our purposes. The representational richness and capacity to generalize means that more advanced LLMs can perform reasonably well at new tasks without any additional training, known as *zero-shot learning* (Brown et al., 2020). For example, an LLM trained using texts in multiple languages could perform translation without being explicitly trained to do so. Analysts can provide models with more guidance by providing examples of the input and desired output, known as *few-shot learning*. For example, one could show a sentence and a translation before asking for another sentence to be translated. This additional information should help to improve the quality of the output. Returning to the task of text classification, the capacity to make such predictions with little or no training makes these models particularly promising for empirical research, as they have the potential to perform accurately without requiring time-consuming and costly annotation procedures. To further improve performance, we can also fine-tune an LLM for classification using an entire training corpus to a pre-trained model, which is more analogous to a

standard SML framework. In what follows, we consider how these different training regimes affect predictive accuracy and the trade-offs involved when deciding between them.

Second, the inputs and outputs of the most recent generation of LLMs differ from traditional frameworks. In the standard approach to text classification, texts are converted into numeric vectors, commonly known as "features," which are then input into a model alongside the class labels. Machine learning algorithms are then used to learn weights to map features onto the labels and predict the most likely label (Martin and Jurafsky, 2009, Evans and Aceves, 2016).[4] In contrast, the most recent generation of LLMs are text-to-text transformers, where the input and output both consist of texts (Raffel et al., 2020). When used for classification, the text to be classified is input into the model, and the output is a text-based label.[5] Moreover, LLMs can be given textual *prompts* that provide additional context and instructions alongside the input texts. An emerging field known as *prompt engineering* explores the efficacy of different prompting strategies for text generation and interacting with other generative models like text-to-image transformers (Liu and Chilton, 2022). We consider the implications of text-to-text interfaces for classification tasks and assess how prompts affect predictive performance.

Third, due to the cost of training LLMs — which can run into millions of dollars considering hardware and electricity costs (Bender et al., 2021) — researchers increasingly rely upon models trained by third parties. Some of these models, like BLOOM,[6] are open-source, and companies like OpenAI, Google, and Meta have released weights that allow anyone to use some models, but most of the largest models are only accessible using paid application programming interfaces (APIs). Reliance on third-party models owned and controlled by a handful of corporations raises concerns about transparency, reproducibility, and research ethics (Spirling, 2023). Others have highlighted the drawbacks of large models, including prohibitive training costs, excessive energy consumption, and the way models learn stereotypical associations and biases from training data (Bender et al., 2021). The release of ChatGPT and other generative AI tools like Google's Bard[7] and Anthropic's Claude[8] have amplified these concerns and raised new questions regarding the societal implications of generative AI, including the risks associated with plagiarism, misinformation, and propaganda (Kreps et al., 2022, Thorp, 2023). We provide guidance to help researchers understand the trade-offs involved in choosing between proprietary and open-source models and discuss how to evaluate models for potential biases.

Overall, transformer-based LLMs have enormous potential for the social sciences. These models promise high accuracy at a relatively low cost, as a single expert analyst can feasibly develop a

training dataset to fine-tune an LLM for new classification tasks within a relatively short period (Do et al., 2022). The goal of this article is to systematically evaluate how model selection, learning regime, and task complexity impact predictive performance and to provide recommendations for sociologists interested in using LLMs for supervised text classification and related tasks.

2.2 Stance detection in social media posts

We evaluate LLMs for supervised text classification through a case study focused on a task known as stance detection (Somasundaran and Wiebe, 2010). The goal of stance detection is to identify the attitudes or opinions expressed towards a target, such as a politician or policy. The methodology was developed by computer scientists but has recently been adopted by computational social scientists interested in analyzing opinions expressed online (see Aldayel and Magdy, 2021 for a recent review). We expect it will be a particularly fruitful tool for sociologists as the technique can be used to measure attitudes, beliefs, and opinions expressed in texts.[9]

To perform stance detection, documents are typically annotated with labels like "Support/Oppose" or "Favor/Against," along with a third category, such as "Neither," to capture ambivalent or irrelevant texts (Somasundaran and Wiebe, 2010, Mohammad et al., 2016, Aldayel and Magdy, 2021). Many readers will be more familiar with a related technique known as sentiment analysis, which is used to categorize the valence or overall tone of a text (e.g. "Positive," "Negative," or "Neutral"). Sentiment analysis has gained popularity among social scientists due to the availability of off-the-shelf sentiment lexicons and classifiers. Sociologists have used these tools to measure attitudes towards immigrants (Flores, 2017), women (Shor et al., 2015), and ethnic and racial minorities (Felmlee et al., 2020, Voyer et al., 2022). While stance and sentiment are often correlated (e.g. statements of support to use more positive language), it is common to observe mismatches. As Bestvater and Monroe (2022, 19) note, "political opinions are typically complex and multidimensional enough that it is trivial to express them either negatively or positively." They demonstrate that using sentiment as a proxy for stance can result in substantial measurement error, particularly when the correlation between sentiment and stance is weak. Unlike sentiment analysis, which is often applied using keyword lexicons, stance detection typically necessitates the development of custom classifiers trained on relevant annotated corpora (Sen et al., 2020, Bestvater and Monroe, 2022).[10] Sociologists have recognized some of these limitations and taken steps to validate sentiment measures. For example, Flores (2017) weights sentiment keywords by their proximity references to immigrants to help ensure that these keywords correspond to the target of interest. Nonetheless, we anticipate

Prompt	***Output***
Zero-shot	
Text: Donald Trump is the best candidate to lead our country! Stance: Text: Donald Trump should never be allowed to hold an elected office! Stance: Text: Hillary Clinton is the best candidate to lead our country! Stance: Text: Hillary Clinton should never be allowed to hold an elected office! Stance:	Positive Oppose Support Negative
One-shot	
Text: Donald Trump should never be allowed to hold an elected office! Stance: Oppose Text: Donald Trump is the best candidate to lead our country! Stance: Text: Hillary Clinton is the best candidate to lead our country! Stance: Support Text: Hillary Clinton should never be allowed to hold an elected office! Stance:	 Support Oppose

Table 1: **Zero-shot and one-shot stance detection using GPT-3 Davinci**

that stance detection would be a more appropriate measurement strategy in most sociological applications of sentiment analysis.

The requirement for domain-specific, custom classifiers makes stance detection an ideal case to evaluate. The process of developing a training dataset and evaluating a classifier can be costly and time-consuming, but advanced LLMs can be adapted relatively easily with only a few lines of code and promise to achieve strong performance with relatively few high-quality examples (Do et al., 2022). Earlier work showed how neural network architectures can achieve strong performance against stance detection benchmarks (Augenstein et al., 2016) and recent work demonstrates that LLMs achieve reasonable accuracy in zero- and few-shot settings (Allaway and McKeown, 2020, Burnham, 2023). Before turning to our main analysis, we briefly preview how LLMs can be used for stance detection with minimal training.

The top panel of Table 1 shows four hypothetical statements expressing stances towards Donald Trump and Hillary Clinton. We use these statements to conduct a set of stylized stance detection experiments using GPT-3 Davinci, the most powerful model evaluated below (Brown et al., 2020). In each case, the statement is concatenated with a prompt, "Stance: ". These texts are then input into GPT-3. In all cases, the output from the model consists of a single word, demonstrating how

the prompt helped to constrain the response. In this zero-shot set-up, the model is generally correct but conflates stance and sentiment, returning "Support" and "Oppose" in two cases and "Positive" and "Negative" in the other two. To address this problem, we repeat the latter two examples using one-shot learning. The bottom panel of Table 1 shows the prompts and output. In each case, a completed example is passed to the model along with the text and prompt. Note how this information has steered the model towards providing a stance rather than a sentiment in both cases.

These examples illustrate the potential for LLMs to tackle difficult classification tasks, showing how these models can be adapted to new tasks with relative ease. Stance detection is a particularly instructive case for exploring the capabilities of LLMs because there are multiple ways to implement the task with varying levels of complexity. We consider two complementary datasets and show how the structure of each necessitates different modeling and evaluation metrics. We compare models trained to predict stance for a single target to those that predict it for multiple targets to assess how the complexity of the classification task impacts predictive performance. Across these tasks, we consider how performance varies as a function of the model architecture and learning regime.

3 Data

We use two datasets containing social media posts annotated for the stance towards the two leading candidates in the 2016 US Presidential election. This is an ideal case to explore because the election was a classic example of what John Zaller (1992) calls a "two-message issue," where most people tend to pick a particular side or, in this case, a candidate. By comparing two datasets from different social media platforms, we can assess different approaches to stance annotation and the extent to which observed variation is a function of the task specification or the context.

The first dataset consists of 1,691 tweets annotated for their stance towards Donald Trump or Hillary Clinton (Mohammad et al., 2016). Each tweet corresponds to a single target (Trump or Clinton) and is labeled with one of three stances: "Favor," "Against," or "None." This dataset has been widely used as a stance detection benchmark by machine learning researchers.[11] The target distribution is imbalanced with 707 tweets mentioning Trump and 984 mentioning Clinton (see Table A1 in the Appendix). We hold out 339 tweets (20% in total) for testing, balanced evenly across the two candidates. There is also an imbalance with respect to the stances. Tweets mentioning Trump tend to favor his candidacy, whereas those mentioning Clinton tend to be against her. Across both targets, tweets with the Against label occur more than twice as often as those annotated as Favor. In our analyses, we consider how this class imbalance impacts predictive accuracy.

The second dataset is a random sample of 2400 top-level comments (i.e., not replies to other comments) written on the Facebook pages of Donald Trump and Hillary Clinton during the election campaign. Each comment was annotated by a single author with expertise in the topic. Following (Do et al., 2022), we consider this to be a realistic setting for many applied researchers who may not have the resources to invest in the annotation of large datasets, which often necessitates training research assistants or weeding out poor performers on crowdsourcing platforms. Moreover, prior work demonstrates that larger corpora with a single annotation can be preferable to multiple annotations per example (Barberá et al., 2020). Unlike the SemEval dataset, which restricts each tweet to a single target, each comment was annotated for its stance towards Trump *and* Clinton. For consistency, we use a three-category annotation scheme similar to the SemEval task: "Favor," "Against", "Neutral/None."[12] The Favor or Against labels are only used if a comment expresses a stance toward a specific target, i.e., we do not infer that an expression of support for Clinton implies opposition to Trump. This process yields a tuple for each comment, such that a comment like "I'm never voting for Trump!" is associated with stance tuple {Against, Neutral/None}. There are 1200 comments from each Facebook page, but the distribution of the annotations is imbalanced (see Table A2 in the Appendix). Tweets favoring Trump outnumber those against him more than two-to-one, whereas the inverse is true for Clinton. Only a small fraction of comments, 7%, express a stance towards both candidates. The most frequent class is comments with no stance towards either candidate. We use up to 2000 comments for training and reserve 400 for testing.

4 Models

We compare four large language models based on transformer architectures (Vaswani et al., 2017). These models vary with respect to size, training, and capabilities. Each model could be a reasonable choice for an applied research setting, and our goal is to understand how they differ when applied to a common task. The key aspects of each model are summarized in Table 2.

We start with BERT (Devlin et al., 2019), which is trained as a masked language model, where the objective is to predict a missing word in a sentence. For example, "the cat [MASK] a mouse." BERT is bidirectional, meaning that its training involves parsing a sentence in both directions, right to left and left to right, which improves predictive performance.[13] BERT is likely the most widely used LLM by social scientists, featuring in several recent sociological analyses (Ren and Bloemraad, 2022, Bonikowski et al., 2022, Le Mens et al., 2023). We use `bert-base-uncased`, the smaller version of the model. It is relatively modest in scale compared to more recent language models,

Model	BERT	BART-MNLI	GPT-3 Davinci	GPT-3 Ada
Parameters	110M	407M	175B	350M
Interaction	PyTorch	Huggingface	OpenAI API	OpenAI API
System	Personal laptop	HPC cluster	Third-party server	Third-party server
Open-source	✓	✓	✗	✗

Table 2: **Large language model comparisons**

with 110 million parameters and twelve layers of transformer modules. Unlike more recent models, BERT cannot be prompted using text inputs, so cannot perform zero-shot or few-shot learning. Instead, it is fine-tuned by modifying its neural network structure. We add a classification layer with randomly initialized weights. Each text is converted into BERT's embedding representation and input into the model, along with the associated stance labels. The weights in the classification layer are adjusted to minimize the cross-entropy between the predicted and true labels using the Python library PyTorch (Paszke et al., 2019).[14] This was implemented using a MacBook Pro laptop with an M1 Max 32-core GPU and 64GB of RAM and took approximately five minutes for each model.

The second model, BART (Lewis et al., 2019), was developed by researchers at Meta. It is trained on the same dataset used to train RoBERTa (Liu et al., 2019), an extension of BERT using a larger training dataset and additional fine-tuning that achieved strong performance on benchmarks. We use `bart-large-mnli`, a 406M parameter variant of BART with additional fine-tuning on the Multi-Genre Natural Language Inference (MNLI) corpus, a crowd-sourced dataset of 433k sentence pairs annotated with textual entailment information to enable the model to perform zero-shot learning (Yin et al., 2019), which adds two classification layers and 1M parameters. BART-MNLI can be downloaded from HuggingFace, a company that provides free access to state-of-the-art transformer models and an accompanying Python library (Wolf et al., 2020).[15]. To fine-tune the model, we use a high-performance computing (HPC) cluster with an Intel Xeon Gold 6230, a Tesla V100-32GB GPU, and 192GB of RAM. The time elapsed when fine-tuning each model was less than 30 minutes.

The third and fourth models are variants of GPT-3 (Generative Pre-trained Transformer), released by OpenAI in 2020. It is the third iteration of LLM developed by the company, beginning with GPT in 2018 (Radford et al., 2018). GPT-3 is one of the largest language models to date and performs well across various tasks, demonstrating particularly strong performance in zero- and few-shot settings (Brown et al., 2020). OpenAI has released several variants of the GPT-3.[16] We compare `text-ada-001` and `text-davinci-003` — hereafter Ada and Davinci — released in November 2022. Ada, the smallest variant, has 350 million parameters and is trained on 40GB of text data. Davinci is the largest model, with 175 billion parameters, trained on 45TB of text. The Davinci

model is also known as InstructGPT, as it has undergone additional refinement using RLHF to improve its performance (Ouyang et al., 2022).

Unlike BERT and BART-MNLI, these models are not open-source and cannot be downloaded. Instead, we pay to use OpenAI's API to interact with models running on the company's servers. OpenAI charges more for the larger, more computationally-expensive versions. At the time of writing, for every 1000 tokens of input — equivalent to approximately 750 words — Ada costs $0.0004, and Davinci costs $0.02. In many settings, the goal of supervised text classification is to apply a trained model to a large corpus. A realistic task might involve classifying a million or more comments on a social media platform. If we have a corpus of 1M documents, each consisting of 50 words, this will equate to $0.02 for every 15 documents or a total cost of $400 using Davinci, compared to just $8 for Ada. Many researchers may prefer the cheaper base model to save costs. Indeed, OpenAI recommends using Ada, noting how it performs "only very slightly worse than more capable models once fine-tuned, whilst being significantly faster and cheaper."[17] We compare these two variants to assess whether it is worthwhile to pay fifty times as much for a larger model. OpenAI has recently released an even more powerful model, GPT-4 (OpenAI, 2023), but it is even more expensive to use (between $0.03 and $0.12 per 1K tokens, depending on the variant and task) and cannot currently be fine-tuned.

Each GPT-3 model has a "temperature" hyperparameter controlling the degree of stochastic variation in the output. The default temperature of 0.7 is suitable for generative applications because it ensures that the output varies, even with identical inputs. Since this is a classification task, we set the temperature to zero so the model returns the highest probability sequence of tokens, avoiding random variation in the predicted classes.[18] Other hyperparameters are set to defaults.

Prior work shows that transformer-based models perform favorably compared with conventional machine learning algorithms across a range of tasks relevant to social scientists (Widmann and Wich, 2022, Bonikowski et al., 2022, Wankmüller, 2022). To assess how the LLMs evaluated here perform relative to other approaches, we calculate baseline scores for a subset of the specifications using four different models, varying the feature representation (bag-of-words vs. embeddings) and learning algorithm (support vector machine vs. neural network). These baseline models are described in more detail in subsection A.2.

5 Experiments

5.1 Stance detection tasks

Stance detection can be conceptualized in several different ways. The most straightforward task is to predict the stance towards a single target (ST). In this case, we train separate models to identify the stance towards Hillary Clinton and Donald Trump using each dataset. For the Twitter dataset, we segment the data by the target. For the Facebook dataset, we train models on the entire dataset using only one of the stance labels. This task does not require a model to identify the target at which a stance is directed but is nonetheless challenging. A recent study showed that many off-the-shelf sentiment and stance detection classifiers performed poorly even when the target was held constant (Sen et al., 2020).

A more complex approach, multi-target (MT) stance detection, involves using a single model to jointly predict the stance towards multiple targets. The Twitter dataset is structured such that each tweet has an annotation corresponding to a target and a stance (e.g. "Clinton, Against"). This task thus involves the identification of both a target, with two options, and a stance, with three possibilities. Since each Facebook post is annotated with a stance toward both targets, we do not need to conduct target detection but only predict the stance tuple (e.g. "For [Trump], Against [Clinton]"). Given the three stance labels, there are nine possible pairings. While these tasks are more complex than ST stance detection, a long-standing stream of machine learning research under the umbrella of "multitask learning" suggests that classifiers can benefit from sharing information across multiple tasks (Caruana, 1997). For example, learning to predict the target could help to predict the stance and vice versa.

5.2 Training data and learning regimes

We test several frameworks for each task to examine the relationship between the size of the training dataset and model performance, beginning with zero-shot learning and moving toward fine-tuning. For the Twitter task, we compare zero-shot, few-shot, and models fine-tuned on 10, 100, and all 1351 training examples. In the zero-shot case, the model is shown a prompt and makes a prediction for each example in the test data. We use two variations of few-shot learning. For the ST task, a single tweet corresponding to the relevant target is sampled from the training data and provided along with a prompt. For the MT task, there are two possible targets, so the model is provided with one example pertaining to each target from the training data. The models using 10, 100, and all

examples are trained using a different process. Rather than providing the examples at the time of prediction, the models are fine-tuned using the examples without any prompts, akin to a standard text classification workflow. This is more computationally efficient since it is costly to repeat the prompt and examples each time a prediction is made. There are also limits on the input size that make it impractical to conduct few-shot learning with large numbers of examples — GPT-3 has a maximum of 2049 tokens for the prompt, examples, and returned text combined. For the Facebook task, we conduct the same analyses with two modifications. Since each comment is annotated for both targets, we use one- rather than two-shot learning for the MT task, providing a single example including both stance annotations and a prompt. For the fine-tuning, we additionally compare models trained on 1000 examples and the full dataset, consisting of 2000 examples, enabling us to assess the relationship between training data size and performance for larger samples.[19]

5.3 Prompt engineering

For zero- and few-shot learning, it is necessary to provide information to constrain the model to produce the desired output. The examples in Table 1 show that simple prompts can work for easy examples, but to achieve better performance, we must provide more information on the task and the format of the output (Brown et al., 2020). This is somewhat analogous to giving instructions to human annotators to classify texts appropriately, although there is no guarantee that the model will respond in the intended way.[20] This type of prompting is only possible with the most recent families of transformer-based text-to-text large language models.

To demonstrate the importance of careful prompt engineering, we test three prompts for the two MT classification tasks. We begin with a simple prompt listing the options and the format of the answer (see Table A3 for the full prompts). These prompts are extended by adding a short sentence providing further context. The final pair of prompts is more informative, indicating that statements refer to politicians and represent attitudes. This allows us to assess the extent to which additional context enhances the accuracy of the model. Ceteris paribus, more context should improve performance, although shorter prompts are preferable since the longer prompts consume more tokens. These experiments allow us to assess trade-offs between prompt complexity, predictive accuracy, and economic costs. The best-performing prompts from these experiments are used in the relevant one- and few-shot prediction tasks.

For the one- and two-shot models, we also examine the sensitivity to the examples included with the prompt. Prior work finds that some test examples, particularly those that do not explicitly mention

a target, are considerably harder to predict (Sen et al., 2020, Burnham, 2023). It is thus plausible that some examples may be more helpful than others for distinguishing between classes when used in few-shot settings. For each task, we conduct 100 replications using different examples from the training data. In each case, we take a simple random sample without replacement from the training data (stratified by the target for the two-shot task to ensure one example from each target is shown).[21] Each example is concatenated to the prompt and used to predict the labels for all examples in the test dataset. These experiments enable us to understand the extent to which the predictive performance of few-shot learning varies according to the example(s) provided.

5.4 Evaluation metrics

All models are scored by calculating predictive performance on the held-out test data.[22] The F1-score is used in machine learning to measure the overall predictive performance of each classifier. It is the harmonic mean of precision and recall. Ideally, we want to achieve high precision and recall, although there are cases where it may be preferable to optimize for one over the other (Jensen et al., 2022, 46). For each class, *precision* is the number of true positives divided by the number of true positives and false positives. For example, what proportion of the comments predicted to support Trump are correctly classified? *Recall* is the number of true positives divided by the number of true positives and false negatives. For example, what proportion of *all* the tweets labeled as supporting Trump were classified as supporting Trump? Precision thus captures how accurately a classifier detects a particular class, whereas recall measures how many relevant examples were detected. The F1 score for class k is defined as

$$F1_k = 2\frac{precision_k \cdot recall_k}{precision_k + recall_k}$$

We use a weighted F1 score to calculate the overall performance for each target, $F1_{ST}$. This is calculated by taking a weighted average over each class, where K is the number of classes and N_{test_k} is the number of examples in the test set belonging to each class, and N_{test} is the size of the test set:

$$F1_{ST} = \sum_{k=1}^{K} F1_k \cdot \frac{N_{test_k}}{N_{test}}$$

For example, $F1_{Clinton}$ is a weighted average of the F1 scores across three classes, "For," "Against," and "Neither/None."We also calculate overall scores across both targets, where the latter is defined as the weighted average of the two candidates' F1 scores:

$$F1_{Overall} = F1_{Clinton} \cdot \frac{N_{test_{Clinton}}}{N_{test}} + F1_{Trump} \cdot \frac{N_{test_{Trump}}}{N_{test}}$$

Finally, we also calculate a stricter $F1_{Joint}$ score, which counts a prediction as correct when it exactly matches the original annotation, using the same formula as the single target F1 score. For the Twitter task, this implies that both the target and stance are correctly predicted. In this case, $K = 6$, as there are two possible targets and three stances for each. For the Facebook task, the stances must be correct for both targets, thus $K = 9$, since there are three stance scores for each target and hence nine possible combinations.[23] Additionally, we construct confusion matrices for selected models to visualize error patterns and how they vary across training regimes.

6 Results

6.1 Zero-shot prompt engineering

For each task, we evaluated three prompts described in Table A3. The results of the experiments are shown in Table 3, which displays the single-target and overall F1 scores for each prompt evaluated on the relevant test data, with the best scores in each task in bold. There is more variation for the Twitter task, with the second prompt achieving a slightly better score for tweets where Trump was the target but scoring worst on those where Clinton was the target. The most informative prompt received the highest overall F1 score. For the Facebook task, the most informative prompt performs best across all tests. Overall, these results indicate that more informative prompts increase predictive accuracy. While we do not attempt to exhaustively identify the best prompt, these results underscore the importance of careful prompt selection. However, it is also important to emphasize the trade-offs between accuracy and cost. Comparing Facebook prompts 2 and 3, there is an 11% gain in overall predictive performance (0.55 vs. 0.61), but the prompt is almost 40% longer (43 vs. 60 tokens) and thus will cost 40% more.

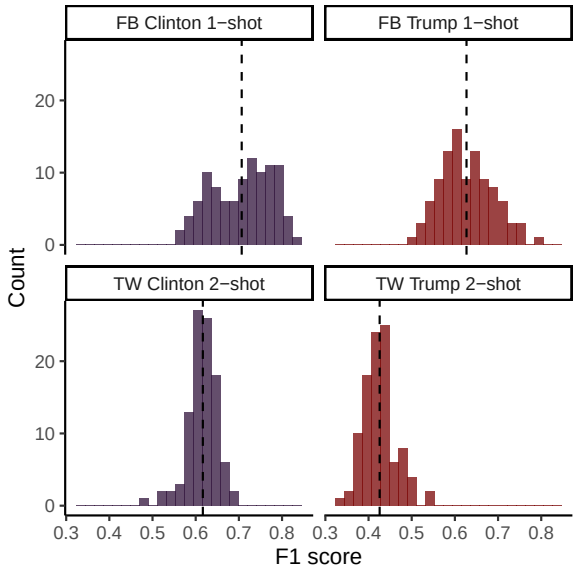
Model	Test Data	Prompt	Trump	Clinton	Overall
GPT-3 Davinci	Twitter (MT)	1	0.42	0.54	0.49
		2	**0.43**	0.49	0.46
		3	0.41	**0.60**	**0.52**
	Facebook (MT)	1	0.59	0.66	0.62
		2	0.71	0.71	0.71
		3	**0.77**	**0.72**	**0.75**

Table 3: **Zero-shot F1 scores by prompt variations.** *Note: The best scores for each dataset and target are in bold.*

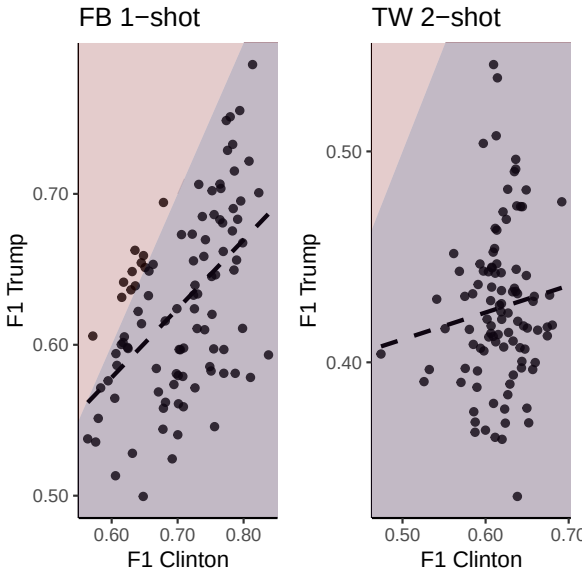
6.2 One- and two-shot example selection

The one- and two-shot analyses were performed using the most informative prompts from the previous section. Figure 1 shows the variation in performance across examples used in the one- and two-shot tasks. Across both datasets, there is substantial variability in predictive performance on the test data, highlighting sensitivity to the examples. The F1 scores for the Twitter task range from 0.33 to 0.69, and the scores for Facebook range from 0.50 to 0.84. In general, the models tend to be more accurate at predicting the stance towards Clinton than Trump, an issue we discuss further in the following sections. For the one-shot Facebook model, we observe a strong, positive correlation between the F1 scores for each target. This suggests that the best examples help improve predictions for both candidates. The correlation between the target-specific F1 scores is much weaker for the Twitter model, potentially due to the additional randomness induced by the two-shot setting, where examples were sampled independently for each target. The shaded regions further illustrate how almost all models (all of those used on the Twitter test data) performed more accurately for Clinton than Trump.

To better understand how the selection of examples impacts performance, we estimated a series of regression models, using the example word count, stances, and the interaction between the stances to predict the target-specific F1 scores. The full regression results are reported in the Appendix in Table A4 and Table A5. Overall, we find that examples that favor one or the other candidate are associated with in statistically-significant increases in performance across the two Facebook tasks: Comments supporting Clinton are associated with a 0.05 increase in the Clinton F1 score and a 0.07 increase in the Trump score compared to those with no stance towards Clinton; comments favoring Trump are associated with a 0.03 increase in the Trump F1 score. The R-squared values increase when interaction terms are added, and the stance interactions for the Trump task are statistically-significant, highlighting how the combination of stances expressed in an example



(a) F1 distributions by target. Dashed lines indicate means.



(b) Correlations by target. Shaded regions indicate when models perform better for Trump (red) or Clinton (blue). The regression line is indicated in black.

Figure 1: **One/two-shot example variation and predictive performance for MT tasks.**

matters. The patterns in the Twitter predictions are less clear but suggest that longer tweets help to improve the Clinton predictions and that tweets favoring Trump improve the Trump predictions. Overall, these results highlight how LLM classifiers using few-shot learning can be highly sensitive to the choice of examples, with evidence of wide variation in the out-of-sample F1 scores across different examples.

6.3 Single-target and multi-target stance detection

6.3.1 Twitter task

We now assess the predictive performance of each model across various training regimes for each task, beginning with the single-target (ST) Twitter results. These results are summarized in the first three panels in the left column of Figure 2 (Table A6 in the Appendix shows the F1 scores for each task). The zero-shot results show that some models can perform reasonably accurately at predicting the stance for a single target without any training examples. Davinci shows consistent performance across both targets, whereas BART-MNLI performs substantially better at predicting the stance toward Clinton. Ada, on the other hand, performs poorly across both tasks. Analysis of the output shows that the model often returned alternative labels outside of those requested. These

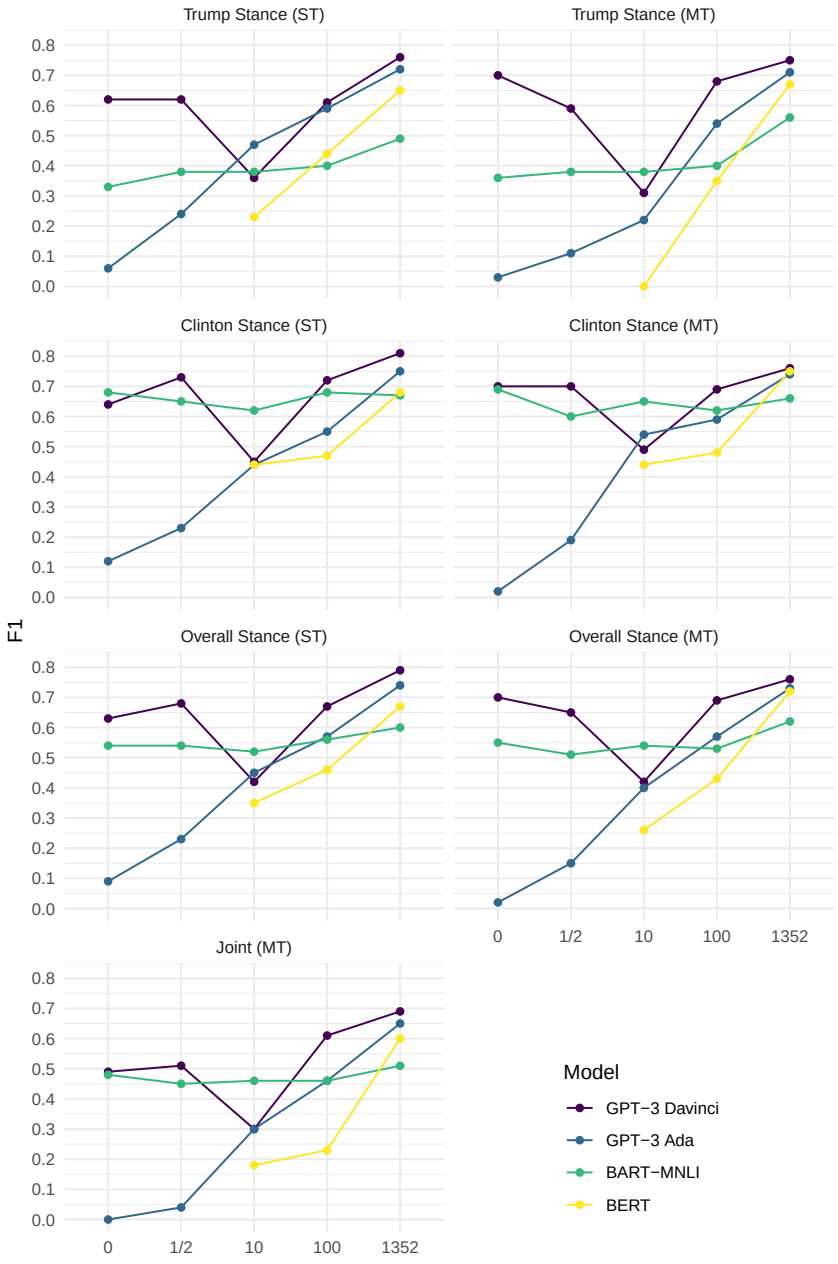


Figure 2: **F1 scores for single-target (ST) and multi-target (MT) stance predictions using Twitter data**

out-of-scope outputs often repeated the prompt, yielding text such as “This statement contains an attitude expressed about,”. This demonstrates a limitation of zero-shot learning, as there is no guarantee that a model will return the desired output. Turning to the one-shot tasks, we also see substantial variation in the performance across models and targets.[24] The Davinci predictions for Clinton improve, resulting in an aggregate increase in accuracy. BART-MNLI, on the other hand, shows some improvement on the Trump task but a decline in accuracy for the Clinton task. The accuracy of the Ada model improves in all variations, with gains of 0.11 to 0.18 compared to zero-shot, but remains poor compared to the larger models.

The scores for the multi-target (MT) tasks — which predict both target and stance — are shown in the right-hand column of Figure 2.[25] Davinci and BART-MNLI perform slightly better at zero-shot MT stance classification compared to the ST variants. Moreover, these models perform comparably to conventional baselines trained on the entire corpus, with the best model obtaining a joint F1 score of 0.48, equal to BART-MNLI’s score, demonstrating the potential of zero-shot learning. Ada, on the other hand, performs poorly, with even worse performance than the ST task, and often returned out-of-range targets such as "Barack Obama." The performance on the two-shot Davinci, Ada, and BART models was consistently worse than that of the one-shot ST models, indicating that more information is required to make more complex predictions for both target and stance. It may also be the case that the information contained in both tweets could obfuscate the task, making the prompt less effective.

The impact of fine-tuning varies across each of the models considered. Across both ST and MT tasks, the Davinci model shows a drop in performance when comparing the 10-shot to zero and few-shot models. This highlights the importance of the prompt text for settings where training data are sparse. Indeed, OpenAI advises providing at least hundreds of examples when fine-tuning a model.[26] Even using 100 training examples, the performance of the Davinci model is not substantially better than zero- or few-shot models when evaluated on various subtasks, but when considering the strictest evaluation metric, shown in the bottom left, we see that this fine-tuned model outperforms the others. Davinci’s predictive performance improves with more training, as the models trained on all examples achieve the best performance across all tasks. Unlike its larger sibling, Ada’s performance continually improves with additional training. Indeed, for the models fine-tuned on 10 examples, Ada outperforms Davinci on some variations. The training examples rather than the prompt thus appear to have more impact on the predictive performance of the smaller GPT-3 model. The results show evidence of roughly linear increases in accuracy as the number of training examples increases

on a logarithmic scale. The BERT model shows a similar pattern but only achieves comparable performance to Ada when fine-tuned on the entire training dataset. BART-MNLI, on the other hand, shows little evidence of improvement with additional training — despite having been trained on more text and having four times as many parameters than BERT — suggesting that the optimization for zero-shot prediction hinders its capacity to improve with fine-tuning. Nonetheless, all four fine-tuned models outperform the baseline models, reported at the bottom of Figure 2.

The confusion matrices in Figure 3 summarize the results from three of the GPT-3 Davinci MT models. Each heatmap shows the density of predictions across the possible combinations of actual and predicted stances. For example, the top-left cell in the first heatmap shows that 81% of the tweets labeled with the stance "Trump, Favor" were correctly predicted by the zero-shot model. The cell immediately to the right implies that the remaining 19% of these tweets were miscategorized as against Trump. An optimal model should exhibit most of the density along the diagonal. Note how the direction of the errors varies across specifications. In the zero-shot model, we see high accuracy for certain classes but more density spread across the off-diagonals. As the models are fine-tuned on more data, as in the 100-shot model in the middle, there are improvements in stability, with most of the density around the diagonal. In the final model, we can see more clearly how the error structures vary across the two candidates, even after fine-tuning. The model is highly accurate at predicting whether tweets favor Trump, getting 85% of these correct, but tends to misclassify tweets against Trump as either favoring his candidacy or expressing no stance, only getting 57% of these correct. We observe the opposite pattern for Clinton, as 24% of tweets labeled as favor are predicted to be against her, resulting in lower accuracy overall, although this is still a big improvement compared to the model with 100 examples, where 44% of tweets favoring Clinton were predicted to have the opposite stance.

6.3.2 Facebook task

The results of the ST and MT tasks for Facebook are shown in Figure 4, with detailed scores listed in Table A7 in Appendix. Unlike the Twitter experiments, all models use the same training data, but the ST models were only tasked with predicting the stance towards one target, whereas the MT models were provided with a joint label for Trump and Clinton. The general patterns are similar to those from the previous results: Davinci far outperforms other models at the zero- and few-shot tasks and achieves the best accuracy across all tasks except for the case with ten observations; predictive performance increases as the models are fine-tuned using more data; BART-MNLI

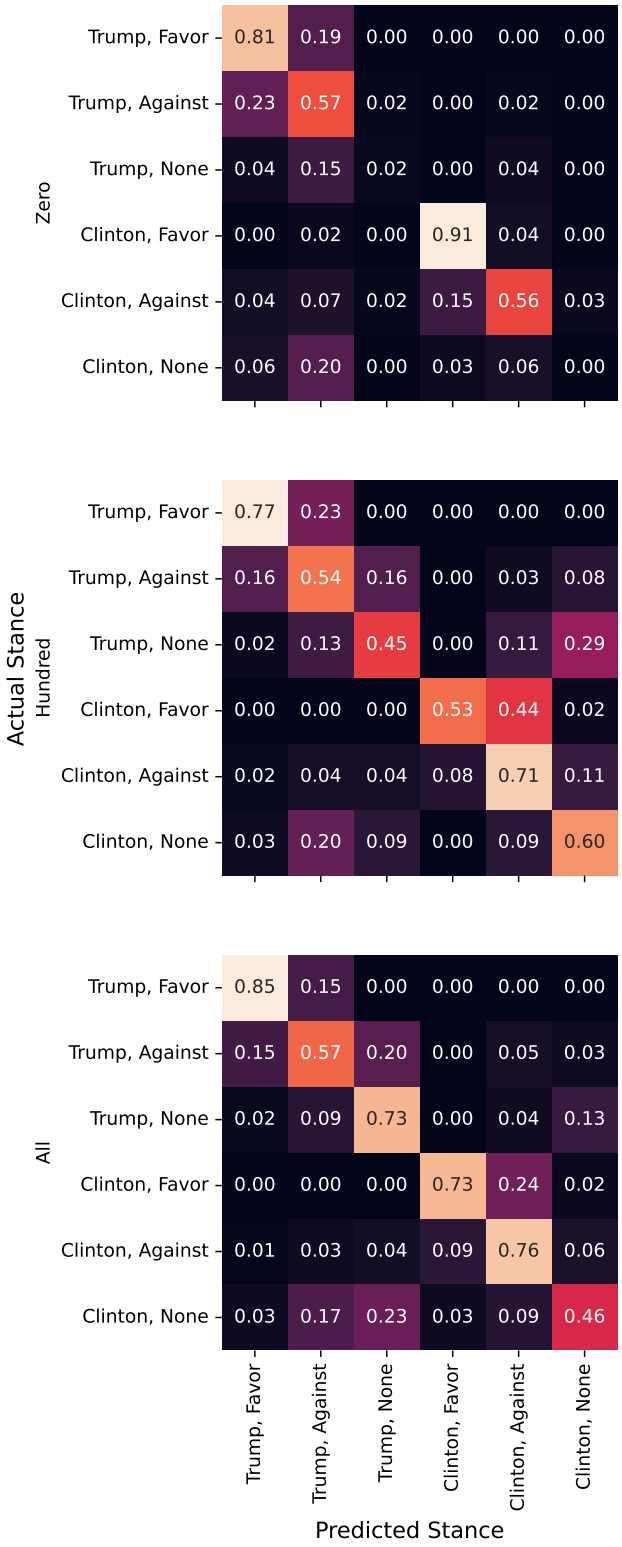


Figure 3: **Confusion matrices for zero-shot and fine-tuned on 100 and all examples for Twitter MT task (GPT-3 Davinci)**. *Note: Row percentages may not sum to one due to out-of-range predictions.*

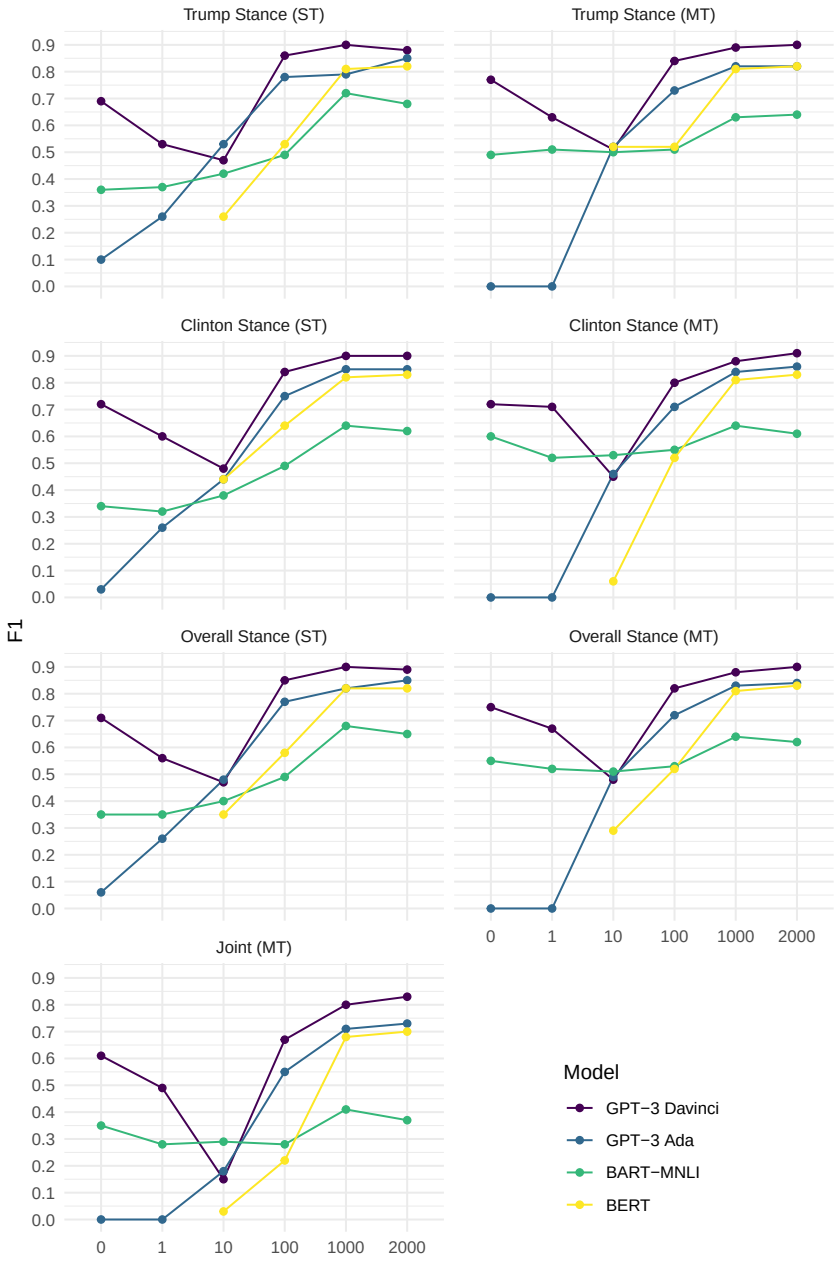


Figure 4: **F1 scores for single-target (ST) and multi-target (MT) stance predictions using Facebook data**

performs poorly when fine-tuned. Turning to the comparison between the ST and MT predictions, the results show comparable performance within-targets (e.g. F1 0.90 vs. 0.91 Clinton predictions from Davinci fine-tuned on all examples). Davinci performs slightly better when training using both stance labels, and the Ada, BART-MNLI, and BERT MT models tend to perform comparably to their ST counterparts. With regard to the baseline models, Davinci again outperforms the best conventional model when used for zero-shot learning, Ada scores better when fine-tuned on 100 or more examples, and BERT scores better after 1000 or more. While the BART-MNLI model outperforms some baselines when fine-tuned on 1000 examples, the final model fares worse than all four baselines.

The discrepancies across the Twitter tasks may be more a function of the differences in data composition than model performance since the Facebook data allow us to vary the labels while holding the training data constant, whereas the size of the Twitter dataset varies across the tasks due to the annotation scheme. The Facebook models also tend to achieve higher F1 scores across most tasks than the Twitter models. This may be attributable to the fact that Facebook comments are longer, conveying more relevant information to the classifier, and because the task does not involve predicting a target. Nonetheless, it is a more complex prediction task, with the Joint (MT) scores evaluating the predictive accuracy across nine possible outcomes compared to six for the Twitter task.

Like the Twitter results, there is evidence that few-shot models can perform worse than prompts alone. The Davinci 1-shot classifier performs worse, on average, than the zero-shot classifier on both ST and MT tasks. This implies that some examples may be detrimental to performance compared to the prompt alone, perhaps due to their greater length and complexity. Nonetheless, the results in Figure 1 show that careful choice of examples can substantially improve predictive performance. Like the Twitter experiments, the Ada results show substantial improvement in the 1-shot setting compared to zero-shot predictions, so it is unclear how far this generalizes to other LLMs.

The larger dataset also allows for two extensions of the previous findings. The comparison between models fine-tuned on 1000 and all 2000 comments suggests evidence of diminishing returns to additional training data, as there are marginal improvements in the overall F1 scores despite a doubling in the size of the training data, consistent with earlier work (Miller et al., 2019, Do et al., 2022, Wankmüller, 2022). It also enables us to better compare BERT and the GPT-3 models, showing that the former essentially converges with Ada after fine-tuning on 1000 or more comments.

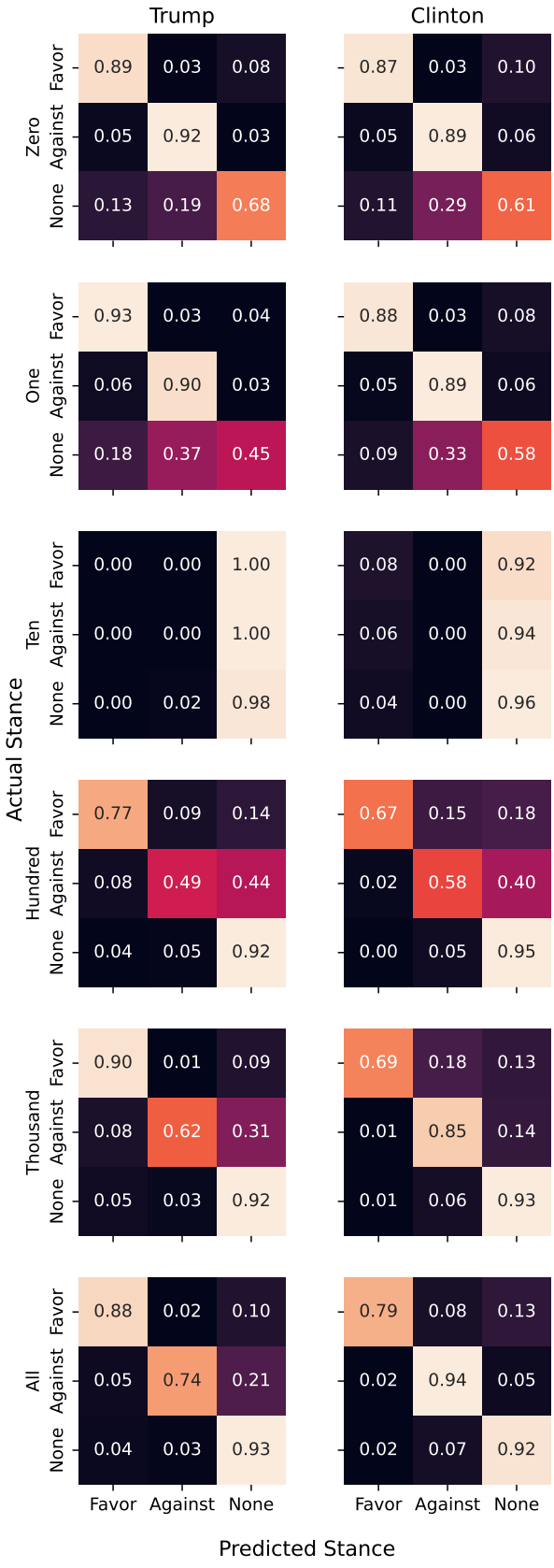


Figure 5: **Confusion matrices for Facebook MT task (GPT-3 Davinci)**

The heatmaps in Figure 5 exhibit similar error patterns to the Twitter task. Since each target gets a prediction in the MT task, there is a separate heatmap for each target. These results illustrate why the 1-shot models perform worse than the zero-shot models, as we can see that more comments with stance "None" are misclassified. Moreover, the model fine-tuned on ten examples predict "None" for almost all examples for both candidates, hence the substantial drop in overall accuracy. But with fine-tuning, we see a gradual convergence around the diagonal. Nonetheless, the error patterns reveal that the models fine-tuned on large samples fare worse than the zero-shot models in some respects. For example, the zeros-shot model correctly identified 92% of the comments against Trump, whereas the model fine-tuned on one thousand comments only predicted 62% of these accurately, mistakenly labeling 31% as having no stance towards him. The model fine-tuned on the entire corpus shows some improvement but still has lower accuracy for anti-Trump tweets than the zero-shot model. The results also show similar patterns in the direction of errors across candidates. The model is more accurate at identifying anti-Clinton than anti-Trump comments but less accurate at identifying pro-Clinton than pro-Trump comments. As such, there is some systematic bias in the predictions that will lead the model to underrepresent *opposition* to Trump and *support* for Clinton by mistakenly predicting that anti-Trump tweets and pro-Clinton tweets have no stance towards the candidates. Such errors occurred 21% and 13% of the time, respectively.

7 Discussion and Conclusions

7.1 Findings

Large language models represent an important methodological innovation that opens up new ground for the sociological study of texts. We examine the performance of LLMs for text classification, a form of supervised machine learning used to automatically sort texts into categories based on the schema defined by the analyst. Stance detection is a particularly useful form of text classification for sociologists since we often want to use texts to infer attitudes, opinions, and beliefs, and is more suitable to the task than sentiment analysis, which is often misapplied (Bestvater and Monroe, 2022). Our findings have most direct relevance to research in political sociology, since our case study focuses on the positions taken toward leading candidates on Twitter and Facebook during the 2016 Presidential election, but these techniques can easily be extended to other domains. LLMs can be adapted for any text classification task and the capacity to perform zero- and few-shot learning make it simple to experiment with the capabilities of these models across a range of different tasks.

Our results show that LLMs can accurately identify stances in social media texts, outperforming conventional machine learning approaches using both bag-of-words and embedding representations. Models trained to predict stance across multiple targets or to predict stance and target typically performed as well or better than simpler models trained only to predict stance for a single target. This highlights how LLMs can be reliably applied to challenging classification problems. GPT-3 Davinci, the most powerful model tested — with the largest number of parameters and trained on the most data — can perform reasonably well in zero-shot settings without any task-specific training. Depending on the task, the addition of a handful of examples along with the prompt can further improve performance. But these approaches should be used with caution, as the predictive performance of these models is sensitive to both the wording of the prompt and the examples provided. Models fine-tuned on only ten examples generally fare worse, further demonstrating the importance of the prompt for zero- and few-shot tasks, but predictive accuracy generally improves with fine-tuning on more data. Smaller models tend to perform poorly unless fine-tuned on larger datasets, but after such training can approach the accuracy of larger models at a fraction of the cost. The smaller BERT model fine-tuned on the entire corpus performed almost as well as the GPT-3 models across most tasks. Although BART-MNLI is an extension of the BERT framework, it performs relatively poorly on the fine-tuning tasks, indicating that the additional fine-tuning for zero-shot learning hindered its capacity to function as a generalizable foundation model.

Before turning to our recommendations, it is important to note two limitations of the experiments reported here. First, our context — the 2016 Presidential election — is one that all of the models used may be "aware" of, in the sense that texts related to the election campaign was likely contained in their training data. LLMs are now trained on vast and varied corpora, but it is plausible that performance will be worse when applied to unfamiliar contexts. Second, while we consider the four models evaluated to be sufficient to outline key differences between various LLM frameworks, it is plausible that other related models would perform differently. More work is needed to systematically evaluate the growing number of LLMs available to academic researchers. Similarly, each model has various hyperparameters, such as the temperature parameter for GPT-3 models or the number of epochs that a BERT model is fine-tuned for. In most cases, we used the defaults, but it is plausible that varying these parameters could alter performance.

7.2 Recommendations

We use the results of our experiments to develop a set of recommendations to assist researchers in selecting the appropriate strategy for using LLMs for text classification tasks, summarized in Figure 6.

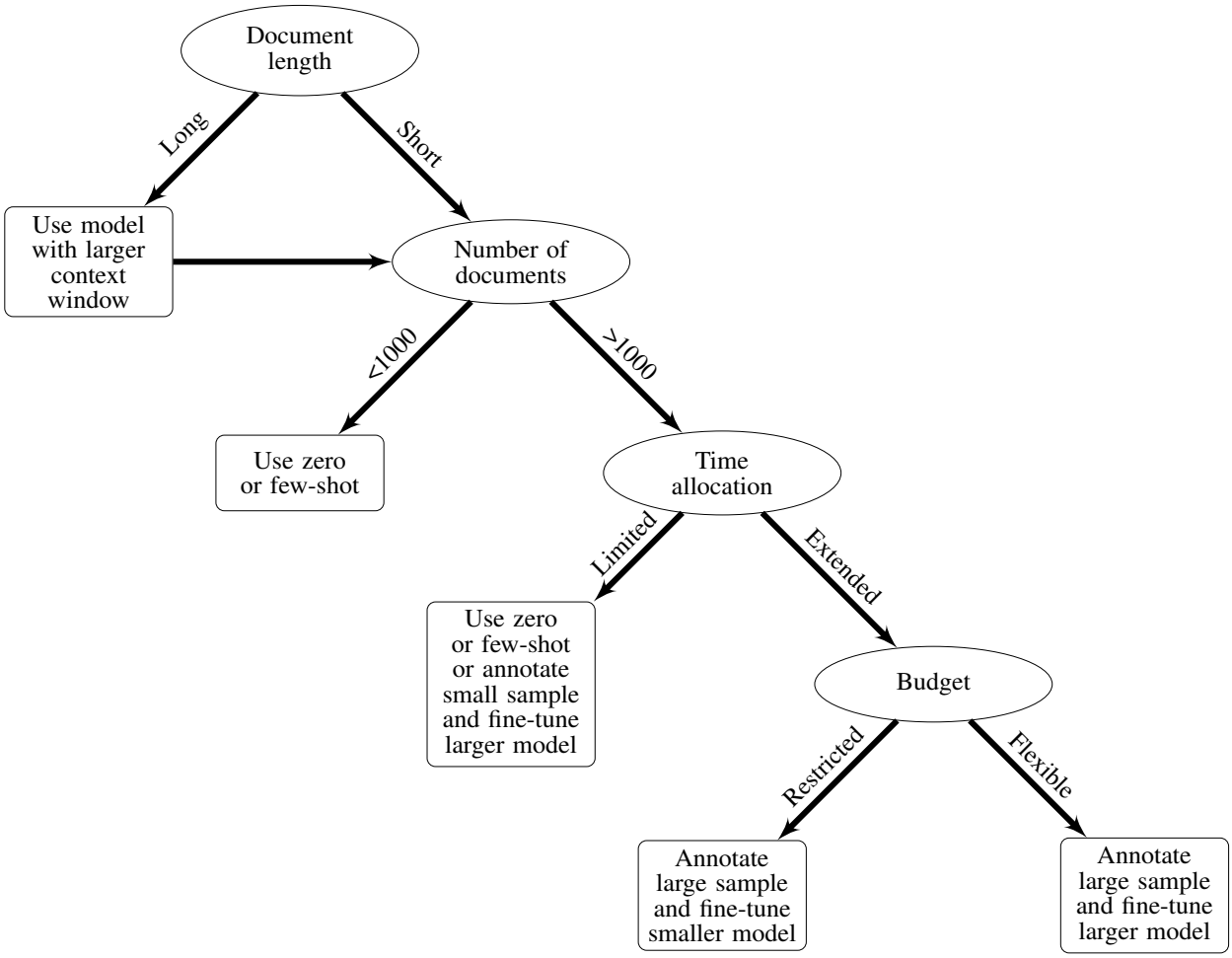


Figure 6: **Selecting a suitable approach to using LLMs for text classification**

The social media posts used in our experiments are mostly short, although some Facebook comments extend to several paragraphs. If the research design requires coding long documents consisting of multiple pages, such as newspaper articles, academic papers, or legal filings, then it will be necessary to use models with larger "context windows," which determine how much information can be processed by a model. Such models can handle long-range dependencies in texts, e.g. identifying how repeated mentions of a character across the pages of a novel correspond to the same entity. Whereas GPT-3 can handle just over 2k tokens, GPT-4 has a context window of 32k

tokens, and Anthropic's LLM Claude can process up to 100k tokens, equivalent to approximately 75,000 words.[27] Longer documents can, of course, be broken down into shorter segments. Often, sentences or paragraphs are more theoretically meaningful units of measurement (Barberá et al., 2020, Bonikowski et al., 2022), so we expect that the models considered here will be suitable for many sociologically-relevant text classification tasks even if documents are considerably longer than social media posts. Ultimately, the unit of analysis should depend on the research question and theoretical considerations.

The next consideration is the number of documents to be classified. If the sample is small, then zero- or few-shot learning is likely to be the preferred solution, as it will be relatively inexpensive to use an advanced model. Of course, one could also hand-code the data, but we expect these techniques will increasingly complement qualitative approaches (e.g. to validate or extend hand-coding). We recommend experimenting with different prompts and examples to optimize these approaches, as our experiments demonstrate that variations in the prompt and examples can have substantial impacts on both predictive performance and cost. If the number of documents is large, we recommend developing an annotated dataset for fine-tuning. In Figure 6, we use 1000 as a cut-off for large versus small-N, but the number is arbitrary, and it is up to the researcher to decide whether it is worthwhile developing a training dataset. Regardless of the approach to classification, we recommend annotating a small sub-sample of documents to validate any automated coding.

If the goal is to classify a large sample of documents, the annotation procedure and model selection will likely be constrained by the time and resources allocated to the task. If time is limited, we recommend annotating a small corpus and then using a larger, more powerful model. Our experiments above show that GPT-3 Davinci performs well on some tasks when fine-tuned on only 100 labeled documents. This is consistent with the approach proposed by Do et al. (2022). One could also experiment with zero- or few-shot learning, which can perform comparably for some tasks, but this approach will likely be costly to implement at scale since advanced models are necessary to achieve accuracy comparable to smaller models trained on more data. If you have more time but limited resources, we recommend annotating a larger corpus and fine-tuning a smaller LLM since our results show that smaller LLMs fine-tuned on modest corpora of annotated data can perform almost as well as much larger models. Of course, the most accurate results can be obtained by developing a large annotated corpus and fine-tuning a state-of-the-art LLM, but it is time-consuming and expensive to annotate data and fine-tuned large models. As such, we expect

that many researchers will be willing to sacrifice some accuracy to use a smaller model at a fraction of the cost.

No text classification model will perfectly identify the stance or any other quantity. Indeed, human raters can be inaccurate and inconsistent (Joseph et al., 2021). While optimizing predictive accuracy is the end goal in much of the research in computer science (Molina and Garip, 2019), the objective of most social science research is to use these tools for measurement (Grimmer et al., 2022). From this perspective, less accurate classifiers will provide noisier measurements, but the amount of noise may have little effect on the downstream estimates. As Burnham (2023) demonstrates, even zero-shot stance detection can generate similar conclusions to conventional machine learning classifiers when outputs are used as proxies for a quantity of interest in a regression task. Based on our results, we expect smaller, cheaper models trained on larger annotated corpora will be the optimal solution for many sociologically-relevant classification tasks.

There are several other considerations that we anticipate will guide model selection. The type of model may depend on the technical expertise of the research team. While the costs of GPT-3 compared to open-source solutions might seem prohibitive, API-based models can be run without high-performance personal computers or HPC infrastructure necessary to download and fine-tune larger open-source models. Moreover, using these infrastructures often requires technical knowledge such as familiarity with Python and relevant packages like PyTorch and TensorFlow, command line programming, and an understanding of hardware such as GPUs. Most social scientists do not have such training and thus may find it easier to adopt API-based solutions that require minimal programming to operate. With some basic scripting experience, it is relatively straightforward to use OpenAI's API. We expect the gap will lessen as more interactive solutions and more advanced open-source models emerge, but as things stand, proprietary API-based solutions are more user-friendly and accessible.

On the other hand, considerations related to transparency, reproducibility, and ethical and legal obligations may tip the scales in favor of open-source solutions (Spirling, 2023). LLMs vary with respect to transparency, both in terms of the model architecture and the data used for training. For example, we know what data were used to train BERT and how its internal embeddings work, but the data used to train GPT-4 and its exact architecture have not been disclosed. Proprietary models can also be less reproducible, implying that there is no guarantee that other researchers could obtain the same results in the future due to changes implemented to these models (Spirling, 2023). For example, OpenAI withdrew a code-generation model used in hundreds of computer

science publications from its API. Facing public criticism,[28] the company promised to continue to make the model available to researchers.[29] There are also ethical and legal considerations related to the use of closed-source, proprietary models (Spirling, 2023). Data input by users can be absorbed into the training data and used to improve these models, raising privacy concerns if sensitive data are input and could feasibly be reproduced. Scholars in Europe have noted that this would amount to data sharing that violates the European Union's General Data Protection Regulation (GDPR) regulations.[30] In light of these complaints, OpenAI recently announced that user inputs would not be used for further training, but it is unclear whether this practice will be adopted by other companies.[31]

Regardless of the choice of LLM, it is critical to consider how the composition of the training data can impact downstream applications. Bias in machine learning systems has been well-documented, as systems can learn stereotypical associations and patterns that are reproduced and reified. For example, an audit of computer vision models found models were less accurate at identifying the gender of darker-skinned female faces due to under-representation in the training data (Buolamwini and Gebru, 2018). Hate speech detection models can be biased against African-Americans, disproportionately flagging their speech as hateful, offensive, and abusive due to biased annotations and skewed training examples (Sap et al., 2019, Davidson et al., 2019). LLMs are no exception. Indeed, the fact that these models are trained using large swathes of the internet increases the risk of biases (Bender et al., 2021, Cheng et al., 2023). This could have serious implications for the quality of downstream tasks. For example, GPT-3 returned violent stereotypes casting Muslims as terrorists when completing innocuous prompts (Abid et al., 2021). In this case, it is not unreasonable to expect that the anti-Muslim bias might make classifiers less accurate at certain tasks, such as detecting Islamophobic hate speech. Recent work also documents how political biases learned to training data can affect downstream tasks like hate speech and misinformation detection (Feng et al., 2023).

Efforts to better document training data and models will make it easier for researchers to understand the strengths and weaknesses of particular models (Mitchell et al., 2019, Gebru et al., 2021), but such documentation provides little insight into how models generalize to more specific tasks of interest to social scientists. Various strategies have been used to address bias and other problematic behaviors in LLMs, from removing certain material from training data to training models to avoid specific types of output. While such efforts may make these models fairer and more reliable, they could also be detrimental to certain types of sociological inquiry, where biases and stereotypical associations are the objects of interest (Kozlowski et al., 2019). We encourage practitioners to

critically evaluate models to identify biases and scrutinize how they could affect performance on particular tasks. One straightforward approach to mitigating bias is to conduct careful audits, using templates to evaluate responses under different inputs (Röttger et al., 2021), and to compare multiple models to see how they fare on the same task.

Overall, we expect that LLMs will open up many new avenues for computational sociology. James Evans (2022, 1869) recently predicted that it would take a decade or more for the generative capacity of LLMs to be utilized by social scientists, but growing public interest and awareness of these technologies since the release of ChatGPT in late 2022 has accelerated their adoption for text analysis tasks. LLMs are increasingly becoming multimodal, trained on images and texts (Radford et al., 2021). Not only can multimodal models generate texts, but also images, enabling tools like DALL-E[32], MidJourney[33], and StableDiffusion[34] that can produce rich, detailed images from text-based prompts. Sociologists have only recently begun to explore the possibilities opened up by automated image classification (Zhang and Pan, 2019, Hwang et al., 2023). We expect that visual representations output by generative AI will serve as a rich resource for cultural sociology, extending earlier work on textual representations (Kozlowski et al., 2019, Stoltz and Taylor, 2021). Similar techniques can be extended to other modalities, including video and audio. While we focus on text-based LLMs, these tools will open up many opportunities for sociologically-relevant classification tasks. This article is a step towards understanding the strengths and weaknesses of different approaches to integrating large language models and related techniques for other modalities into sociological research.

Notes

[1]The original model was released in late 2018, and the paper has over 70,000 citations on Google Scholar as of July 2023.

[2]`https://www.washingtonpost.com/technology/interactive/2023/ai-chatbot-learning/`

[3]`https://blog.google/technology/ai/google-palm-2-ai-large-language-model/`

[4]LLMs can also be used similarly to earlier word embeddings, as new texts can be converted into LLM-specific vectorized representations, which can be input as features in standard classifiers (Rodriguez and Spirling, 2022, Bonikowski et al., 2022).

[5]While one can still conduct pre-processing before inputting texts, such as stopword removal or lowercasing, LLMs typically have customized tokenizers and internal vectorized representations, making this process largely redundant.

[6]`https://huggingface.co/bigscience/bloom`

[7]`https://bard.google.com/`

[8]https://www.anthropic.com/index/introducing-claude

[9]Stances inferred from social media are not necessarily equivalent to those obtained from surveys or that such approaches can replace traditional opinion polls. Comparisons between human annotation of social media posts and survey responses by the same users show variation in the correspondence between the two sources (Joseph et al., 2021) due to differences in measurement and temporal resolution. Thus, stance detection represents a reliable way of measuring the qualities of digital communications but is not a panacea for survey research in the digital age.

[10]Sentiment can also be expressed in domain-specific ways, so generic tools are not always reliable (Hamilton et al., 2016)

[11]The dataset and information can be found here: https://alt.qcri.org/semeval2016/task6/

[12]We do not distinguish between Neutral and None since our main interest is in whether an expression supports or opposes a candidate. An alternative approach is to use a two-stage classifier, first predicting whether or not a target is mentioned, then predicting the stance. However, this is more cumbersome and costly to implement as it requires training and evaluating multiple classifiers.

[13]See Wankmüller (2022) for a more detailed explication of BERT's architecture and Bonikowski et al. (2022) for further discussion.

[14]The model was trained for 8 epochs, and weights were optimized using the AdamW optimizer.

[15]The model can be downloaded here: https://huggingface.co/facebook/bart-large-mnli.

[16]A version of this model was adapted for conversations and released as ChatGPT in late 2022.

[17]https://platform.openai.com/docs/guides/fine-tuning/preparing-your-dataset

[18]This practice is advised by OpenAI. See example code provided for fine-tuning a classifier: https://github.com/openai/openai-cookbook/blob/main/examples/Fine-tuned_classification.ipynb

[19]OpenAI reports that "performance tends to linearly increase with every doubling of the number of examples." See https://platform.openai.com/docs/guides/fine-tuning/preparing-your-dataset

[20]This is not a problem for the BART-MNLI model as it is constrained to yield answers in the categories specified by the analyst.

[21]We truncate any long Facebook comments to avoid excessive token consumption. The mean is 272 characters, and SD is 604, so the maximum allowed length is 876 characters. A total of 5 examples out of 100 were truncated.

[22]We do not perform any cross-validation as it would prevent us from assessing how the size of the training data affects performance, and we are not optimizing any hyperparameters. Cross-validation is also costly when fine-tuning LLMs since k-fold cross-validation requires k separate fine-tuning and prediction runs.

[23]In practice, we never observe comments that favor both candidates, so there are only eight combinations in the training and test data.

[24]The one- and two-shot results are the average F1 scores across all prompts tested in subsection 6.2.

[25]Figure 2 shows stance prediction scores only, which are comparable to the ST task. Target-only F1 scores are listed in Table A6. Note how some several models score very highly, since targets can often be identified by simple keywords, although strong performance can be an artefact. The BERT

10 model achieves a perfect F1 score when predicting whether a tweet mentions Clinton because it predicts that Clinton is the target for *all* tweets, hence an F1 score of 0 in the adjacent column for Trump.

[26]https://platform.openai.com/docs/guides/fine-tuning/preparing-your-dataset

[27]https://www.anthropic.com/index/100k-context-windows

[28]See https://twitter.com/deliprao/status/1638014532680335363 and https://aisnakeoil.substack.com/p/openais-policies-hinder-reproducible

[29]https://twitter.com/OfficialLoganK/status/1638332437531971585

[30]https://twitter.com/LeonDerczynski/status/1611390172087652352

[31]https://techcrunch.com/2023/03/01/addressing-criticism-openai-will-no-longer-use-cus

[32]https://openai.com/product/dall-e-2

[33]https://www.midjourney.com/home

[34]https://github.com/Stability-AI/stablediffusion

References

Abid, A., Farooqi, M., and Zou, J. (2021). Persistent Anti-Muslim Bias in Large Language Models. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 298–306, Virtual Event USA. ACM.

Aldayel, A. and Magdy, W. (2021). Stance detection on social media: State of the art and trends. *Information Processing & Management*, 58(4):102597.

Allaway, E. and McKeown, K. (2020). Zero-Shot Stance Detection: A Dataset and Model using Generalized Topic Representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8913–8931, Online. Association for Computational Linguistics.

Argyle, L. P., Busby, E. C., Fulda, N., Rytting, C., and Wingate, D. (2023). Out of One, Many: Using Language Models to Simulate Human Samples. *Political Analysis*.

Augenstein, I., Rocktäschel, T., Vlachos, A., and Bontcheva, K. (2016). Stance Detection with Bidirectional Conditional Encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas. Association for Computational Linguistics.

Barberá, P., Boydstun, A. E., Linn, S., McMahon, R., and Nagler, J. (2020). Automated Text Classification of News Articles: A Practical Guide. *Political Analysis*, pages 1–24.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? In *FAccT '21: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, Canada. ACM.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137–1155.

Bestvater, S. E. and Monroe, B. L. (2022). Sentiment is Not Stance: Target-Aware Opinion Classification for Political Text Analysis. *Political Analysis*, pages 1–22.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J. Q., Demszky, D., Donahue, C., Doumbouya, M., Durmus, E., Ermon, S., Etchemendy, J., Ethayarajh, K., Fei-Fei, L., Finn, C., Gale, T., Gillespie, L., Goel, K.,

Goodman, N., Grossman, S., Guha, N., Hashimoto, T., Henderson, P., Hewitt, J., Ho, D. E., Hong, J., Hsu, K., Huang, J., Icard, T., Jain, S., Jurafsky, D., Kalluri, P., Karamcheti, S., Keeling, G., Khani, F., Khattab, O., Koh, P. W., Krass, M., Krishna, R., Kuditipudi, R., Kumar, A., Ladhak, F., Lee, M., Lee, T., Leskovec, J., Levent, I., Li, X. L., Li, X., Ma, T., Malik, A., Manning, C. D., Mirchandani, S., Mitchell, E., Munyikwa, Z., Nair, S., Narayan, A., Narayanan, D., Newman, B., Nie, A., Niebles, J. C., Nilforoshan, H., Nyarko, J., Ogut, G., Orr, L., Papadimitriou, I., Park, J. S., Piech, C., Portelance, E., Potts, C., Raghunathan, A., Reich, R., Ren, H., Rong, F., Roohani, Y., Ruiz, C., Ryan, J., Ré, C., Sadigh, D., Sagawa, S., Santhanam, K., Shih, A., Srinivasan, K., Tamkin, A., Taori, R., Thomas, A. W., Tramèr, F., Wang, R. E., Wang, W., Wu, B., Wu, J., Wu, Y., Xie, S. M., Yasunaga, M., You, J., Zaharia, M., Zhang, M., Zhang, T., Zhang, X., Zhang, Y., Zheng, L., Zhou, K., and Liang, P. (2022). On the Opportunities and Risks of Foundation Models. arXiv:2108.07258 [cs].

Bonikowski, B., Luo, Y., and Stuhler, O. (2022). Politics as Usual? Measuring Populism, Nationalism, and Authoritarianism in U.S. Presidential Campaigns (1952–2020) with Neural Language Models. *Sociological Methods & Research*, 51(4):1721–1787.

Brown, P. F., deSouza, P. V., Mercer, R. L., Della Pietra, V. J., and Lai, J. C. (1992). Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):14.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Buolamwini, J. and Gebru, T. (2018). Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In *Proceedings of Machine Learning Research*, volume 81, pages 1–15.

Burnham, M. (2023). Stance Detection With Supervised, Zero-Shot, and Few-Shot Applications. arXiv:2305.01723 [cs].

Caruana, R. (1997). Multitask Learning. *Machine Learning*, 28(1):41–75.

Cheng, M., Durmus, E., and Jurafsky, D. (2023). Marked Personas: Using Natural Language Prompts to Measure Stereotypes in Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 1504–1532.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. (2022). PaLM: Scaling Language Modeling with Pathways. arXiv:2204.02311 [cs].

Danescu-Niculescu-Mizil, C., West, R., Jurafsky, D., Leskovec, J., and Potts, C. (2013). No country for old members: User lifecycle and linguistic change in online communities. In *Proceedings of the 22nd international conference on World Wide Web*, pages 307–318. ACM.

Davidson, T., Bhattacharya, D., and Weber, I. (2019). Racial Bias in Hate Speech and Abusive Language Detection Datasets. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 25–35, Florence, Italy. ACL.

Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the 11th International Conference on Web and Social Media (ICWSM)*, pages 512–515.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186. ACL.

Do, S., Ollion, , and Shen, R. (2022). The Augmented Social Scientist: Using Sequential Transfer Learning to Annotate Millions of Texts with Human-Level Accuracy. *Sociological Methods & Research*, page 00491241221134526. Publisher: SAGE Publications Inc.

Evans, J. (2022). From Text Signals to Simulations: A Review and Complement to *Text as Data* by Grimmer, Roberts & Stewart (PUP 2022). *Sociological Methods & Research*, 51(4):1868–1885.

Evans, J. A. and Aceves, P. (2016). Machine Translation: Mining Text for Social Theory. *Annual Review of Sociology*, 42(1):21–50.

Felmlee, D., DellaPosta, D., Rodis, P. d. C. I., and Matthews, S. A. (2020). Can Social Media Anti-abuse Policies Work? A Quasi-experimental Study of Online Sexist and Racist Slurs. *Socius: Sociological Research for a Dynamic World*, 6:237802312094871.

Feng, S., Park, C. Y., Liu, Y., and Tsvetkov, Y. (2023). From Pretraining Data to Language Models to Downstream Tasks: Tracking the Trails of Political Biases Leading to Unfair NLP Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, volume Volume 1: Long Papers, pages 11737–11762.

Flores, R. D. (2017). Do anti-immigrant laws shape public sentiment? A study of Arizona's SB 1070 using Twitter data. *American Journal of Sociology*, 123(2):333–384.

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Iii, H. D., and Crawford, K. (2021). Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Grimmer, J., Roberts, M. E., and Stewart, B. M. (2022). *Text as data: A new framework for machine learning and the social sciences*. Princeton University Press.

Hamilton, W. L., Clark, K., Leskovec, J., and Jurafsky, D. (2016). Inducing Domain-Specific Sentiment Lexicons from Unlabeled Corpora. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 595–605. Association for Computational Linguistics.

Hanna, A. (2013). Computer-aided content analysis of digitally enabled movements. *Mobilization: An International Quarterly*, 18(4):367–388.

Hwang, J., Dahir, N., Sarukkai, M., and Wright, G. (2023). Curating Training Data for Reliable Large-Scale Visual Data Analysis: Lessons from Identifying Trash in Street View Imagery. *Sociological Methods & Research*, page 004912412311719.

Jensen, J. L., Karell, D., Tanigawa-Lau, C., Habash, N., Oudah, M., and Fairus Shofia Fani, D. (2022). Language Models in Sociological Research: An Application to Classifying Large Administrative Data and Measuring Religiosity. *Sociological Methodology*, 52(1):30–52.

Joseph, K., Shugars, S., Gallagher, R., Green, J., Quintana Mathé, A., An, Z., and Lazer, D. (2021). (Mis)alignment Between Stance Expressed in Social Media Data and Public Opinion Surveys. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 312–324, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joyce, K., Smith-Doerr, L., Alegria, S., Bell, S., Cruz, T., Hoffman, S. G., Noble, S. U., and Shestakofsky, B. (2021). Toward a Sociology of Artificial Intelligence: A Call for Research on Inequalities and Structural Change. *Socius*, 7:2378023121999581. Publisher: SAGE Publications.

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Kozlowski, A. C., Taddy, M., and Evans, J. A. (2019). The Geometry of Culture: Analyzing the Meanings of Class through Word Embeddings. *American Sociological Review*, page 0003122419877135.

Kreps, S., McCain, R. M., and Brundage, M. (2022). All the News That's Fit to Fabricate: AI-Generated Text as a Tool of Media Misinformation. *Journal of Experimental Political Science*, 9(1):104–117.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Le Mens, G., Kovács, B., Hannan, M., and Pros, G. (2023). Using Machine Learning to Uncover the Semantics of Concepts: How Well Do Typicality Measures Extracted from a BERT Text Classifier Match Human Judgments of Genre Typicality? *Sociological Science*, 10:82–117.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Liu, V. and Chilton, L. B. (2022). Design Guidelines for Prompt Engineering Text-to-Image Generative Models. In *CHI Conference on Human Factors in Computing Systems*, pages 1–23, New Orleans LA USA. ACM.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*. arXiv: 1907.11692.

Martin, J. H. and Jurafsky, D. (2009). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall, Upper Saddle River, NJ.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Miller, B., Linder, F., and Mebane, Jr., W. R. (2019). Active Learning Approaches for Labeling Text: Review and Assessment of the Performance of Active Learning Approaches. *Political Analysis*.

Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., and Gebru, T. (2019). Model Cards for Model Reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 220–229. arXiv:1810.03993 [cs].

Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., and Cherry, C. (2016). SemEval-2016 Task 6: Detecting Stance in Tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics.

Molina, M. and Garip, F. (2019). Machine Learning for Sociology. *Annual Review of Sociology*, 45:27–45.

Méndez, J. R., Iglesias, E. L., Fdez-Riverola, F., Díaz, F., and Corchado, J. M. (2006). Tokenising, Stemming and Stopword Removal on Anti-spam Filtering Domain. In Marín, R., Onaindía, E., Bugarín, A., and Santos, J., editors, *Current Topics in Artificial Intelligence*, Lecture Notes in Computer Science, pages 449–458, Berlin, Heidelberg. Springer.

Nelson, L. K. (2017). Computational Grounded Theory: A Methodological Framework. *Sociological Methods & Research*, page 004912411772970.

Nelson, L. K., Burk, D., Knudsen, M., and McCall, L. (2018). The Future of Coding: A Comparison of Hand-Coding and Three Types of Computer-Assisted Text Analysis Methods. *Sociological Methods & Research*, page 004912411876911.

OpenAI (2023). GPT-4 Technical Report. Technical report.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A.,

Welinder, P., Christiano, P., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs].

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., and Antiga, L. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. (2021). Learning Transferable Visual Models From Natural Language Supervision. *OpenAI*, page 47.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving Language Understanding by Generative Pre-Training.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683 [cs, stat].

Ren, C. and Bloemraad, I. (2022). New Methods and the Study of Vulnerable Groups: Using Machine Learning to Identify Immigrant-Oriented Nonprofit Organizations. *Socius: Sociological Research for a Dynamic World*, 8:237802312210769.

Rodriguez, P. L. and Spirling, A. (2022). Word Embeddings: What Works, What Doesn't, and How to Tell the Difference for Applied Research. *The Journal of Politics*, 84(1):101–115.

Röttger, P., Vidgen, B., Nguyen, D., Waseem, Z., Margetts, H., and Pierrehumbert, J. (2021). HateCheck: Functional Tests for Hate Speech Detection Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 41–58. ACL.

Sap, M., Card, D., Gabriel, S., Choi, Y., and Smith, N. A. (2019). The Risk of Racial Bias in Hate Speech Detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678. ACL.

Sen, I., Flöck, F., and Wagner, C. (2020). On the Reliability and Validity of Detecting Approval of Political Actors in Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1413–1426, Online. Association for Computational Linguistics.

Shor, E., Van De Rijt, A., Miltsov, A., Kulkarni, V., and Skiena, S. (2015). A Paper Ceiling: Explaining the Persistent Underrepresentation of Women in Printed News. *American Sociological Review*, 80(5):960–984.

Smith, N. A. (2020). Contextual word representations: putting words into computers. *Communications of the ACM*, 63(6):66–74.

Somasundaran, S. and Wiebe, J. (2010). Recognizing Stances in Ideological On-Line Debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 116–124.

Spirling, A. (2023). Why open-source generative AI models are an ethical way forward for science. *Nature*, 616(7957):413–413.

Stoltz, D. S. and Taylor, M. A. (2021). Cultural cartography with word embeddings. *Poetics*, page 101567.

Thorp, H. H. (2023). ChatGPT is fun, but not an author. *Science*, 379(6630):313–313. Publisher: American Association for the Advancement of Science.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971 [cs].

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, , and Polosukhin, I. (2017). Attention is All you Need. In *NIPS*, page 11, Long Beach, CA, USA.

Voyer, A., Kline, Z. D., Danton, M., and Volkova, T. (2022). From Strange to Normal: Computational Approaches to Examining Immigrant Incorporation Through Shifts in the Mainstream. *Sociological Methods & Research*, 51(4):1540–1579. Publisher: SAGE Publications Inc.

Wankmüller, S. (2022). Introduction to Neural Transfer Learning With Transformers for Social Science Text Analysis. *Sociological Methods & Research*, page 00491241221134527. Publisher: SAGE Publications Inc.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. (2022). Finetuned Language Models Are Zero-Shot Learners. arXiv:2109.01652 [cs].

Widmann, T. and Wich, M. (2022). Creating and Comparing Dictionary, Word Embedding, and Transformer-Based Models to Measure Discrete Emotions in German Political Text. *Political Analysis*, pages 1–16.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., and Funtowicz, M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Yin, W., Hay, J., and Roth, D. (2019). Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach. arXiv:1909.00161 [cs].

Zaller, J. R. (1992). *The nature and origins of mass opinion*. Cambridge University Press.

Zhang, H. and Pan, J. (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.

Zhou, D. (2022). The Elements of Cultural Power: Novelty, Emotion, Status, and Cultural Capital. *American Sociological Review*, page 000312242211230.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. (2020). Fine-Tuning Language Models from Human Preferences.

A Appendix

A.1 Stance distributions

	Trump	Clinton	Sum
Favor	148	163	311
Against	299	565	864
None	260	256	516
Sum	707	984	1691

Table A1: **Twitter: Stance distribution by target**

	Clinton:	Favor	Against	None	Sum
Trump:	Favor	0	110	469	579
	Against	46	15	177	238
	None	230	562	791	1583
	Sum	276	687	1437	2400

Table A2: **Facebook: Stance distribution across targets**

A.2 Baseline models

The first baseline model is a support vector machine (SVM) classifier using bag-of-words (BoW) features. The texts were preprocessed by performing tokenization, stopword removal, and stemming. The resulting tokens were then used to construct term-frequency inverse-document frequency (TF-IDF) weighted vectors representing each n-gram, where n ranges from 1-3. The purpose of TF-IDF weighting is to give higher weight to less frequent terms that should theoretically be better for discriminating between documents (Martin and Jurafsky, 2009). To reduce sparsity, we remove tokens that occur fewer than ten times in each training corpus, occur in more than 80% of documents, and rank outside the top 5000 most frequent tokens. We use the default SVM classifier from the Python package `scikit-learn` (Pedregosa et al., 2011).

To address the potential limitations of sparse BoW features, we implement a second baseline using embedding representations. We use pre-trained embeddings, specifically, GloVe (Global Vectors for Word Representation) embeddings trained on 2 billion tweets (Pennington et al., 2014). Pre-trained GloVe embeddings perform well on many NLP tasks (Rodriguez and Spirling, 2022), and these embeddings trained on Twitter data should better capture the specificities of language use on social media. We use the same pre-processing steps as above, with the exception of stemming, which would prevent words from being matched to the corresponding embedding. Each remaining word is represented as a 200-dimension real-valued vector, and documents are embedded in this vector

space by taking the average of the embeddings for each word. These document embeddings are then used as inputs in the same SVM classifier as above.

To assess whether more richly parameterized classifiers work better, we repeat baselines 1 and 2 using deep neural networks. Specifically, we train two convolutional neural networks (CNNs) using the Python package `PyTorch` (Paszke et al., 2019). Convolutional neural networks were initially proposed for image classification (Krizhevsky et al., 2012) but have demonstrated strong performance on text classification tasks (Kim, 2014). Each model contains 14 layers, including convolutional, MaxPooling, and fully-connected layers. The ReLU activation function is used to constrain weights to positive values. The data are passed through Dropout layers to reduce overfitting. These models with BoW and GloVe are trained for 10 and 150 epochs, respectively, in batches of 16 documents.

We estimate all four models for the MT prediction tasks on each dataset using all training data, thus providing a baseline to compare against the full fine-tuned models.

The results for the Twitter and Facebook tasks are reported at the bottom of Table A6 and Table A7, respectively. In general, we find that the models with BoW features outperform the embedding representations when evaluated using the strictest joint multitarget F1 score, despite the additional information encoded in the latter from pre-training. This suggests that the TF-IDF weighted n-gram representations better capture key features related to the stance detection task. Regarding the model selection, the results are relatively similar, but the SVM tends to perform slightly better across most tasks, particularly with BoW features.

A.3 Prompt examples

Prompt	Type	Tokens †
Twitter task		
Return the TARGET [Trump/Clinton] and STANCE [Favor/Against/None]. Answer: TARGET, STANCE	Minimal	29
This statement may express a STANCE about a TARGET. Return the TARGET [Trump/Clinton] and STANCE [Favor/Against/None]. Answer: TARGET, STANCE	Contextualized I	43
This statement contains a TARGET and a STANCE. The target is a politician and the stance represents the attitude expressed about them. The target options are Trump or Clinton and stance options are Favor, Against or None. Provide the answer in the following format: TARGET, STANCE	Contextualized II	60
Facebook task		
Return the STANCE [Favor/Against/None] for Trump and Clinton. Answer: Trump: STANCE, Clinton: STANCE	Minimal	30
This statement may express a STANCE towards Trump, Clinton, or both. Return the STANCE [Favor/Against/None] for Trump and Clinton. Answer: Trump: STANCE, Clinton: STANCE	Contextualized I	45
This statement may express a STANCE towards two politicians, Trump and Clinton. Stance represents the attitude expressed towards them. The stance options are Favor, Against or None. Provide the answer in the following format Trump: STANCE, Clinton: STANCE	Contextualized II	54

Table A3: **Prompt variations**
† Tokens needed for GPT-3 models calculated using OpenAI's tokenizer: `https://platform.openai.com/tokenizer`

A.4 Few-shot regression models

	C1	C2	C3	T1	T2	T3
wordcount	0.00	0.00	0.00	0.00	0.00	0.00
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
TrumpAgainst		-0.01	0.00		0.01	0.04
		(0.02)	(0.03)		(0.02)	(0.03)
TrumpFavor		0.02	0.04		0.01	0.03*
		(0.02)	(0.02)		(0.01)	(0.02)
ClintonAgainst		0.02	0.03		0.01	0.03
		(0.02)	(0.02)		(0.01)	(0.02)
ClintonFavor		0.03	0.05*		0.04*	0.07**
		(0.02)	(0.02)		(0.02)	(0.02)
TrumpAgainst × ClintonAgainst			0.03			0.04
			(0.08)			(0.06)
TrumpFavor × ClintonAgainst			-0.05			-0.08*
			(0.04)			(0.03)
TrumpAgainst × ClintonFavor			-0.08			-0.12*
			(0.06)			(0.05)
Intercept	0.71***	0.69***	0.68***	0.63***	0.61***	0.61***
	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)
R2	0.000	0.048	0.082	0.007	0.060	0.170
R2 Adj.	-0.010	-0.003	0.002	-0.004	0.009	0.097

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$. N=100.

Table A4: **Regression models predicting target-specific F1 score for Facebook one-shot MT task**

	C1	C2	C3	T1	T2	T3
wordcountTrump	0.00	0.00	0.00	0.00	0.00	0.00
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
wordcountClinton	0.00*	0.00*	0.00**	0.00	0.00	0.00
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
TrumpAgainst		0.00	0.02		-0.02**	-0.01
		(0.01)	(0.02)		(0.01)	(0.02)
TrumpFavor		0.00	0.01		-0.01	0.00
		(0.01)	(0.02)		(0.01)	(0.02)
ClintonAgainst		0.00	0.01		-0.01	0.00
		(0.01)	(0.02)		(0.01)	(0.02)
ClintonFavor		0.00	0.02		0.01	0.04
		(0.01)	(0.02)		(0.01)	(0.02)
TrumpAgainst × ClintonAgainst			-0.02			-0.01
			(0.02)			(0.02)
TrumpFavor × ClintonAgainst			0.00			0.01
			(0.02)			(0.03)
TrumpAgainst × ClintonFavor			-0.03			-0.03
			(0.03)			(0.03)
TrumpFavor × ClintonFavor			-0.05			-0.03
			(0.03)			(0.03)
Intercept	0.63***	0.63***	0.62***	0.40***	0.42***	0.41***
	(0.02)	(0.02)	(0.02)	(0.02)	(0.02)	(0.02)
R2	0.072	0.079	0.133	0.018	0.143	0.174
R2 Adj.	0.053	0.019	0.035	-0.002	0.087	0.081

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$. N=100.

Table A5: **Regression models predicting target-specific F1 score for Twitter two-shot MT task**

A.5 Full results

Model	Type	T_{ST_S}	T_{MT_S}	C_{ST_S}	C_{MT_S}	T_{MT_T}	C_{MT_T}	O_{ST_S}	O_{MT_S}	O_{MT_T}	J_{MT}
GPT-3	0	0.62	0.70	0.64	0.70	0.76	0.80	0.63	0.70	0.78	0.49
Davinci	1	0.62	0.59	0.73	0.70	0.76	0.80	0.68	0.65	0.78	0.51
	10	0.36	0.31	0.45	0.49	0.83	0.89	0.42	0.42	0.86	0.30
	100	0.61	0.68	0.72	0.69	0.88	0.94	0.67	0.69	0.91	0.61
	All	**0.76**	**0.75**	**0.81**	**0.76**	**0.95**	0.93	**0.79**	**0.76**	**0.94**	**0.69**
GPT-3	0	0.06	0.03	0.12	0.02	0.11	0.00	0.09	0.02	0.04	0.00
Ada	1	0.24	0.11	0.23	0.19	0.36	0.03	0.23	0.15	0.17	0.04
	10	0.47	0.22	0.44	0.54	0.95	0.84	0.45	0.40	0.89	0.30
	100	0.59	0.54	0.55	0.59	0.83	0.94	0.57	0.57	0.90	0.46
	All	0.72	0.71	0.75	0.74	0.92	0.95	0.74	0.73	**0.94**	0.65
BART-MNLI	0	0.33	0.36	0.68	0.69	**0.95**	0.90	0.54	0.55	0.92	0.48
	1	0.38	0.38	0.65	0.60	0.91	0.88	0.54	0.51	0.89	0.45
	10	0.38	0.38	0.62	0.65	0.90	0.89	0.52	0.54	0.89	0.46
	100	0.40	0.40	0.68	0.62	0.90	0.87	0.56	0.53	0.88	0.46
	All	0.49	0.56	0.67	0.66	0.89	0.87	0.60	0.62	0.88	0.51
BERT	0	-	-	-	-	-	-	-	-	-	-
	1	-	-	-	-	-	-	-	-	-	-
	10	0.23	0.00	0.44	0.44	0.00	**1.00**	0.35	0.26	0.58	0.18
	100	0.44	0.35	0.47	0.48	0.32	0.90	0.46	0.43	0.66	0.23
	All	0.65	0.67	0.68	0.75	0.88	0.92	0.67	0.72	0.90	0.60
SVM w/ BoW	All	-	0.53	-	0.57	0.80	0.95	-	0.55	0.89	0.48
SVM w/ GloVe	All	-	0.49	-	0.47	0.70	0.92	-	0.47	0.83	0.34
CNN w/ BoW	All	-	0.61	-	0.58	0.94	0.80	-	0.59	0.86	0.47
CNN w/ GloVe	All	-	0.46	-	0.52	0.88	0.84	-	0.49	0.86	0.40

Table A6: **Twitter: Single and Multitarget Predictions for Trump and Clinton.**
Best model for each metric shown in bold. Models on two examples for multitarget (MT) predictions.
C = Clinton, T = Trump, O = Overall, J = Joint.

Model	Type	T_{ST}	T_{MT}	C_{ST}	C_{MT}	O_{ST}	O_{MT}	J_{MT}
GPT-3	0	0.69	0.77	0.72	0.72	0.71	0.75	0.61
Davinci	1	0.53	0.63	0.60	0.71	0.56	0.67	0.49
	10	0.47	0.51	0.48	0.45	0.47	0.48	0.15
	100	0.86	0.84	0.84	0.80	0.85	0.82	0.67
	1000	**0.90**	0.89	**0.90**	0.88	**0.90**	0.88	0.80
	All	0.88	**0.90**	**0.90**	**0.91**	0.89	**0.90**	**0.83**
GPT-3	0	0.10	0.00	0.03	0.00	0.06	0.00	0.00
Ada	1	0.26	0.00	0.26	0.00	0.26	0.00	0.00
	10	0.53	0.52	0.44	0.46	0.48	0.49	0.18
	100	0.78	0.73	0.75	0.71	0.77	0.72	0.55
	1000	0.79	0.82	0.85	0.84	0.82	0.83	0.71
	All	0.85	0.82	0.85	0.86	0.85	0.84	0.73
BART	0	0.36	0.49	0.34	0.60	0.35	0.55	0.35
-MNLI	1	0.37	0.51	0.32	0.52	0.35	0.52	0.28
	10	0.42	0.50	0.38	0.53	0.40	0.51	0.29
	100	0.49	0.51	0.49	0.55	0.49	0.53	0.28
	1000	0.72	0.63	0.64	0.64	0.68	0.64	0.41
	All	0.68	0.64	0.62	0.61	0.65	0.62	0.37
BERT	0	-	-	-	-	-	-	-
	1	-	-	-	-	-	-	-
	10	0.26	0.52	0.44	0.06	0.35	0.29	0.03
	100	0.53	0.52	0.64	0.52	0.58	0.52	0.22
	1000	0.81	0.81	0.82	0.81	0.82	0.81	0.68
	All	0.82	0.82	0.83	0.83	0.82	0.83	0.70
SVM w/ BoW	All	-	0.76	-	0.71	-	0.73	0.54
SVM w/ GloVe	All	-	0.64	-	0.60	-	0.62	0.39
CNN w/ BoW	All	-	0.70	-	0.68	-	0.69	0.49
CNN w/ GloVe	All	-	0.62	-	0.61	-	0.61	0.39

Table A7: **Facebook: Single and Multitarget Predictions for Trump and Clinton.** Best model for each metric shown in bold. C = Clinton, T = Trump, O = Overall, J = Joint.