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

July 31, 2024

ABSTRACT

Advances in large language models (LLMs) have transformed the field of natural language processing and have enormous potential for social scientific analysis. We explore the application of LLMs to supervised text classification. As a case study, we consider stance detection and examine variation in predictive accuracy across different architectures, training regimes, and task specifications. We compare ten models ranging in size from 86 million to 1.7 trillion parameters and four distinct training regimes: prompt-based zero-shot learning; few-shot learning; fine-tuning; and instruction-tuning. The largest models generally offer the best predictive performance, but fine-tuning smaller models is a competitive solution due to their relatively high accuracy and low cost. For complex prediction tasks, instruction-tuned open-weights models can perform well, rivaling state-of-the-art commercial models. We provide recommendations for the use of LLMs for text classification in sociological research and discuss the limitations and challenges related to the use of these technologies.

Keywords large language models · natural language processing · stance detection · computational social science

Acknowledgments. We are grateful to members of the American Sociological Association Collective Behavior and Social Movements Section Small-Group on Computational Methods and attendees at the ASA session on Text as Data at the 2023 Annual Meeting in Philadelphia for helpful comments and suggestions and Marina Rivera Ramos for research assistance. 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, and OpenAI for providing GPT-4 credits through the Researcher Access Program.

Youngjin Chae is a PhD student in the Department of Sociology at Rutgers University. He studies family instability, parenthood politics, and population health using causal inference and natural language processing techniques.

Thomas Davidson is an Assistant Professor in the Department of Sociology at Rutgers University. He studies online activism, far-right politics, and hate speech using a range of computational techniques.

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

1 Introduction

Interest in the topic of artificial intelligence (AI) has exploded since the release of OpenAI's ChatGPT in late 2022. Advances in language modeling, notably the development of large language models (LLMs) has resulted in significant breakthroughs in many areas of natural language processing (NLP). These developments represent a paradigm shift in machine learning, as these systems have evolved from being "narrow specialists" designed for single tasks to "competent generalists" that can perform many new tasks without any additional training (Radford et al., 2019). The text-to-text interface makes it relatively simple to interact with these technologies (Raffel et al., 2020) and further adaptations to improve their capability at following instructions (Wei et al., 2022) enable LLMs to serve as a foundation for many downstream tasks (Bommasani et al., 2022). LLMs and other generative AI thus promise to open up new opportunities in many areas of social scientific research, from text analysis and agent-based modeling to literature review and hypothesis generation (Grossmann et al., 2023; Bail, 2024; Davidson, 2024). In this paper, we evaluate the use of LLMs as a methodology for supervised text classification, building on work showing that even relatively small models can achieve impressive performance with limited training (Do et al., 2022; Wankmüller, 2022; Bonikowski et al., 2022; Ziems et al., 2024). We provide an overview of LLMs and recent developments in NLP and machine learning, evaluate how the predictive accuracy of LLMs compares to conventional techniques and varies across different models and learning regimes, and demonstrate how the generative capabilities of these models can advance our abilities to analyze and classify complex, structured data. We use the results from these analyses to provide practical recommendations for the usage of these techniques in sociological research.

As a case study, we use LLMs to detect stances expressed in social media posts (Somasundaran and Wiebe, 2010; Sobhani et al., 2015; Mohammad et al., 2017). Stance detection represents an important application of supervised text classification that can be used to measure attitudes, opinions, and beliefs in observational texts that sociologists have previously analyzed using sentiment analysis (Shor et al., 2015; Flores, 2017; Felmlee et al., 2020). We use three different stance detection datasets related to the 2016 US presidential election, consisting of a popular benchmark dataset from Twitter (Mohammad et al., 2016) and two novel sets of Facebook data. In each case, we use LLMs to predict the stances expressed towards the two leading candidates in the election. These analyses allow us to triangulate the performance of LLMs across related tasks and to illustrate multiple approaches to text classification.

We begin by evaluating three different approaches, or learning regimes, that can be adopted when using LLMs for text classification. First, leveraging the capacity of models to generalize based on large amounts of text consumed during pre-training, the most advanced LLMs can perform prompt-based zero-shot learning without any additional training (Radford et al., 2019). We explore how some LLMs can detect stances based on instructions alone and assess how this performance varies depending on the length and level of detail provided in the prompt. Second, we consider whether labeled exemplars input alongside prompts result in improved performance (Brown et al., 2020) and conduct experiments to assess the sensitivity to particular exemplars. Finally, LLMs can be fine-tuned using larger sets of annotated training data, an approach more akin to conventional machine learning. We conduct a series of evaluations to assess how predictive performance varies based on the model and the magnitude of the training data. Across these experiments, we test ten different LLMs, ranging from relatively small models like BERT (Devlin et al., 2019) to OpenAI's state-of-the-art GPT-4o (OpenAI, 2023). As baselines, the results of these analyses are compared to random guessing and conventional machine learning classifiers trained directly on each set of texts.

To illustrate the capacity of LLMs to perform more difficult prediction tasks and handle complex, structured data, we formulate a novel approach to analyzing comment-reply threads. This analysis uses a fourth training regime known as instruction-tuning, where a model is fine-tuned with instructions and pairs of inputs and outputs (Wei et al., 2022), bringing together the strengths of prompting and fine-tuning. The data were annotated taking the full conversational context into account and the classification task involves not only predicting the stance of a comment but also the stances of any associated replies. This is a more challenging prediction problem, as there are multiple different predictions and the number of predictions required varies across threads depending on the number of replies. We show how the combination of structured data formats and detailed prompts can be used to perform this complex classification task using zero-shot learning then use instruction-tuning to further adapt models to perform this task. This analysis demonstrates how LLMs can be used to study opinion dynamics in more realistic, conversational contexts and outlines a framework for using LLMs on other tasks involving more complex data and coding schemes.

With sufficient training, our results show that most models evaluated outperform conventional machine learning approaches, but there is substantial variability in performance across architectures and learning regimes. Zero-shot and few-shot learning only work well when using the largest, most powerful models, and predictive performance can be sensitive to the composition of the prompt and

exemplars provided. The largest model evaluated, GPT-4o, can outperform fine-tuned models at predicting the stance of Facebook comments without any task-specific training. When fine-tuning models using more training data, the largest models also achieve the best performance, but far smaller models can perform almost as accurately as their larger counterparts when fine-tuned on larger sets of annotated training data. The results from our thread-prediction task show how LLMs trained to predict multiple stances in comment-reply threads can achieve high levels of accuracy, and the conversational context results in substantial improvement compared to models that make predictions using each text alone. Moreover, we find that cutting-edge open-weights models are becoming considerably more powerful, as Llama3-70B shows comparable zero-shot performance to GPT-4o and is significantly more accurate after instruction-tuning.

Based on these results, we develop a series of recommendations for sociologists interested in using LLMs for text classification. We discuss how the choice of model will vary depending on the nature of the task, the number of documents to be classified, the amount of annotated data available, and available computing resources. We also consider the limitations and challenges related to the use of these models, discussing issues including interpretability, reliability, bias, reproducibility, and privacy, both as they pertain to our analyses and other social scientific applications, and discuss the trade-offs between proprietary models and open-weights alternatives in the context of these considerations. Overall, we anticipate that these experiments and recommendations will serve as a valuable resource for sociologists interested in deploying LLMs for text classification and other related tasks.

2 Background

2.1 Language modeling and transfer learning

We begin with an overview of language modeling before turning to the most recent advances in the field. Language modeling involves the development of computational models that learn probabilistic representations of language. The main objective of a language model is to predict the next word or set of words in a sequence, although there are other variations of this basic task. For example, what is the most likely word to end the sentence: "The cat caught a ..."? Each 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." In a *bi-gram* model, the previous word in a sequence is used to predict the next word. More formally, a bi-gram language

model is used to obtain $P(w_i|w_{i-1})$, the probability that word w_i is used given the previous word w_{i-1}. Given a corpus of text, these probabilities can be calculated by counting the number of times a given pair of words co-occurs sequentially and normalizing by the frequency of w_{i-1} (Martin and Jurafsky, 2024). *N-gram* language models extend this logic by using the previous n words in a sequence to predict the next word. Considering the example above, we should expect to make more accurate predictions if the model has more context (e.g., "cat caught a" versus "caught a" or "a"). The origins of language modeling date back to the early 20th century, when Andrey Markov used transition probabilities to model character sequences in literature, and the mathematical properties of these models were further developed by Claude Shannon (1948), who drew on statistical mechanics to study uncertainty in n-gram models (see Martin and Jurafsky, 2024; Li, 2022; Manning, 2022 for further historical background). By the 1990s, computer scientists had developed techniques to compute probabilistic language models from larger corpora of text with applications to tasks including speech recognition, translation, and spelling correction (Bahl et al., 1983; Brown et al., 1990, 1992). However, these models become intractable as n increases and are sensitive to the composition of the training corpus, making them difficult to generalize to new data. Despite these limitations, n-gram language models can 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 these approaches difficult to operationalize beyond relatively simple tasks. By the early 2010s, the availability of large corpora of online text and the development of neural networks optimized to run on graphics processing units (GPUs) — computer chips originally designed for rendering graphics in video games — made it possible to train more powerful language models (Manning, 2022). Notably, Mikolov et al. (2013a,b) proposed the Word2vec architecture, using 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 represent words as dense numeric vectors and have opened up new possibilities for the sociological study of language and culture (Kozlowski et al., 2019; Stoltz and Taylor, 2021; Rodriguez and Spirling, 2022).

Neural language models can be adapted to perform other types of natural language processing tasks (Dai and Le, 2015; Howard and Ruder, 2018; Radford et al., 2018; Devlin et al., 2019), through a process known as *transfer learning*. First, a neural language model is *pre-trained* using variants of

the next-word prediction task, enabling the model to "learn" a general representation of language. Second, the pre-trained model can be modified to enable it to perform a new task, such as translation or question-answering, using a process known as *fine-tuning*. This works similarly to conventional supervised machine learning techniques. The main difference is that, rather than learning parameters from scratch, the existing weights in the pre-trained model are updated as a model continues to train on a new dataset. The technique was pioneered in computer vision, as models trained on diverse sets of images could be adapted to detect new, previously unseen imagery by making use of the generic representations they have already learned (Yosinski et al., 2014) (see Zhang and Pan, 2019 for a sociological application). Transfer learning represents an important breakthrough in machine learning insofar as it alleviates the need to train a custom model for every new task.

2.2 Large language models

The latest generation of these neural language models are known as *large* language models, or LLMs, due to their size and the amount of text they are trained on. This label encompasses a wide variety of models, which we discuss in further detail below. These models have recently grown in size following evidence of "scaling laws," as they show improvement in performance in both language modeling and downstream tasks as they get larger (Kaplan et al., 2020; Brown et al., 2020; Raffel et al., 2020). Take, for example, the series of GPT models from OpenAI: The original GPT (Radford et al., 2018), released in 2018, had 117 million parameters and was trained on BookCorpus, a collection of 7,000 books; a year later, GPT-2 was released, with 1.5 billion parameters, trained on over 8 million websites (Radford et al., 2019); its successor GPT-3 – a version of which was released to the public as ChatGPT – was scaled up to 175 billion parameters and was trained on much of the open internet, including the Common Crawl dataset, digitized books, and all of Wikipedia (Brown et al., 2020); the exact details of the latest model, GPT-4, are have not been disclosed (OpenAI, 2023), but it is trained on texts, images, and audio and is rumored to contain 1.7 trillion parameters. The largest models excel at transfer learning, and the capacity to generalize well to many new tasks has led them to be termed "foundation models" (Bommasani et al., 2022). For example, GPT-4 not only achieves strong performance on a variety of machine learning benchmarks, but outscored the majority of people on tests including the Uniform Bar Exam, Advanced Placement Biology, and the Verbal GRE (OpenAI, 2023). In parallel with these scaling efforts, developers have also created smaller LLMs, like Meta's Llama, Google's Gemma, and Mistral AI's eponymous models that retain some of the key features of their larger cousins.

While neural language models can have a variety of different formats (e.g. (Mikolov et al., 2013a; Howard and Ruder, 2018; Jiang et al., 2024)), the most popular approaches are based on an architecture known as the *Transformer*. Proposed by Vaswani et al. (2017), the transformer consists of two key components, an *encoder* and a *decoder*, which were used in earlier work in machine translation (Cho et al., 2014). The encoder takes in an input sequence of text tokens $x = x_1, ..., x_n$, such as text in the original language, and converts them into a sequence of continuous values, or embeddings, $h = h_1, ..., h_n$. The main innovation is the addition of an *self-attention* mechanism, which parses text in both directions, left-to-right and right-to-left. This enables the model to compute a weight for each token in the sequence based on its relevance to every other token. Martin and Jurafsky (2024, 214) describe the self-attention mechanism as "a way to build contextual representations of a word's meaning that integrate information from surrounding words, helping the model learn how words relate to each other over large spans of text." The decoder then uses the continuous representation from the encoder to generate an output text. It works by mapping the embeddings from the encoder onto a conditional probability distribution of the next tokens and autogressively generating the output tokens one by one, from left-to-right. At each step, the previously generated tokens are recursively input into the decoder, providing context to predict the next token. Through this process, the model outputs a sequence of text tokens $y = (y_1, ..., y_m)$, such as a translation into another language. In practice, the transformer consists of a network containing a stack of encoder and decoder blocks that repeat this process several times before generating the final output (See Wankmüller, 2022 and Martin and Jurafsky, 2024 for further detail). Critically, when applied to language modeling, the self-attention mechanism in the transformer improves efficiency and enables models to capture long-range dependencies in texts, effectively solving the tractability issues that constrained n-gram language models.

There are many different variations on the transformer. Broadly, these architectures can be grouped into three categories, but all three make use of the attention mechanism. Encoder-only models such as BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) — one of the most widely used transformer models[1] — use only the encoder portion of the transformer. BERT is pre-trained using "masked" language modeling, where 15% of the input tokens are hidden, or masked, and must be predicted by the model. It uses bi-direction attention, parsing inputs from left-to-right and right-to-left, using the surrounding context to predict the masked tokens.[2] This makes the model particularly suited for learning embeddings containing rich semantic information that can be helpful for tasks like sentiment analysis (Smith, 2020). However, these models are insufficient

for text generation since they lack the generative decoder module. Decoder-only models, on the other hand, do not use bi-directional attention. Instead, models like those in OpenAI's GPT family, use attention to parse the input from left-to-right, producing each output token autoregressively, making them optimized for generating texts (Radford et al., 2018). Decoders are sometimes referred to as "causal" language models because they use masking so the model cannot "see into the future" (Raffel et al., 2020, 17). This prevents a model from attending to later tokens in the input sequence, $w_{i+1} : w_m$, when generating the next token y_i corresponding to input token w_i. Finally, encoder-decoder systems such as Meta's BART and Google's Flan-T5 retain a structure that more closely resembles the original transformer (Lewis et al., 2019; Raffel et al., 2020), based on the assumption that both modules are beneficial for various downstream tasks. Recent comparisons show that decoder-only models have achieved the strongest performance across many benchmarks and are preferred due to a comparatively easier pre-training process and smaller model size (Fu et al., 2023). In our analysis, we compare models with these different architectures to assess how they differ in performance on text classification tasks.

2.2.1 From zero-shot learning to instruction-tuning

Perhaps the most notable feature of the most recent generation of LLMs is the capacity to interface with models using text. Almost any natural language processing problem can be posed by providing a text as an input and generating a new text as an output (Radford et al., 2019; Raffel et al., 2020). For example, a model might take a sentence as input and yield a translation as its output. The text-to-text interface represents a fundamental shift from classical approaches to natural language processing, where the analyst first has to convert texts into numeric vectors, often referred to as "features," which are then input into a model. Typically, decisions must be made about how to "tokenize" the input, splitting the text into words, sets of words, or smaller units like sequences of characters. These units are then used to construct vector representations, which can vary from simple counts to embeddings derived from language models, depending on the application. Other features, such as non-textual information like follower counts on social media, could also be appended to the vector representation. Depending on the task, machine learning algorithms are then used to learn weights that map features onto outcomes, or, in unsupervised learning, to inductively summarize the data (Martin and Jurafsky, 2024; Evans and Aceves, 2016).[3] In contrast, the raw texts can be directly input to text-to-text models, and tokenization and vectorization happen automatically, as the inputs are converted into numeric representations compatible with the particular architecture.[4]

Other contextual information can also be provided alongside the texts, as we demonstrate in our final set of analyses. The decoder module then autoregressively generates an output sequence based on the conditional probability distribution of next words given the input.

To control the way models convert inputs into outputs, written instructions known as *prompts* can be included with the inputs. Extending the potential of transfer learning, this affordance presents a more radical possibility: pre-trained LLMs can be applied to new tasks without any additional task-specific training, known as *zero-shot learning* (Radford et al., 2019). For example, one could provide a newspaper article and ask the model to indicate whether it contains a certain kind of framing. The model could then output a reasonable answer, despite never having been trained to perform the task. Unlike earlier approaches to transfer learning, these techniques do not require that a model's weights be updated using training data and a single model can be prompted to perform many different tasks using instructions alone. A nascent field known as *prompt engineering* explores the effects of different kinds of prompts. There is evidence that performance can vary substantially as a result of variations in the prompt. For example, asking a model to provide a short explanation for its reasoning, known as *chain-of-thought* prompting, can result in better accuracy (Wei et al., 2023a) and simply including the sentence "Let's think step by step" at the end of the prompt can improve zero-shot performance (Kojima et al., 2024). One can also provide models with more guidance by including training examples along with the prompt, known as *few-shot* or *in-context* learning (Brown et al., 2020). For example, one could show a sentence and a translation before asking for another sentence to be translated. The additional information contained in the exemplar can help to improve the quality of the output, particularly for difficult tasks where a prompt alone is insufficient. When performing in-context learning, models use both the semantic information from the prompt and labels and the input-label mappings from the exemplars to make predictions (Wei et al., 2023b). Taken together, prompt-based learning regimes represent a paradigm shift in the way that we interact with language models, opening up new possibilities for social scientific inquiry.

A shortcoming of prompting is that there is no guarantee that models will follow instructions or generate the desired outputs. In some sense, it is happenstance that the most likely tokens generated by the model often correspond to the outputs we are interested in. To enhance the capacity of LLMs to follow instructions, pre-trained models can be fine-tuned using large quantities of instruction and output pairs (Wei et al., 2022), known as *instruction-tuning*. This technique leads to improved performance for zero-shot and few-shot learning tasks and better quality text generations. Instruction-tuning the largest models can result in additional performance gains (Chung et al., 2022),

further demonstrating the importance of scaling. LLMs can be adapted to automatically improve their capacity to follow instructions via a process known as reinforcement learning with human feedback (RLHF) (Ziegler et al., 2020), which is central to the conversational capabilities of chatbots like ChatGPT. To perform RLHF, human raters begin by picking the best responses to queries from a set of possible candidates and this information is used to train the model to optimize its generations to be more consistent with human preferences (Ouyang et al., 2022). Instruction-tuning can also be used to make models more efficient by using outputs from larger "teacher" models as instructions to tune smaller models (Wang et al., 2023). For example, Taori et al. (2023) used 52K outputs from a GPT-3 model with 175B parameters as instructions to fine-tune Llama 7B. Their model, dubbed Alpaca, generates texts that are qualitatively similar to those from GPT-3. LLMs that have undergone these additional instruction-tuning techniques can serve as a strong foundation for subsequent tasks (Bommasani et al., 2022), and, as we explore below, further instruction-tuning can be used to adapt them to social scientific tasks (Wei et al., 2022).

2.2.2 New methodological challenges

LLMs have created many new opportunities for methodological advancement, but these technological developments also raise several new challenges. Due to the cost of training LLMs — which can run into hundreds of millions of dollars considering hardware and electricity costs (Strubell et al., 2019) — researchers increasingly rely upon models trained by third parties. Some of these models, like BLOOM,[5] are *open-source*, meaning that all training data, code, and the models are publicly available. Due to the resource investments needed to train these models, however, most have been developed by a handful of technology companies. Rather than fully open-sourcing their models, companies like Meta, Google, and Mistral have released models that anyone can use while keeping other details such as the code and training data private, known as *open-weights* models. Other models, particularly the largest state-of-the-art LLMs, are only accessible using paid application programming interfaces (APIs), and the use of these closed models in academic research raises concerns about transparency, reproducibility, and privacy (Spirling, 2023; Palmer et al., 2024). Others have raised concerns that these models will reproduce stereotypes and biases due to the training on vast amounts of unvetted data (Bender et al., 2021; Weidinger et al., 2022). Several recent papers consider how these factors affect the use of LLMs in social scientific research (Grossmann et al., 2023; Bail, 2024; Davidson, 2024). At the end of the paper, we return to these

issues as they pertain to our analysis and provide guidance to aid researchers in addressing these challenges and limitations when using LLMs.

2.3 Text classification using large language models

Unlike other machine learning algorithms, language models are not strictly supervised or unsupervised and can be used as a foundation for many downstream applications (Bommasani et al., 2022). We focus on the use of LLMs for text classification, an application of supervised machine learning used to categorize texts into pre-determined classes. This task differs from 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 workplace feedback (Nelson et al., 2023). The capacity to perform new tasks with little or no additional training makes LLMs particularly promising for empirical research (Do et al., 2022). The use of LLMs lowers the bar for entry by reducing the costs associated with data annotation which makes supervised text classification a less viable option for many social scientists. Several studies show how transformer-based language models outperform conventional machine learning algorithms at various classification tasks including the detection of emotional language (Widmann and Wich, 2022), nationalist, populist, and authoritarian rhetoric (Bonikowski et al., 2022), and discussion of policy in news articles (Do et al., 2022).

Most existing work fine-tunes encoder-only models, but some recent scholarship applies zero- and few-shot learning, building upon studies by AI labs that demonstrate strong performance across a range of NLP tasks (e.g. Radford et al., 2019; Brown et al., 2020; Raffel et al., 2020). These studies have mixed findings regarding the efficacy of the latest innovations in LLMs. Wankmüller (2022) conducts experiments using variants of the BERT models, which can be adapted to perform rudimentary zero-shot learning, but finds they fare poorly compared to fine-tuned models when used to predict political sentiments and toxicity. An analysis across 15 psychological constructs found that zero-shot GPT-3 and GPT-4 often performed equally or better than fine-tuned models and the GPT-4 performed particularly well across multiple languages (Rathje et al., 2023). The most comprehensive analysis compares the performance of zero-shot FLAN and GPT models with

a baseline encoder model, RoBERTa, which shares the BERT architecture but is trained on more data (Ziems et al., 2024). Across 20 different prediction tasks including emotion, misinformation, and ideology detection, they show that the larger generative models rarely outperform fine-tuned RoBERTa. Models with more parameters tend to perform better than smaller variants, consistent with other work (Raffel et al., 2020; Brown et al., 2020), but there is variation across architectures, with encoder-decoder FLAN models achieving the highest score on 9 tasks and the larger decoder-only GPT models performing best on the remainder. They also conduct few-shot learning using the FLAN models but see little evidence of improvement. Beyond size and architecture, the authors also point to the importance of subsequent training, showing that the variant of GPT-3 that had undergone RLHF outperformed the pre-trained version by an average of 3.5 points (based on F1 scores described below). Overall, this study finds that fine-tuning may still be a superior strategy to zero- and few-shot learning.

We build upon this work in several ways. To better understand how the learning regime impacts performance, we compare four different approaches to prediction, ranging from zero-shot and few-shot learning to fine-tuning and instruction-tuning. Across these different learning regimes, we evaluate how choices including variation in prompts, training examples, and the size of the training data affect predictive accuracy. We also consider a broader range of models, varying in size, architecture, and openness. Our goal is to consider how these different choices impact predictive accuracy to identify the most effective approaches and to illuminate the trade-offs researchers using these models will face.

2.4 Stance detection in social media posts

We use stance detection as a case study to evaluate the performance of LLMs for supervised text classification. The goal of stance detection is to identify the attitudes, beliefs, or opinions expressed toward a target, such as a person, institution, or policy. The methodology was developed by computer scientists interested in studying opinions expressed in online debates (Somasundaran and Wiebe, 2010; Sobhani et al., 2015; Mohammad et al., 2017) and has become a popular approach for studying social media discourse (see Aldayel and Magdy, 2021 and Küçük and Can, 2020 for reviews). Prior work has demonstrated that neural network architectures can achieve strong performance on stance detection benchmarks (Augenstein et al., 2016) and recent research studies that LLMs achieve reasonable accuracy in zero- and few-shot settings (Allaway and McKeown, 2020; Burnham, 2023; Ziems et al., 2024). We build upon these studies by considering how variation across models and

learning regimes impact stance detection and by considering how the generative capabilities of LLMs enable more sophisticated approaches to measuring stance in online discussions.

To perform stance detection, documents are annotated with labels like "Support/Oppose" or "Favor/Against," typically with a category, such as "Neither" or "Neutral" that captures ambivalent or irrelevant texts (Somasundaran and Wiebe, 2010; Mohammad et al., 2016). Often, stance detection also involves the identification of a target towards which the stance is directed, known as "multi-target" stance detection (Sobhani et al., 2017). 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") and has gained popularity among social scientists. Following Bestvater and Monroe (2022), we contend stance detection would be a more suitable measurement strategy in many sociological applications of sentiment analysis, where sentiment is used as a proxy for attitudes towards groups like 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) (Mohammad et al., 2017), 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." Using sentiment as a proxy for stance can result in substantial measurement error, particularly when the correlation between sentiment and stance is weak.[6] We searched seven major generalist or methodology journals in sociology and found twenty mentions of sentiment analysis and several applications of the technique, but no mention of stance detection.[7] We suspect the neglect of stance detection in sociology stems from the fact that it has historically been difficult to implement because it requires the development of custom, domain-specific classifiers trained on relevant annotated corpora (Sen et al., 2020; Bestvater and Monroe, 2022), whereas sentiment analysis can be performed using off-the-shelf lexicons or classifiers.[8] We anticipate that stance detection will be a valuable methodology in many areas of sociological research since scholars are often interested in measuring attitudes, beliefs, and opinions expressed in texts.[9]

3 Data

We use three 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 political scientist 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 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.

3.1 Twitter dataset

The first dataset consists of 1,691 tweets annotated for their stance towards Donald Trump or Hillary Clinton, derived from SemEval, a widely-used benchmark dataset (Mohammad et al., 2016).[10] Each tweet corresponds to a single target, Trump or Clinton, and is labeled with one of three stances: "Favor," "Against," or "None." 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. A potential problem with this dataset is that the data may have already been "seen" by language models, either during pre-training as the data were scraped from the internet or because platforms used the dataset for instruction-tuning. This "data contamination" may violate a fundamental tenet of machine learning as the models may have already seen the test data, although there is evidence that such contamination does not result in perfect memorization (Brown et al., 2020).

3.2 Facebook comments

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 from June 1, 2016, until the election on November 8. This is a novel dataset so there is no risk that annotations have already been seen during pre-training.[11] 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 more efficient than 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!" would be labeled {Against [Trump], Neutral/None [Clinton]}. 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 2000 comments for training and reserve 400 for testing.

3.3 Facebook comment-reply threads

Our third set of analyses extends the stance detection task to longer comment-reply threads. To understand online conversations such as political debates on social media, we cannot simply consider individual tweets or comments but must investigate the threaded structure of online conversations (Backstrom et al., 2013; Berry and Taylor, 2017; Shugars and Beauchamp, 2019). In theory, threads provide additional context that should help the model predict the stance expressed in comments and it makes little sense to analyze replies abstracted from the conversational context (Murakami and Raymond, 2010; Walker et al., 2012; Sridhar et al., 2015). Extending insights from multitask learning (Caruana, 1997), we expect that models will be able to predict comments *and* replies more accurately when they are situated in the context of a thread rather than taken out of context. However, it is difficult to integrate this kind of structured information into conventional classification approaches. Generally, most models using thread data make inferences about entire threads, such as whether the conversation devolves into personal attacks (Zhang et al., 2018) or whether a comment promotes discussion (Naab et al., 2023). This typically involves the construction of hand-picked features to characterize features of the thread and its constituent texts that can be input alongside representations of texts. Graph-based methods have been proposed for inferring stances of posts in reply threads, but these techniques also rely on hand-crafted features that are difficult to scale (Zubiaga et al., 2016). In contrast, text-to-text LLMs enable us to use the entire thread as input and for the model to automatically parse the information about each text. We can also request variable-length responses, such that models can return different predictions depending on the length

of the thread. Our goal is to jointly classify comments and replies from different social media users using a single model.

We draw threads from the same corpus as the previous Facebook task. While some comments can spawn extended discussions, the modal comment receives zero replies, and the modal number of replies among the replied-to only two. Despite their relative rarity, 20.8% of the texts in the corpus are replies, so a substantial amount of discourse is missed if we only consider top-level comments. Each thread includes the page name, the original post text, and the text of any comments and replies, as well as pseudonyms for their authors.[13] To avoid annotating excessively long threads, we restrict our analyses to the first five replies (Only 0.56% of comments in the corpus have more than 5 replies). We construct balanced training ($N = 1200$ threads) and tests ($N = 300$) datasets by randomly sampling threads from the corpus. In each case, there are equal numbers of threads with between zero and five replies (including longer threads that are truncated to the first five replies). For each thread, we use the same annotation scheme as the Facebook task, labeling each comment and reply with a stance toward both Trump and Clinton. We use Support/Oppose rather than Favor/Against as the stance labels, as we prefer these terms, but chose to retain the terminology used in the SemEval task (Mohammad et al., 2016) for the previous analyses.

An initial balanced set of threads ($N = 36$) was annotated by one of the authors and a graduate student with expertise in American politics. The annotators were instructed to take the thread context into account to help make decisions about stance. As such, the interpretation of a comment could be influenced by both the original post and any replies, and any replies could be understood in the context of the entire thread. In this initial evaluation, 90% of the stance labels were consistent across the two annotators. Nearly all disagreements related to whether a stance was present or not, rather than the valence of the stance (e.g. one annotator rated a comment as Support and another Neutral rather than one Support and another Oppose). After discussing the discrepancies, the graduate student then annotated the remaining threads. These evaluation data were also used for prompt development to avoid overfitting the prompt to the training data during development.

Each thread is represented using Javascript Object Notation (JSON), a common data format that allows us to represent threads in a nested structure. This format is well-suited for LLMs, since they are often trained on large amounts of code and associated data structures (Ziems et al., 2024). Alternative formats such as XML or YAML could also function similarly. The following example shows a comment with three replies on a post from Hillary Clinton's Facebook page:

JSON thread example:


```
{"post": {
    "page": "hillaryclinton",
    "text": "Equality is on the ballot.\nJustice is on the ballot.\nOur progress is on the
    ballot. IWillVote.com"
  },
  "comment": {
    "author": "Ian",
    "text": "Sorry but it's time to step down!!! Nothing you say or do anymore is believable."},
  "replies": [
    {"reply_id": 1,
      "author": "Kristie",
      "text": "Go Hillary!"},
    {"reply_id": 2,
      "author": "Arman",
      "text": "Kristie Obama just cancelled all his upcoming Hillary campaign events ! Whatever
      information the FBI has found must be completely devastating for Clinton. So devastating,
      that President Obama can no longer even be seen as supporting her candidacy! This FBI
      announcement has \"criminality\" written all over it."},
    {"reply_id": 3,
      "author": "Sabrina",
      "text": "She needs to step down"}]
}
```


In this example, a commenter responded to Clinton's post, expressing a negative stance by asking her to "step down" and questioning her honesty. Another Facebook user, Kristie (all names are pseudonyms), replied to this comment with a short supportive message, "Go Hillary!". The reply is followed by a longer response, tagging Kristie, and sharing a message insinuating that Clinton is a criminal. Finally, a fourth person enters the conversation, reiterating the claim from the original comment. We also use the JSON format to store the stance labels. The text below shows the labels corresponding to this thread:

JSON labels:


```
{"comment": {
    "stanceTrump": "Neither",
    "stanceClinton": "Oppose"},
```

```
"replies": [
  {"reply_id": 1,
    "stanceTrump": "Neither",
    "stanceClinton": "Support"},
  {"reply_id": 2,
    "stanceTrump": "Neither",
    "stanceClinton": "Oppose"},
  {"reply_id": 3,
    "stanceTrump": "Neither",
    "stanceClinton": "Oppose"}]}
```

In general, this task is a much more demanding prediction problem, as the number of labels varies across threads. Threads with one comment and five replies have 9^6 unique combinations of labels and, for any given input, there are $\sum_{n=1}^{6} 9^n \approx 589$k unique permutations, assuming that a model must output not only the stance labels but the appropriate quantity of predictions. Overall, this task enables us to examine the potential of LLMs to handle more complex data structures than traditional ML models and how these models can be used to provide valuable social scientific insights by enabling us to understand language in a more realistic, conversational context.

4 Models

We compare several language models of varying architecture, size, and open-weights availability to examine which factors have the greatest bearing on the overall predictive performance. Broadly speaking, these models can be categorized into three groups: small, encoder-only models that specialize in encoding semantic information; medium to large decoder-only and encoder-decoder models that are capable of autoregressive text generation; and large proprietary, decoder-only models that can only be run using third-party APIs. The key aspects of each model are summarized in Table 1 and Table 2 details how each model is used in our analyses.

The encoder-only models are all based on the BERT architecture (Devlin et al., 2019), which has been used in several recent sociological analyses (Ren and Bloemraad, 2022; Bonikowski et al., 2022; Le Mens et al., 2023). BERT is a bi-directional encoder model trained as a masked language model, where the objective is to predict one or more masked words in an input. For example, "the cat [MASK] a mouse." We use the smaller `bert-base-uncased` version.[14] BERT is relatively modest in scale compared to more recent language models, with twelve layers of transformer modules that

total 110 million parameters. We also evaluate two extensions of the architecture. SentenceBERT (SBERT) adds a novel pooling operation to output vectors of BERT (Reimers and Gurevych, 2019) to generate fixed-size document-level (i.e., entire input text) embeddings. DeBERTa (Decoding-enhanced BERT with disentangled attention), incorporates two techniques that improve the attention mechanism and the encoding process. We use versions of SBERT and DeBERTa that are comparable to BERT in size, namely `all-mpnet-base-v2` and `deberta-v3-base`, and are the most recent versions at the time of our analysis.

The second, medium-to-large family includes three different architectures, each with billions of parameters. These models are not only far larger than the encoder-only models but also equipped with the autoregressive decoder component that allows for text generation and can be further fine-tuned to follow instructions in the prompts, making them ideal candidates for zero- and few-shot learning. FLAN-T5 builds on the Text-to-Text Transfer Transformer (T5) architecture wherein both the input and the output are natural language (Raffel et al., 2020). FLAN-T5 was created by instruction-tuning T5 on over one thousand NLP tasks to improve its generalizability and instruction-following, including the use of chain-of-thought prompting (Chung et al., 2022). We use the largest variant, `flan-t5-xxl`, which achieved strong performance on a stance detection task derived from our Twitter dataset (Ziems et al., 2024). Next, Mistral, developed by French company Mistral AI, is a decoder-only architecture intended to balance high performance and efficiency. It incorporates two techniques into the transformer architecture — Grouped-Query Attention (GQA) (Ainslie et al., 2023) and Sliding Window Attention (SWA) (Jiang et al., 2023) — both of which were devised to reduce memory overhead that occurs during the handling of longer text sequences. We experiment with Mistral's 7B parameter model, `Mistral-7B-v0.3`, released in May 2024. Finally, Llama, developed by Meta, is a family of models that achieve competitive performance by training on *more* data without necessarily creating a *larger* model (Touvron et al., 2023). We use the two third-generation Llama models released in April 2024, `Llama-3-8B`, which is comparable in size to the other models in this category, and `Llama-3-70B`, the largest open-weights model we evaluate. Both models are trained on 15 trillion tokens of text from public sources, but the exact training data are not disclosed. These models are also decoder-only and use the same GQA procedure as Mistral, as well as modifications to the attention mechanism and embedding representations that enhance performance (Shazeer, 2020; Su et al., 2023).

The final, large proprietary model family consists of several GPT variants released by OpenAI. GPT-3 is the third iteration of LLM developed by OpenAI and demonstrates strong performance in zero-

Model	Architecture	Parameters	System	Open-weights
BERT	Encoder	110M	Personal computer	✓
SBERT	Encoder	109M	Personal computer	✓
DeBERTa	Encoder	86M	Personal computer	✓
FLAN-T5 XXL	Encoder-Decoder	11B	HPC cluster	✓
Mistral-7B	Decoder	7B	HPC cluster	✓
Llama3-8B	Decoder	8B	HPC cluster	✓
Llama3-70B	Decoder	70B	HPC cluster	✓
GPT-3 Ada	Decoder	350M	API	✗
GPT-3 Davinci	Decoder	175B	API	✗
GPT-4o	Decoder	1.7T	API	✗

Table 1: **Model comparisons**

and few-shot settings (Brown et al., 2020). We compare `text-ada-001` and `text-davinci-003` — hereafter Ada and Davinci — released in November 2022.[15] Ada, the smallest variant, has 350 million parameters and is trained on 40GB of text data (making it more comparable in size to the encoder-only models described above). Davinci is the larger model, with 175 billion parameters, trained on 45TB of text. The version we use has undergone additional refinement using RLHF to improve its performance (Ouyang et al., 2022) and shows better performance on social science classification tasks compared to the previous version, `text-davinci-002` (Ziems et al., 2024). A version of this model was adapted for conversations and released as ChatGPT in late 2022. We compare these with, GPT-4o, the second version of GPT-4 that was released in 2024, one of the largest language models at the time of writing. The technical details have not been made public, but GPT-4 is trained on text, images, and audio and is rumored to have 1.7 trillion parameters.[16] We were unable to fine-tune GPT-4o, so it is only used for the zero- and few-shot analyses.[17] We use the OpenAI API to interact with the models running on their servers. OpenAI generally charges more for the larger, more computationally expensive versions. At the time of our experiments with GPT-3 in early 2023, for every 1000 tokens of input — equivalent to approximately 750 words — Ada costs $0.0004, and Davinci costs $0.02, but the fees have since decreased due to improved efficiency. The largest model, GPT-4o, currently (July 2024) costs $0.0025 per 1000 tokens of input if the API provides data in batches and a smaller, cheaper version is also available.

4.1 Baselines

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 approaches, 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. We also compare our results against a random baseline. Our final thread-prediction task is structured as a text-generation task and cannot be performed using classical algorithms. Instead, we use the best-performing model from our Facebook comment analyses to predict stances using each comment and reply separately.

5 Experiments

5.1 Zero-shot and few-shot learning

We perform zero-shot learning for all three tasks. In each case, we provide a prompt and a test example as input and use the model output as a prediction. For the tweet and comment prediction tasks, we also implement few-shot learning. Since each tweet only includes a single target we provide one example for each target for the few-shot learning task (2-shot), whereas we use a single example for the Facebook comment task (1-shot). While more examples could theoretically be used — as many as fit into the *context-window*, which defines the maximum number of tokens that can be input into a model at once (Brown et al., 2020) — we focus on a simple case to make a comparison to zero-shot learning.

In general, the encoder-only models (BERT, SBERT, and DeBERTa) cannot be prompted using text inputs, so cannot perform zero- or few-shot learning. However, it is possible to further adapt these models to perform a rudimentary form of zero-shot learning.[18] To explore this possibility, we use a version of DeBERTa adapted for zero-shot classification (Laurer et al., 2024), `DeBERTa-v3-base-mnli-fever-anli`. The model is fine-tuned on 763,913 pairs of sentences from three datasets designed for Natural Language Inference (NLI). The goal of NLI is to measure relationships between pairs of sentences, known as the hypothesis and the premise (Dagan et al., 2006). The relationship is considered to be *entailment* if a hypothesis follows from a premise, *contradiction* if the hypothesis contradicts the premise, and neutral if they are independent. In the case of stance detection, we can consider the input text as the premise and the stance label as the hypothesis (Yin et al., 2019; Burnham, 2023). For example, the premise "Hillary will be a great president" entails the hypothesis "Favors Clinton". The model works by computing cosine similarities between embedding representations of the prompt and the stance labels and selecting the closest

one. This means that the model can select labels for observations without any additional training even though it cannot process prompts or few-shot examples like more sophisticated text-to-text models.

The other models are all text-to-text and can process full prompts and other input examples. Where possible, we use variants of each model that have been instruction-tuned and optimized for conversation via RLHF and related techniques. Specifically, the models are `Mistral-7B-Instruct-v0.3`, `Meta-Llama-3-8B-Instruct`, and `Meta-Llama-3-70B-Instruct`. We use the standard version of FLAN-T5 XXL, `flan-t5-xxl`, which is designed to follow instructions. Regarding the OpenAI models, Ada has not undergone any instruction-tuning, whereas the version of Davinci used — `text-davinci-003` — and GPT-4o, have been tuned for instruction-following via RLHF.

5.1.1 Prompt engineering

For zero- and few-shot learning, it is necessary to provide information to constrain the model to produce the desired output. Relative 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). To examine the impact of prompt engineering, we test three prompts for each text classification task that vary in length and the amount of information about the task. This allows 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 few-shot models, we also evaluate the sensitivity to the exemplars included with the prompt. There is evidence that performance can vary depending on the choice of exemplars (Zhao et al., 2021). Moreover, prior work on stance detection 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 will be more helpful than others when distinguishing between classes in few-shot settings. For each task, we conduct 100 replications using different examples from the training data. In each case, we take a random sample without replacement from the training data (stratified by the target for the two-shot task to ensure one example corresponding to each target is shown).[19] Each example is concatenated to the prompt and used to predict the labels for all texts in the test dataset. These experiments enable us to measure how the predictive performance of few-shot learning varies according to the exemplar(s) provided.

Model	Zero-shot	Few-shot	Fine-tuned	Instruction-tuned
BERT			✓	
SBERT			✓	
DeBERTa	✓		✓	
FLAN-T5 XXL	✓	✓	✓	
Mistral-7B	✓	✓	✓	
Llama3-8B	✓	✓	✓	✓
Llama3-70B	✓	✓	✓	✓
GPT-3 Ada	✓	✓	✓	
GPT-3 Davinci	✓	✓	✓	
GPT-4o	✓	✓		

Table 2: **Learning regimes by model**
The table lists the learning regimes used for models across all tasks. We perform zero-shot and few-shot learning for the Twitter and Facebook tasks and zero-shot learning for the Facebook comment-reply task. Fine-tuning is conducted on the Twitter and Facebook datasets for text classification. Instruction-tuning is conducted on the Facebook comment-reply dataset for text generation, using a subset of the decoder-only models that show strong performance in previous tasks.

5.2 Fine-tuning

We evaluate how the performance of fine-tuning varies depending on the amount of data used. For the Twitter task, we compare models fine-tuned on 10, 100, and all training examples. We repeat the same for the Facebook task, additionally comparing 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.

The encoder-only models are fine-tuned by adding a layer known as a *classification head* to each neural network. This layer has randomly initialized weights, where each possible label is represented by a numeric parameter. The classification head consists of a fully-connected layer that outputs a logit for each label, and a softmax activation function that converts the logits into probabilities that sum to one. As the training examples are passed through the network, both the existing parameters and those in the classification head are updated through backpropagation to minimize the cross-entropy between the predicted and true labels. Once trained, the test data are input into the fine-tuned models, which then output predicted probabilities for each class. For the larger open-weights encoder-decoder and decoder-only models, the process differs in two ways. First, the language modeling head of each model, the final part of the decoder component, is replaced entirely by the classification head. Fine-tuning thus curtails the generative capacity of these models as they are adapted to generate the stance labels. Second, due to the size of these models, it was necessary to use a technique known as QLoRA to perform this fine-tuning efficiently (Dettmers et al., 2023). In short, QLoRA enables us to only update a small fraction of the available parameters and to store

the matrices in a compressed format (see subsection A.3 for further discussion). Fine-tuning of GPT-3 Ada and GPT-3 Davinci was performed using the OpenAI API.[20] OpenAI does not reveal exactly how its fine-tuning system works, but the process appears to be similar as the fine-tuned models are constrained to only produce the tokens corresponding to the class labels and lose their generative capabilities. Each model has various hyperparameters that can be modified during fine-tuning. Rather than systematically tuning these parameters, which would be computationally (and financially in the case of the GPT models) expensive, we tried to maintain similar settings across all models and generally used defaults or those reported in previous literature. Further technical details on fine-tuning and hyperparameters are discussed in subsection A.3.

5.3 Instruction-tuning

We use instruction-tuning (Wei et al., 2022) for the thread-prediction task. This generative approach allows us to incorporate a dynamic label structure, rather than specifying the fixed number of labels in advance as is required for standard fine-tuning (if there are k classes then the classification head is a k-dimensional vector). Classification tasks can be framed as a variant of instruction-tuning where inputs are the text to be classified, a set of instructions, and the labels. In our case, we provide the instructions in a format known as a *system prompt*, which shapes the overall behavior of the model, and we continue to train it on all Facebook thread JSONs and corresponding stance JSONs. When input with a new thread from the test set, the instruction-tuned model should generate a JSON object containing the correct stances for each comment and reply. As the Facebook threads in our data have zero to five replies, the model should be able to identify the number of replies in each thread *and* the correct stances, generating output text with appropriate headings like the numeric identifier for each reply. We discuss the prompt in more detail in subsection 6.4. To perform these analyses, we instruction-tune the two Llama3 models, which have already undergone instruction-tuning and RLHF (Touvron et al., 2023). This process is implemented using the Supervised Fine-Tuning trainer from the Python library `trl`. We compare the performance of instruction-tuned models to zero-shot learning (including the most powerful model, GPT-4o) and a baseline model predicting comments and replies in isolation using the prompt from the comment prediction task.

5.4 Evaluation metrics

All models are scored by calculating predictive performance on the held-out test data.[21] 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. For each class, *precision* is the number of true positives divided by the number of true positives and false positives and *recall* is the number of true positives divided by the number of true positives and false negatives. Precision captures how accurately a classifier detects a particular class, whereas recall measures how many relevant examples were detected. 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 (e.g. (Jensen et al., 2022, 46)). 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 measure the overall performance across classes, $F1_w$. 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:

$$F1_w = \sum_{k=1}^{K} F1_k \cdot \frac{N_{test_k}}{N_{test}}$$

For example, $F1_{Clinton}$ is a weighted average of the F1 scores across three classes, "For," "Against," and "Neither/None." We calculate several different versions of each score depending on the task. For Twitter, we score models separately for target and stance prediction. For Facebook, we calculate the stance prediction accuracy for each target. We mostly focus on the strictest performance metrics, which we term $F1_{joint}$, which count a prediction as correct when it exactly matches the original annotation. 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 target. 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.[22] Given the large label space for the thread-prediction task, we calculate aggregate scores at the comment/reply level.

Since the test datasets are relatively small it is possible that the scores on the held-out data do not adequately account for variation that would occur if the models were used on larger corpora of out-of-sample data. To account for this uncertainty, we calculate bootstrap confidence intervals for each metric (Efron and Tibshirani, 1986). These are obtained by drawing N predictions with replacement, where N is the size of the test data, and using these to calculate performance scores.

The procedure is completed 10,000 times and the results are aggregated to obtain 95% bootstrap confidence intervals. These intervals capture variation due to the composition of the test data, demonstrating uncertainty in the application of models to new data, although they do not capture other sources of uncertainty such as the composition of the training data or stochastic variation in the models.

6 Results

6.1 Prompt engineering and zero-shot learning

To assess the relationship between prompts and predictive performance, we created three different prompts for each task. We begin with a simple prompt (`minimal`) listing the minimum information needed to perform the task: the stance options and the format of the answer (see Table 3 for the full prompts). If this works, we assume that the model uses existing semantic information from pre-training to "understand" what target and stance are and how they relate to the input. These prompts are extended by adding a short sentence describing the task in a more natural style that is closer to the type of instructions provided to instruction-tuned models (`sentence`). The final pair of prompts is more informative, describing how the statements refer to politicians and represent attitudes (`context`). This allows us to assess the extent to which additional context enhances the accuracy of the model. Ceteris paribus, we anticipate that more context should improve performance.

The results of the experiments are shown in Figure 1, which displays the target-specific and joint F1 scores for each prompt evaluated on the relevant test data. Overall, the more informative prompts tended to receive the highest joint F1 score across both tasks. This pattern is most evident in the Facebook data, where the contextualized prompt significantly outperforms the minimal example. The second prompt also results in some improvement, suggesting that more complete sentences can be helpful, although the difference is not statistically significant after accounting for variation in the test data. Looking at the target-specific scores, the results show that the performance for each candidate improves across the two prompts, particularly for the stance toward Trump. The results from Twitter are noisier, as the simple prompt seems to work better than the intermediate one, but the final prompt still performs better overall. Breaking the performance down by candidate shows that this is due to an increase in performance for Clinton and a small decline for Trump. In general, the labels in the Twitter dataset appear to be more difficult to predict, an issue we discuss further

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}`	`Sentence`	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}`	`Context`	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}`	`Sentence`	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}`	`Context`	54

Table 3: **Prompt variations**

† Token consumption for GPT-3 models calculated via OpenAI's tokenizer: `https://platform.openai.com/tokenizer`

below. While we do not attempt to identify the best prompt exhaustively, these results underscore the importance of careful prompt selection.

6.2 Exemplar selection for few-shot learning

The one- and two-shot analyses were performed using the most informative prompts from the preceding analysis. Figure 2 shows the results of our experiments analyzing how performance varies as a function of the exemplars provided in the few-shot tasks. Across both datasets, there is substantial variability in predictive performance on the test data, highlighting sensitivity to the exemplars. 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 are more accurate at predicting the stance towards Clinton than Trump, likely because the stances expressed toward Clinton are more consistent. For the one-shot Facebook model, we observe a strong, positive correlation between the F1 scores for each target. This suggests that the best exemplars help improve predictions for both candidates. The

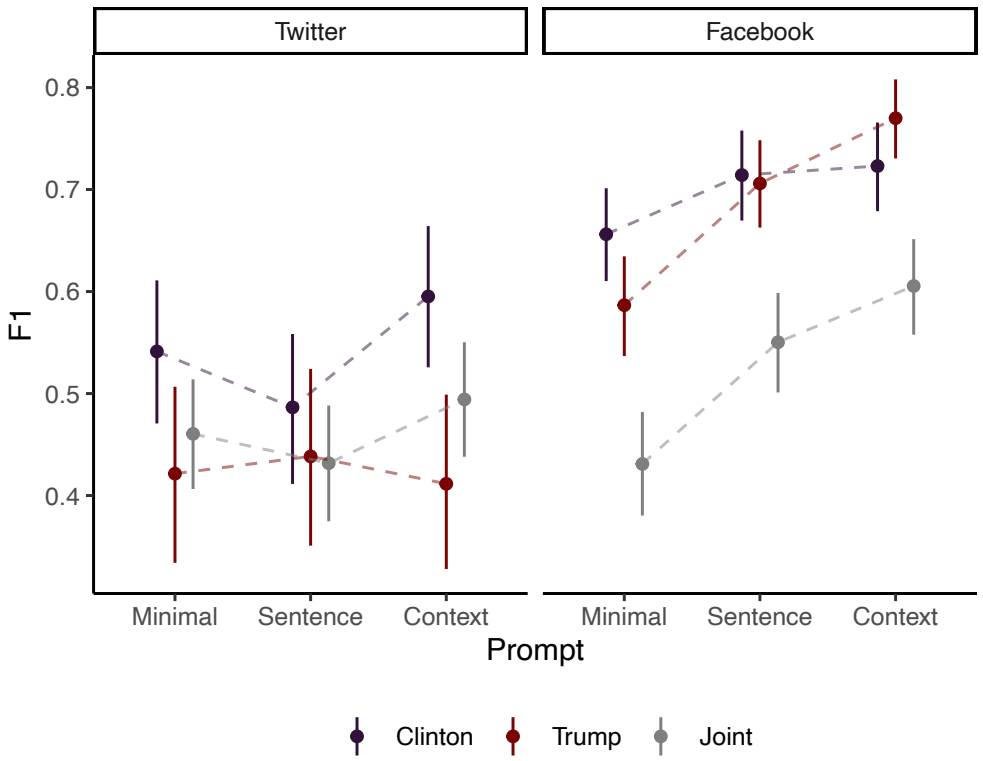
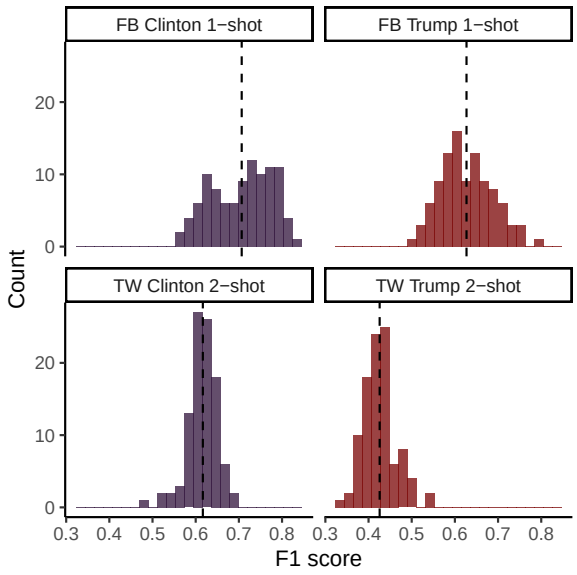


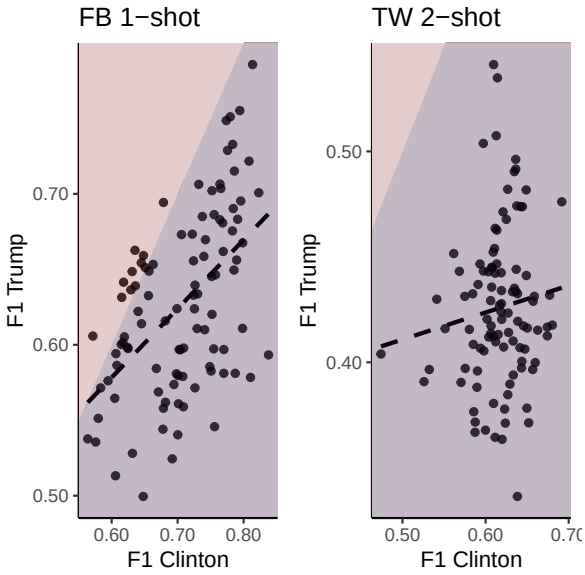
Figure 1: **Zero-shot F1 scores by prompt variations**
This figure shows the test set F1 scores for zero-shot learning with GPT-3 Davinci using the three different prompts for each task, with Twitter on the left and Facebook on the right. Separate F1 scores for each candidate are shown as well as the strictest joint F1 score. The error bars are 95% bootstrap confidence intervals.

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 exemplars 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. Overall, these results highlight how LLM classifiers using few-shot learning can be highly sensitive to the choice of exemplars, with evidence of wide variation in the out-of-sample F1 scores across different exemplars.

To better understand how the characteristics of the exemplars impact performance, we estimated a series of regression models, using the stances in the exemplars to predict the target-specific F1 scores, while controlling for word count to account for the effect of differences in exemplar length. The full regression results are reported in the Appendix in Table A3 and Table A4. Overall, we find that comments that favor one candidate over the other are associated with statistically significant increases in performance in the Facebook predictions: 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



(a) F1 distributions by target. The top row the results for the Facebook task and the bottom row for the Twitter task. Dashed lines indicate the mean F1 score across all exemplars.



(b) Correlations between target-specific F1 scores. The left plot shows the results for the Facebook task and the right plot the Twitter task. Shaded regions indicate when models perform better for Trump (red) or Clinton (blue). The regression line is indicated in black.

Figure 2: **Exemplar variation and predictive performance.**
These plots show the variation in overall predictive performance across different exemplars used in 1-shot or 2-shot learning. Panel (a) shows the overall distribution of F1 scores and panel (b) shows the correlations between the target-specific F1 scores for each dataset.

those with no stance towards Clinton; comments favoring Trump are associated with a 0.03 increase in the Trump F1 score. The R-squared statistics increase when the interaction between the stances for the two candidates is added, and the stance interactions in the Trump model are statistically significant, highlighting how the combination of stances expressed in an example matters. The patterns in the Twitter predictions are less clear but suggest that longer tweets help to improve the Clinton predictions and that tweets opposing Trump aggravate the Trump predictions. These results indicate how the qualities of the exemplars provided can impact the predictive performance of few-shot learning models. We account for this variability in the following analyses by using the modal prediction across all exemplars as our final prediction.

6.3 Stance detection across models and learning regimes

Figure 3 shows our main results for the tweet and comment prediction tasks. The points are the joint F1 scores, representing the overall accuracy at predicting the target and stance of tweets and the stances for both targets in comments, along with bootstrap confidence intervals. Table A5 and Table A6 contain the F1 scores for all models, as well as the precision and recall metrics. Overall,

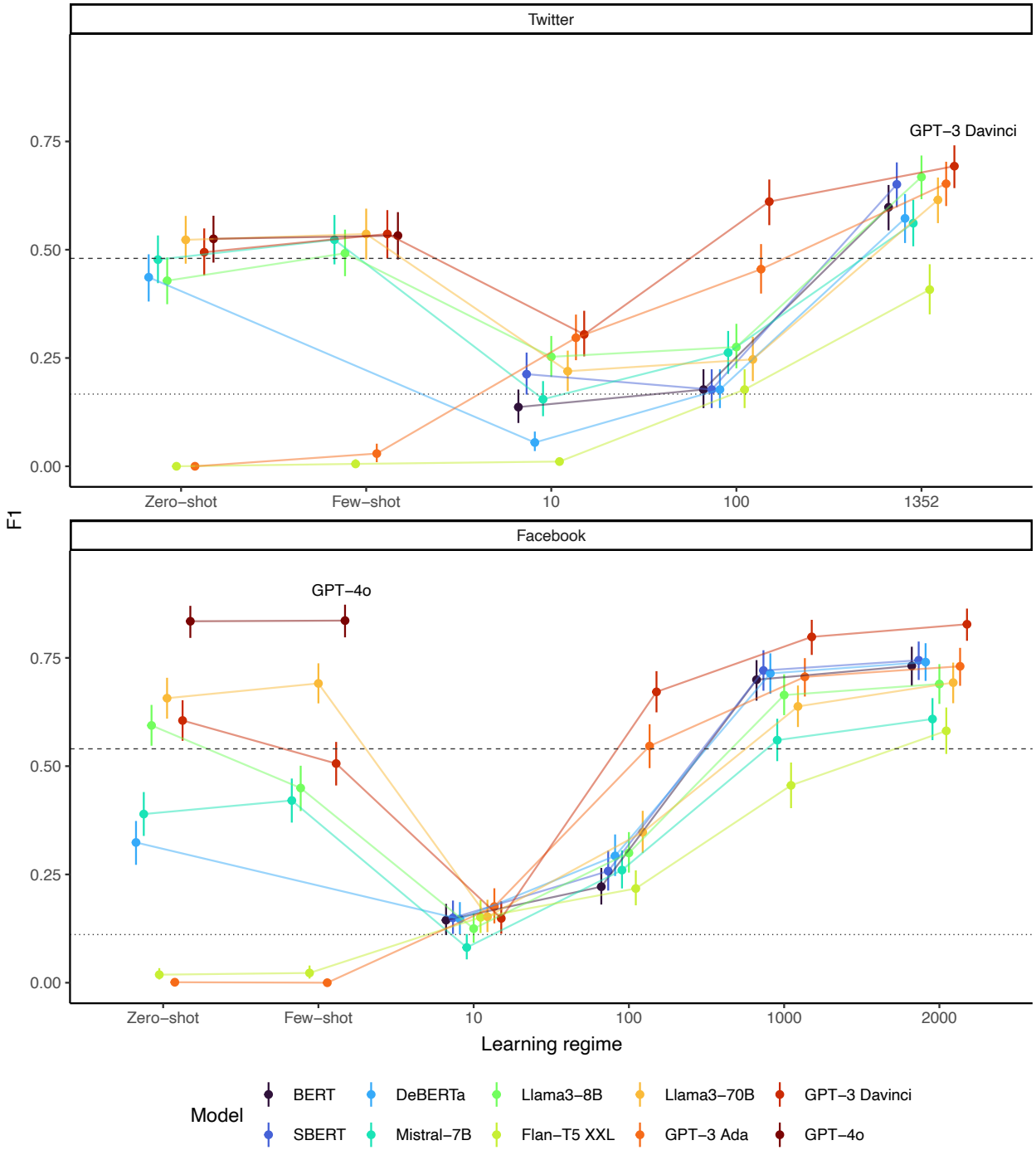


Figure 3: **Predictive performance by learning regime and model**
The top panel shows the overall scores for Twitter, where the F1 score is the joint accuracy across both stance and target on the held-out test data. The bottom panel shows the scores for Facebook, where the F1 score represents the joint accuracy of the stance predictions for both targets on the held-out test data. The best model for each outcome is labeled in the figure. The dashed horizontal line in each panel represents the F1 score of the best-performing baseline model and the dotted horizontal line represents a random baseline. Error bars are 95% bootstrap confidence intervals.

we observe substantial variation in predictive performance across the different models and learning regimes. Starting with the zero-shot and few-shot models, we see that the largest decoder-only models tend to perform best (Llama3-70B, GPT-3 Davinci, and GPT-4o) at predicting the stance and target for tweets. Indeed, zero-shot GPT-4o achieves the highest best precision for both stance and target prediction of any of the models tested (Table A5). The encoder-only DeBERTa model performs comparably to the smaller Llama3-8B model at zero-shot learning, although both perform worse than the best baseline model (both the CNN and SVM with bag-of-words features received a joint F1 score of 0.48). At the bottom of the figure, we see that the smaller GPT-3 Ada and the encoder-decoder Flan-T5 XXL perform poorly for these prompt-based learning regimes, with worse accuracy than random guessing. Inspection of the predictions shows that these models mostly failed to return the predictions in the correct format, sometimes yielding only one portion of the answer or irrelevant text sequences, as shown for the example texts in Table 4. Turning to the Facebook task, GPT-4o substantially outperforms the other models at both zero-shot and few-shot learning. It even scores better than all of the fine-tuned models on the task with a joint F1 score of 0.84 when used as a one-shot learner (the one-shot scores are calculated using the modal labels across all 100 exemplar texts). GPT-3 and the Llama3 models also outperform the baseline. The zero-shot GPT-3 Ada and Flan-T5 XXL, on the other hand, also show worse-than-random performance at predicting the stances expressed in Facebook comments. Across both tasks, adding the exemplars alongside the prompt results in improvement for some models but a decline in others, so it is not clear that this strategy is superior to the prompt alone for these tasks. For example, Llama3-8B and GPT-3 Davinci substantially decline in accuracy on the Facebook task when few-shot learning is used.

The models fine-tuned on 10 examples perform uniformly poorly across both tasks, far below the simple baselines, and, at best, marginally better than random guessing. This further illustrates the strong relative performance of the prompt-based strategies, which can perform significantly better with little or no training data. Focusing on the Twitter task, most models still fare poorly when using 100 examples. We observe the most significant upticks in performance for the two GPT-3 models, although only the larger Davinci model outperforms the baseline model. DeBERTa and Flan-T5 XXL also show significant improvement, but their overall performance is still weak. The three encoder-only models perform comparably to random guessing when predicting targets and stances in tweets. All models fine-tuned using the entire dataset (N=1352) show substantial improvements, all but one significantly outperforming the baseline. The largest fine-tunable model, GPT-3 Davinci, has the best overall performance on the task (F1 = 0.69). However, the smaller GPT-3 Ada and

Twitter		Zero-shot			Fine-tuned		
Text	Label Target, Stance	FLAN-T5 XXL	Llama3-70B	GPT-3 Ada	FLAN-T5 XXL	Llama3-70B	GPT-3 Ada
Hey Hillary Clinton...How's things working out on #Servergate? Bury enough evidence yet? #tcot	Clinton, Against	, A	C, A	,	C, A	C, A	C, A
Donald Trump is a joke. His a simple-minded idiot, and I have nothing else to say about him. He can go F**K him self.	Clinton, None	, A	T, A	, F	T, A	T, A	T, A
Watching what Donald Trump said about Mexicans was shocking! Let's not give this appalling man a platform.	Trump, Against	, A	T, A	,	T, A	T, A	T, A
@GeraldoRivera @realDonaldTrump Don't apologize for the truth. People are too sensitive. #MakeAmericaGreatAgain	Trump, Favor	, F	T, F	,	C, A	T, F	T, F
Facebook		Zero-shot			Fine-tuned		
Text	Label Trump, Clinton	FLAN-T5 XXL	Llama3-70B	GPT-3 Ada	FLAN-T5 XXL	Llama3-70B	GPT-3 Ada
I'm with her...as President Obama says "the most qualified individual ever to run for the office." Our next President!	None, Favor	F, F	A, F	F,	F, N	N, F	N, F
Fairy tales are more then true not because they tell us dragons exist,but because they tell us dragons can be beaten.	None, None	N,	N, N	A,	N, N	N, A	N, N
I agree Trump is Bad. But YOU are bad too. I'm NOT picking between the lesser of two evils. Trump WILL beat you! Only Bernie CAN defeat Him. #notwithher #bernieorbust	Against, Against	A, A	A, A	F,	N, F	N, A	F, A
"Tweeting" at 3 and 4 o'clock in the morning is the sign of a sick and deranged individual. He brags about his beautiful wife, why in the hell isn't he in bed with her?????????	Against, None	A,	A, N	F,	N, N	A, N	A, N

Table 4: **Example texts and predictions from zero-shot and fine-tuned models**
This table shows the predictions for four different examples in the test data for each task. For each example, the predictions from three models are shown: FLAN-T5 XXL, Llama3-70B, and GPT-3 Ada. Predictions are shown for the zero-shot and fine-tuned versions of each model, where the latter was trained on the entire training dataset. The Label column shows the true label for each text. The values underneath each model are the text output. Missing values indicate each model failed to generate the labels in the correct format. To conserve space, the labels have been abbreviated as follows: C=Clinton, T=Trump, F=Favor, A=Against, N=None.

Llama-3 8B also perform reasonably well, along with the other fine-tunable models, except for Flan-T5 XXL. The results show that fine-tuning does not suffer from the same incompleteness problem as zero-shot learning, as all predictions consist of valid target-stance pairs, as illustrated in

Table 4. In general, the results from fine-tuning models to predict stances in tweets indicate roughly linear increases in accuracy as the number of training examples increases on a logarithmic scale.

The overall patterns are similar for the Facebook task, with GPT-3 Davinci performing best when fine-tuned (F1 = 0.83), equaling the zero-shot performance of GPT-4o (Table A6 shows that the model was marginally worse at predicting the stances towards Trump). Ada and the encoder-only models all show improvement with more fine-tuning and achieve higher scores than most of the other decoder-only models. The two Llama models perform similarly, while Mistral-7B and Flan-T5 XXL show less improvement. In general, fine-tuning on larger amounts of data tends to substantially improve performance across both tasks. There are, of course, some exceptions. Llama3 70B shows strong performance on the prompt-based task but scores slightly worse on both tasks than its smaller sibling when fine-tuned. Its overall accuracy when fine-tuned on the full dataset is comparable to its few-shot performance. Flan-T5 XXL shows consistent improvement as it is fine-tuned but still performs significantly worse than other models. Finally, 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).

The scores for the Facebook task are generally higher than those on Twitter, particularly for the zero-shot and few-shot models. This is despite the fact that Facebook comment prediction is a more complex task, with nine possible outcomes compared to six for the Twitter task and many tweets explicitly mentioning one of the two candidates, often making it easy to infer the target (GPT-3 Davinci achieves F1 = 0.89 for the target prediction component). We expect that these discrepancies are due to the composition of the two datasets. The Facebook comments are generally longer, providing more relevant information to the classifier. More critically, a closer inspection of the Twitter data reveals some data quality issues. For example, the second tweet in Table 4 is labeled as mentioning Clinton but not expressing a stance but the text clearly contains a strong anti-Trump sentiment. All three fine-tuned models predicted what appears to be the correct label but are considered incorrect according to the labeled data. We identified other similar examples with inaccurate labels when inspecting the data. As such, the Twitter task scores may be an underestimate of the predictive accuracy of these models due to flaws in the evaluation metrics. Additionally, the fact that the Twitter predictions were worse despite the data being available online, and potentially used to train some of these models, provides further evidence that data contamination does not lead to perfect memorization, consistent with prior work (Brown et al., 2020).

6.4 Stance prediction in Facebook comment-reply threads

We now move to the final part of the analysis, predicting stances in comments and replies in discussion threads on Facebook. Each thread consists of a post (by either Donald Trump or Hillary Clinton), followed by a comment, and between zero and five replies. The task is to predict the stance towards the two candidates in the comment and every reply in each thread. These models take the full JSON object as input along with the following prompt that describes the task and the format of the output:

```
System prompt:
You will receive JSON inputs representing discussion threads from social media during
the 2016 US Presidential election. Each thread includes a post, one comment about the
post, and up to five replies to the comment. Your task is to identify the stance
expressed towards two politicians, Donald Trump and Hillary Clinton, in the comment and
each reply. Each text may express a stance towards one, both, or none of the politicians.
You will always provide a stance towards each politician separately.

Stance Options:
        Support: Positive attitude towards the politician.
        Oppose: Negative attitude towards the politician.
        Neither: No clear stance or irrelevant content.

Instructions:
        - Identify the stance for Trump and Clinton in the comment and each reply using the
          stance options provided.
        - Always provide a stance even if the content is offensive or ambiguous.
        - There will be between zero and five replies to each comment. If there are fewer
          than five replies, provide stances for the available replies only.

Output Format:
Strictly follow this JSON format. Replace the STANCE placeholder with the actual stance.
Do not add any other tokens:

{ "comment": {
        "stanceTrump": "STANCE",
        "stanceClinton": "STANCE"},
  "replies": [ {"reply_id": 1,
```

```
                "stanceTrump": "STANCE",
                "stanceClinton": "STANCE"},
                {"reply_id": 2,
                "stanceTrump": "STANCE",
                "stanceClinton": "STANCE"},
...]}
```

The prompt has several different components. It begins with a paragraph describing the task, explaining the inputs and the context. Next, the stance labels and short explanations are provided. Additional instructions are then listed in bullet points, including an emphasis that the model should return output even for "offensive or ambiguous" content since experimentation with the model showed that it sometimes refused to analyze texts including offensive language. The instructions also specify how to handle threads of different lengths. Finally, a precise description of the output format is given, including instructions not to return any other tokens. This is important because instruction-tuned models often include other language to make the responses more conversational (e.g. "I am happy to help with your request, ..."). A truncated JSON is included to show how the output should be formatted. While there is some redundancy in the prompt, we erred on the side of providing more detail and were satisfied that the prompt steered the model to produce the appropriately formatted output through tests on the evaluation data. Of course, further prompt engineering could result in additional improvement.

As a baseline, we use GPT-4o, the best-performing model from the comment prediction task. Each comment and reply is classified based on the text alone using the same prompt as the comment prediction task (the class labels are changed accordingly). We then calculate joint F1 scores across all comments and replies. The results for all three sets of models are shown in Figure 4. In this case, the baseline model performs considerably worse than in the previous analysis (F1 0.64 versus 0.84). We expect this discrepancy is attributable to the fact that the comments and replies in each thread were annotated in context, such that information in other texts that was used to infer the stances is not available to the classifier, whereas the previous dataset was annotated only using the text alone. When the new prompt and full JSON inputs are used, GPT-4o shows substantial improvement, demonstrating the importance of the information in the thread. The smaller Llama3-8B model performs poorly at the zero-shot task, with lower accuracy than the baseline, suggesting that it cannot sufficiently parse the prompt and thread. The larger version, Llama3-70B, on the other hand, achieves slightly better performance than OpenAI's state-of-the-art model, although the difference

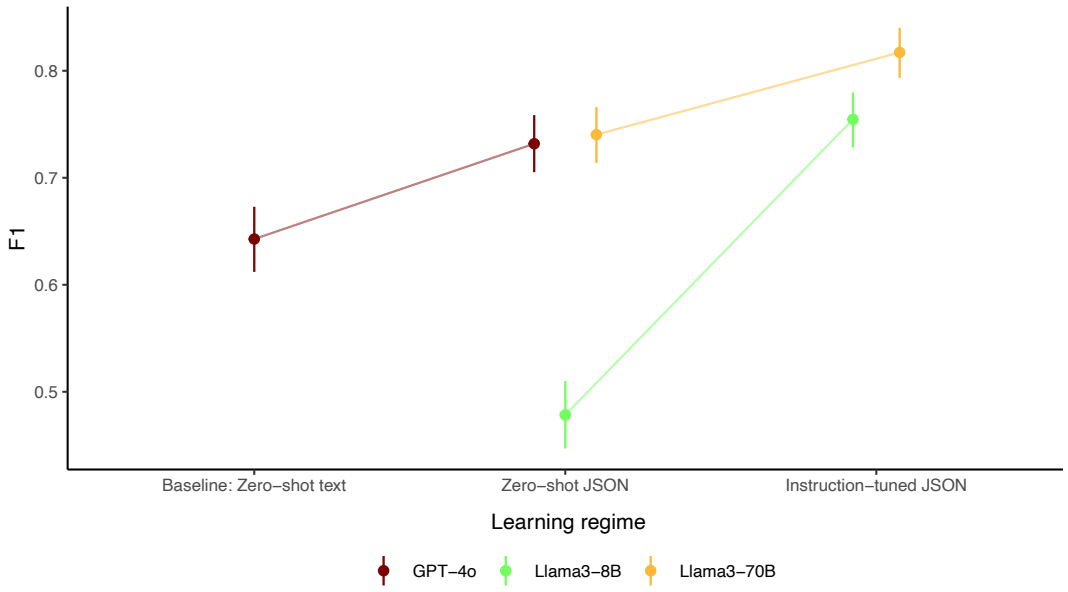


Figure 4: **Aggregate thread-prediction F1 scores by learning regime and model**
The figure shows the aggregate F1 scores, representing the joint accuracy across both targets for all comments and replies in the held-out test data, for GPT-4o, Llama3-8B, and Llama3-70B. The baseline model uses the text only and the same prompt as the Facebook comment prediction task. The zero-shot JSON models use the full thread JSON and prompt. These models are instruction-tuned on the JSON data. Error bars are 95% bootstrap confidence intervals.

is not statistically significant. When instruction-tuned on 1200 annotated threads, both Llama3 models show significant improvement. The joint F1 score from Llama3-8B jumps from 0.48 and 0.75, marginally higher than the zero-shot scores of the larger models. This shows that even the smaller models can achieve strong performance with more task-specific training. Llama3-70B achieves significantly better performance than all other models, with a joint F1 score of 0.82, only slightly below the 0.84 obtained by the best comment-level model in the other Facebook task. When the scores are split by candidate, we see the instruction-tuned model performs remarkably well for the individual predictions, achieving F1 scores of 0.89 and 0.90 for Clinton and Trump respectively (see Table A.5). This showcases how the generative capacity of these models can be leveraged to achieve strong performance on complex, structured text classification tasks.

Figure 5 shows the joint F1 score broken down by the thread position and length, which enables us to understand how the predictive accuracy varies according to whether a text is a comment or reply, and according to the position of replies. Due to the relatively small sample sizes in the test data (each position-length combination is observed at least fifty times), few differences are statistically significant, but there are some suggestive patterns. The first panel shows the performance across comments by the thread length. Comments with two replies appear to be more difficult to predict than others, but the differences are relatively small, ranging between 0.82 and 0.9. There is stronger

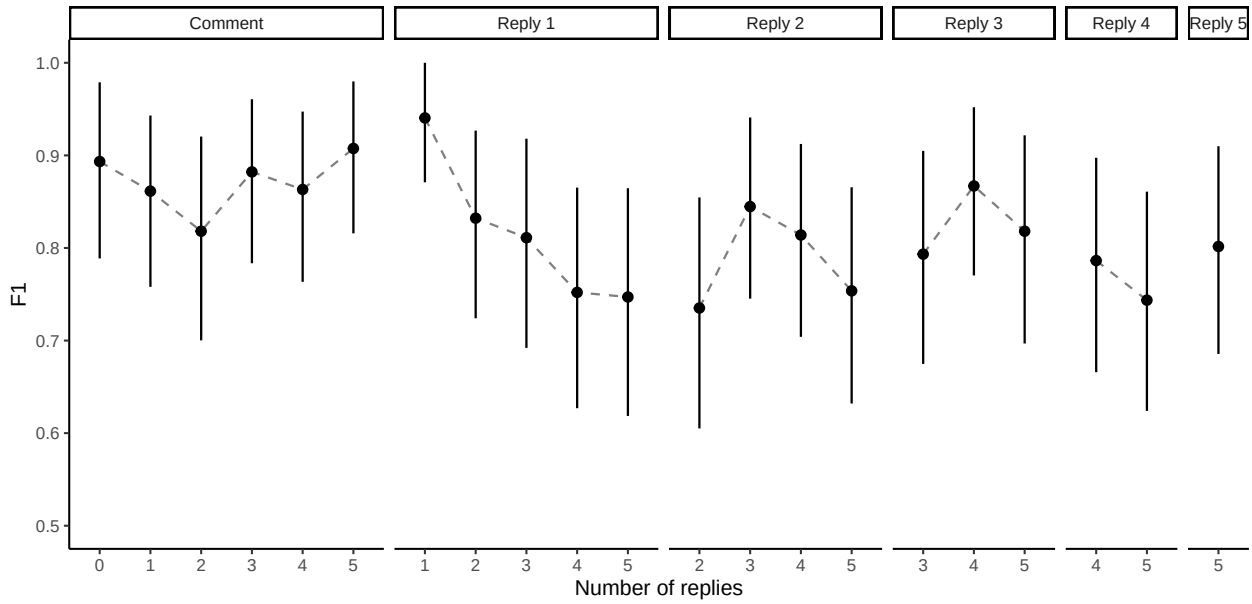


Figure 5: **F1 scores by thread position and length**
The left-hand panel shows the held-out F1 scores for top-level comments and each of the following panels shows held-out performance for sequentially numbered replies. Scores within each panel are grouped based on the number of replies in the thread. Error bars are 95% bootstrap confidence intervals.

evidence of a pattern when considering the initial replies. The stance expressed in replies that do not have any follow-up replies is the easiest text to predict, with an F1 score of 0.94, but the stance in the initial reply becomes more difficult to predict as the length of the thread increases. Besides the initial replies, the accuracy ranges between 0.74 and 0.87 across all thread lengths and reply positions. For the second and third replies, it appears that the stance is predicted more accurately when at least one additional reply is included in the thread. These results thus demonstrate how some parts of the thread can be more difficult to predict than others. Nonetheless, the fine-tuned model can predict the stance of both comments and replies with much higher accuracy than a baseline using text alone, and achieves accuracy comparable to models fine-tuned to predict stance in comments alone.

7 Discussion

7.1 Findings

Large language models represent an important methodological innovation that opens up new ground for the sociological study of texts. We examined the performance of LLMs for text classification, a form of supervised machine learning used to automatically group texts into categories based on the schema defined by the analyst. Our results show that LLMs can accurately identify stances in social

media texts and outperform conventional machine learning approaches by a considerable margin. By experimenting with four different learning regimes, we demonstrate the different ways in which these models can be deployed.

When provided with a prompt containing detailed instructions, the largest, most powerful model tested — OpenAI's GPT-4o — can perform text classification with high accuracy in zero-shot settings. Comparable performance was only achieved by fine-tuning its predecessor, GPT-3 Davinci, on thousands of labeled examples. This demonstrates how LLMs can be adapted to new tasks without any task-specific training, representing a paradigm shift in the use of machine learning tools. While the closed, state-of-the-art models perform best, smaller, instruction-tuned open-weights also show promising zero-shot performance. Using prompts alone, both the 8B and 70B parameter variants of Llama3 outperform the conventional baseline when predicting the stance of Facebook comments. Our evaluations of few-shot learning are less conclusive. Predictive performance is sensitive to both the wording of the prompt and the exemplars provided. Depending on the model, the addition of one or two exemplars along with the prompt can somes improve performance but can also result in significant declines. Nonetheless, we only considered minimal cases, and it is plausible that utilizing the full context-windows would lead to substantial improvements (Brown et al., 2020), particularly for the larger models that can accommodate more exemplars. Moreover, we only tested a handful of hand-crafted prompts, so better performance may have been obtained by using newly developed prompting techniques (Wei et al., 2023a; Kojima et al., 2024) and leveraging LLMs to assist with prompt engineering (Zhou et al., 2022; Khattab et al., 2023; Yuksekgonul et al., 2024).

Regarding fine-tuning, models that are shown only ten examples tend to perform poorly, highlighting the importance of the prompt for zero- and few-shot tasks. A small number of examples is thus insufficient to provide meaningful updates to the classification heads. OpenAI's models improve markedly with as few as one hundred fine-tuning examples, but once a model is trained using one thousand or more examples, the smaller models, including both encoder-only models in the BERT family and larger decoder-only models, begin to approach the performance of the far larger classifiers. Indeed, all of the billion parameter models — except GPT-3 Davinci on the Facebook task — were rivaled by the smaller models. The largest open-weights models showed relatively weak performance when fine-tuned: Flan-T5 XXL models performed worse than other models on both tasks, and Llama3-70B was equaled or bettered by its 8B sibling. This shows that fine-tuning

smaller encoder-models can be much more efficient than trying to adapt larger transformers, despite the use of sophisticated optimization techniques (Dettmers et al., 2023).

While the fine-tuning techniques adapt LLMs to perform multi-class classification, our use of instruction-tuning for the thread-prediction task demonstrates how the generative capability of LLMs can be leveraged to perform much more complex tasks (Wei et al., 2022). The largest models, GPT-4o and Llama3-70B performed reasonably well at generating the JSON responses in a zero-shot setting, outperforming a baseline model predicting text without context by a considerable margin. While the smaller Llama3-8B model fared relatively poorly, the model matched the zero-shot performance of these larger models after instruction-tuning. The larger 70B parameter variant showed even more improvement, demonstrating the returns to scale when instruction-tuned LLMs are applied to complex text classification tasks (Chung et al., 2022). Overall, when evaluated at the comment and reply level, this model often achieved comparable performance to the previous comment-level task, despite the additional complexities involved in jointly predicting the stances of multiple comments and replies. Whereas previous work finds that LLMs can struggle with longer documents and conversational data (Ziems et al., 2024), our findings show that structured data and instruction-tuning can enable these models to process complex sequences of information. This highlights how LLMs can be leveraged to perform difficult, multifaceted kinds of text classification tasks with high accuracy, opening up new possibilities for computational social science.

Stance detection is a particularly useful form of text classification for sociologists and other social scientists since we often want to use texts to infer attitudes, opinions, and beliefs (Bestvater and Monroe, 2022). Our case study has the most direct relevance to research in political sociology since our case study focuses on the stances expressed about candidates during an election campaign, 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 makes it simple to experiment with the capabilities of these models across a range of different tasks (Davidson, 2024). Nonetheless, it is important to note that there may be substantial variation in performance depending on facts such as the type of inputs, the complexity of the labeling scheme, and the degree to which the labeling involves either expert knowledge or subjective decision-making (Ziems et al., 2024). While we are confident that LLMs will work effectively for many types of text classification and other NLP tasks, our findings do not imply that these techniques will be the most effective for all tasks. Rather, our goal is to provide researchers with concrete recommendations for using these techniques, to which we now turn.

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. Our main recommendations are summarized in Figure 6.

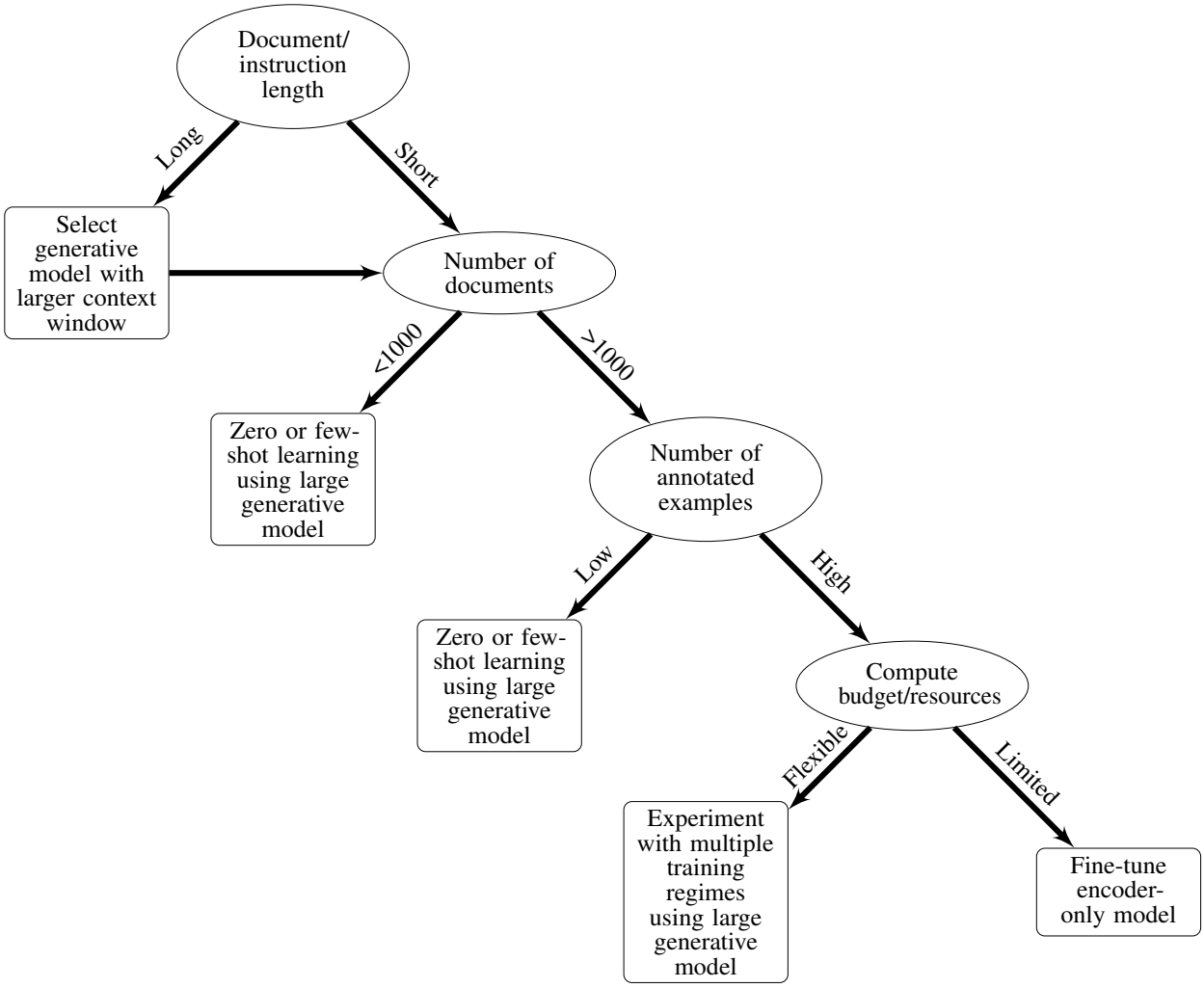


Figure 6: **Selecting a suitable approach to using LLMs for text classification**

The social media posts used in our initial experiments are mostly short, although some Facebook comments extend to several paragraphs. If the research necessitates analyzing long documents consisting of multiple pages, such as newspaper articles, academic papers, or legal filings, then it will be necessary to use models with longer context-windows. The version of GPT-3 used here could handle just over 2k tokens (approximately 1500 words), so would not be sufficient for processing long texts, but more recent models can accommodate considerably more text. GPT-4o

has a context-window of 128k tokens and Google's largest Gemini model can handle up to 1M tokens.[23] Such models can handle long-range dependencies like identifying how repeated mentions of a character throughout a novel correspond to the same entity. Longer context-windows can also be advantageous for few-shot learning since they can use more training exemplars along with the prompt and may be needed for complex instruction-tuning tasks. Longer documents can also be more difficult to classify (Ziems et al., 2024), so using larger, more powerful models may also be desirable. Longer documents can, of course, often 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). 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. Whereas conventional supervised learning approaches typically require relatively large sets of annotated training data, LLMs now make it possible to perform text classification on relatively small datasets. 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) (Ibrahim and Voyer, 2024). We recommend experimenting with different prompts and exemplars to optimize these approaches, as our experiments demonstrate that both can have substantial impacts on both predictive performance and cost. Maximizing diversity by selecting a range of exemplars with a variation in both texts and labels may be a promising strategy. If the number of documents is large — we use 1000 as a cut-off for large versus small-N, but the number is arbitrary — the strategy will depend on the amount of annotated data available. If only a small number of annotated examples are available, we also recommend using zero or few-shot learning, which can achieve strong performance with little or no annotated data. It may also be possible to fine-tune a state-of-the-art model with a modest amount of annotated data Do et al. (2022). Our experiments show that GPT-3 Davinci performs reasonably well on both tasks when fine-tuned on only 100 labeled documents. If zero/few-shot models perform poorly or are too expensive to use, then it will be necessary to annotate more data to achieve a satisfactory model. The use of LLMs to augment annotations to help produce larger training datasets is also a promising avenue for making the development of training data corpora faster and cheaper (Gilardi et al., 2023; Heseltine and Clemm von Hohenberg, 2024; Ziems et al., 2024). Regardless of the approach used, it is critical to annotate a sample of documents to test and validate the performance of any classification model.

If the goal is to classify a large sample of documents and labeled data are available, then the options are only constrained by compute budgets and availability. In many settings, the goal of supervised text classification is to apply a trained model to a much larger unlabelled corpus. A realistic task might involve classifying a million or more comments on a social media platform. While the costs of using advanced models have decreased as the technologies have become more efficient, this can still be expensive when using commercial systems. As things stand, using commercial models at scale could easily cost hundreds, even thousands, of dollars for the kinds of classification problems that arise in computational social science.[24] The open-weights alternatives can be considerably cheaper, particularly if hardware is readily available. We used two university-owned Nvidia A100 GPUs to fine-tune Llama3-70B, which retailed for $12,995 at the time of writing, but similar hardware can be rented via cloud computing providers for a few dollars per hour.[25] As compute costs continue to decline, we expect open-weights models to be a much more economical solution for large-scale classification tasks. If advanced computing infrastructure or funds to pay for APIs and cloud computing are unavailable, the smaller encoder-only models that can be fine-tuned on a laptop will be preferable for text classification tasks. Indeed, we expect many researchers will prefer to sacrifice some accuracy to use a smaller model at a fraction of the cost of a larger model. However, it will not be feasible to perform complex text generation tasks like instruction-tuning at scale.

Those with more resources will have more options at their disposal, and we encourage experimentation with larger generative models to identify the optimal approach for a specific task. Our results show that prompt-based learning strategies can be highly effective when using the largest models and that fine-tuning and instruction-tuning can lead to even better performance. For relatively simple prediction tasks, fine-tuning large models may offer some improvement compared to smaller encoder-only models or prompt-based strategies. However, both prompting-based learning and fine-tuning struggle for difficult prediction tasks that are commonly encountered in social scientific research (Ziems et al., 2024). Instruction-tuning (Wei et al., 2022) combines the instructional capabilities of prompt-based approaches with the customizability of fine-tuning, making it a powerful technique that could make supervised learning more applicable to a range of different areas of sociological research. We expect that instruction-tuning will be particularly helpful in cases where the number of labels is large, the coding scheme is complex, multiple labels are needed, or the inputs are highly structured. But tasks can be difficult for a variety of reasons and this list is not exhaustive. While the largest decoder models are best suited for this task (Chung et al., 2022), our

results show that strong performance does not necessarily require the use of commercial models, since open-weight models can perform capably when instruction-tuned.

Due to the complexities involved in working with LLMs, the choice of model will also depend on the technical expertise of the research team. The use of the more advanced open-weights models requires technical knowledge such as familiarity with Python and relevant packages like PyTorch and TensorFlow, command line programming, and the ability to work with hardware such as HPCs, cloud computing environments, and GPUs. The Huggingface libraries make it relatively straightforward to interact with many open-weights models with some Python experience, but there is a steeper learning curve concerning the infrastructure. Social scientists who do not have such training may find it easier to adopt API-based solutions that require minimal programming to operate. However, we expect that this gap will lessen as more interactive solutions and more advanced open-source models emerge.

Optimizing predictive accuracy is the end goal in much of the research in computer science (Molina and Garip, 2019), but the development of an accurate classifier is typically only an intermediate goal in social scientific research, where the objective is to use the model for measurement (Grimmer et al., 2022). Even when we are satisfied that a model performs accurately and avoids problematic output, it is critical to emphasize that models cannot perfectly identify stances or any other attributes in texts. Indeed, human raters are inaccurate and inconsistent at detecting stances (Joseph et al., 2021), as demonstrated by our analyses of the Twitter dataset. Recent work highlights how biases in the classification process can introduce biases in downstream tasks such as regression. To address this issue, we recommend using bias-correction procedures that have been proposed when using output from classifiers, either as dependent or independent variables. These procedures work by adjusting the estimates using human-annotated data to correct for biases in the predictions from the classifier (Egami et al., 2023, 2024; TeBlunthuis et al., 2024). The combination of high-performance LLM classifiers and careful statistical adjustment procedures will allow social scientists to leverage machine learning classifiers to make valid measurements across a range of different domains.

7.3 Limitations and challenges

Beyond these technical and practical considerations, several limitations and challenges must be considered when using LLMs. We consider four issues and their implications for text classification: (1) interpretability, transparency, and reliability; (2) bias and bias mitigation; (3) reproducibility, and (4) privacy. We discuss each issue in turn but point interested readers to several recent works that

address these topics more comprehensively in the context of social scientific research (Bail, 2024; Ziems et al., 2024; Davidson, 2024) and related work that considers these and other challenges associated with these technologies (Weidinger et al., 2022; Bommasani et al., 2022).

7.3.1 Interpretability, transparency, and reliability

Machine learning models are often considered "black boxes" (Pasquale, 2015) that defy the kinds of interpretation social scientists typically seek from statistical models (Mullainathan and Spiess, 2017; Hofman et al., 2017; Davidson, 2019). Not only does the scale and complexity of LLMs make such interpretation even more challenging, but the lack of transparency regarding both open-weight and closed-source commercial models further complicates such efforts. We often do not know what data models were trained on, both in terms of the texts used for pre-training and any additional instructions provided for RLHF. As such, it is exceedingly difficult to explain why an input generates a given output, both with respect to the weights in the model and the data used to train it. We thus caution researchers against using these tools in domains where interpretability or explainability are important (Rudin, 2019). Reliability is also a related concern when using LLMs, as there is no guarantee that the outputs will be accurate. LLMs, particularly the larger models that have been instruction-tuned, are prone to generating plausible sounding but wrong responses, often referred to as "hallucinations." In our experiments, we found that smaller models would often generate responses outside of the permitted answers when used for zero- or few-shot learning, showing how prompts cannot always constrain the outputs to the required labels. While researchers must be aware of these limitations, we think these issues pose a greater threat to tasks where interpretability is critical and the reliability of the outputs is more difficult to measure. Since the objective of classification tasks is prediction rather than explanation, interpretability and transparency are not necessarily required, and the quality of the predictive model should always be evaluated using out-of-sample validation, such that reliability issues can readily be identified and measured.

7.3.2 Bias and bias mitigation

Bias in machine learning systems has been well-documented, as systems can learn stereotypical associations and patterns that are reproduced and reified (O'Neil, 2016; Noble, 2018). For example, 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). The fact that LLMs are trained using enormous

swathes of the internet multiplies the risk of biases (Bender et al., 2021; Bommasani et al., 2022; Weidinger et al., 2022). These problems extend to multimodal models trained on text and images, which often generate stereotypical representations of marginalized groups (Bianchi et al., 2023). Bias can have serious implications for the quality of downstream tasks when LLMs are used as foundational models. An audit of several BERT-based classifiers showed evidence of bias against members of marginalized groups and people with other stigmatized conditions or identities (Mei et al., 2023). Another study found that GPT-3 returned violent stereotypes casting Muslims as terrorists when completing innocuous prompts (Abid et al., 2021). Other work documents how political biases learned from training data can affect the performance of LLMs for hate speech and misinformation detection (Feng et al., 2023).

Various strategies have been used to address bias and other problematic behaviors in LLMs, from removing offensive material from training data (Raffel et al., 2020) to instructing models to refuse certain requests (OpenAI, 2023). While these efforts may make these models fairer, they can also be detrimental to sociological inquiry, where biases and stereotypical associations are the objects of interest (Argyle et al., 2023; Bail, 2024; Davidson, 2024). Researchers seeking to classify content that covers sensitive topics may thus face difficulties, particularly when using the most powerful instruction-tuned models, which have been adapted to avoid outputs related to various social issues. There is evidence that these models can often refuse completely innocuous queries (Röttger et al., 2024). In our evaluations, we found that zero- and few-shot classifiers sometimes refused to evaluate inputs containing contentious political claims. In some cases, it is possible to address this through prompting or instruction-tuning. In our thread-prediction task, we provided specific directions in the prompt to process offensive content and did not encounter any refusals. But in other cases, these bias mitigation procedures cannot be circumvented. We attempted to fine-tune GPT-4o mini, a version of the model released in July 2024, but received a message that the operation was blocked by the moderation system because the training data "contains too many examples that violate OpenAI's usage policies". This affected all of our datasets and was likely triggered by the political content and the presence of some offensive language in the texts. In this case, the company's mitigation efforts prevented what we consider to be a legitimate use of these tools.

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 specific tasks of interest to social scientists. We encourage practitioners to evaluate models to identify biases and scrutinize how

they could affect performance. One straightforward approach to measuring bias in text classification is to use templates to evaluate responses under different inputs (Dixon et al., 2018; Röttger et al., 2021). We also encourage researchers to compare multiple models to see how they fare on the same task and to conduct audits to assess predictive differences across demographic subgroups or other relevant variables to assess whether model performance varies in ways that could bias downstream analyses (e.g. Buolamwini and Gebru, 2018). If mitigation efforts impede legitimate research, we recommend the use of open-weights models that can be more readily modified, although instruction-tuned models can still sometimes refuse legitimate requests (Röttger et al., 2024).

7.3.3 Reproducibility

Due to the stochastic nature of neural network models and estimation procedures, it can be difficult to obtain reproducible results (Liu and Salganik, 2019), and this issue is magnified when considering the scale of LLMs. Moreover, commercial models can be modified or deprecated without notice, making it impossible for other researchers to reproduce results (Spirling, 2023). This is not merely a hypothetical. OpenAI withdrew a code-generation model used in hundreds of computer science publications from its API. Facing public criticism,[26] the company promised to continue to make the model available to researchers.[27] This could also impact our analysis because the two versions of GPT-3 used in our experiments have since been replaced by more advanced models. If perfect reproducibility is important, we concur with Spirling (2023) that researchers use open-weights models that can be stored on disk and re-used, rather than relying on commercial APIs that are subject to change.

7.3.4 Privacy

When using commercial models, the terms of service often mean that 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, as the data could theoretically be extracted from the model (Weidinger et al., 2022). In some jurisdictions, the use of commercial models may result in data sharing that violates data protection policies. For example, scholars in Europe have noted that this would amount to data sharing that violates the European Union's General Data Protection Regulation (GDPR) (Hacker et al., 2023).[28] In our case, it is unclear whether data provided to GPT-3 was used to train GPT-4, potentially compromising our Facebook comment evaluation data. If performing classification using

sensitive data, such as texts containing personally identifying information, we recommend using open-weight models on secure servers, such that data leakage can be prevented.

8 Conclusion

Overall, the recent advancements in LLMs will open up many new avenues for computational sociology. Growing public awareness of these technologies since the release of ChatGPT has accelerated their adoption for text analysis and many other tasks (Grossmann et al., 2023; Bail, 2024; Ziems et al., 2024; Davidson, 2024). Our analyses make several important contributions to research on the use of these tools for text classification. Consistent with prior work (Wankmüller, 2022; Do et al., 2022; Bonikowski et al., 2022; Ziems et al., 2024), we find that early generations of encoder-only models in the BERT family can demonstrate competitive performance when fine-tuned to perform classification tasks. However, large decoder-only models tend to achieve the best performance at predicting stances in social media posts. In the case of the comment prediction task, the best GPT-4o classifier used zero-shot learning. This underscores the potential of LLMs to make supervised text classification more accessible, as these models have the potential to work well with only a single prompt. Contrary to previous work that finds conversational data difficult to analyze (Ziems et al., 2024), our thread-prediction experiments demonstrate how instruction-tuning can be used to perform complex, structured forms of text classification. We anticipate that this technique will enable social scientists to transform rich coding schemes and theoretical frameworks into prompts and instructions that can be used to further adapt LLMs to analyze complex, multifaceted data. Taken together, LLMs provide sociologists with an inventory of tools to perform text classification with considerably greater accuracy and flexibility than more conventional machine learning techniques. We do not argue that one tool is better than the other but rather provide guidance to help researchers to identify the models and learning regimes that will be most appropriate for a given task.

The methods evaluated in this paper are sufficiently general that they can extend beyond stance detection to any supervised text classification task. These techniques can be used for other types of NLP tasks, like question-answering, named entity recognition, and summarization. Moreover, LLMs are increasingly becoming multimodal, trained on texts, images, and other media (Radford et al., 2021; OpenAI, 2023), so similar techniques can be extended to other modalities. These emerging technologies raise many new opportunities and challenges. This article is a step towards

understanding the strengths and weaknesses of different approaches to integrating large language models and other generative artificial intelligence into sociological research.

Notes

[1]The original model was released by Google in late 2018, and the paper has over 100,000 citations on Google Scholar as of July 2024.

[2]BERT was also trained using a next sentence prediction task, where the model was given a partial input sentence and had to select the correct second part of the sentence from several options (Devlin et al., 2019), but subsequent work showed this part of the training contributed little to the overall performance (Liu et al., 2019).

[3]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).

[4]This does not preclude the analyst from conducting preliminary data cleaning and manipulation before the texts are input into the model.

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

[6]We do not mean to imply that sociologists using sentiment analysis are unaware of its limitations. Some have taken steps to improve measurement quality. 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, a model trained to detect attitudes towards immigrants would provide a more direct measure of the quantity of interest.

[7]A keyword search for "stance detection" and "sentiment analysis" in *American Journal of Sociology*, *American Sociological Review*, *Social Forces*, *Socius*, *Sociological Methods & Research*, *Sociological Methodology* was performed in July 2024. Stance detection did not appear in any journals but sentiment analysis appeared twenty times, at least once in every journal queried.

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

[9]Stances inferred from social media are not necessarily equivalent to those obtained from surveys, nor can such approaches 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]The full dataset and information can be found here: `https://alt.qcri.org/semeval2016/task6/`

[11]The only exception to this is GPT-4. OpenAI has a policy that it no longer trains models using data passed through the API as of March 2023 (see `https://platform.openai.com/docs/models/how-we-use-your-data`), but some of our experiments were run in January 2023. As such, it is plausible that some data provided to OpenAI through our GPT-3 analyses could have been used to train GPT-4.

[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, and then predicting the stance. However, this is more cumbersome and costly to implement as it requires training and evaluating multiple classifiers.

[13]It is common for Facebook users to "tag" their friends in their comments and replies, meaning that the user's name is included in the text. The corpus was anonymized by first replacing all names with pseudonyms, ensuring that the names are constant within the thread (e.g. User A writes a comment, User B replies, then User A responds). Regular expressions were then used to identify any tags in the text and replace these with the relevant pseudonym.

[14]The models listed in monospace font correspond to either open-weights models available on Huggingface, which can be downloaded at `https://huggingface.co/models`, or names of models used by OpenAI's API.

[15]Since the completion of our analyses, these models have been deprecated. Ada has been replaced by `babbage-002` and Davinci by `gpt-3.5-turbo-instruct`.

[16]This number has been referenced in leaked materials and is cited in Ziems et al. (2024).

[17]Fine-tuning GPT-4o mini was made available via OpenAI's API in July 2024 but the new moderation policies blocked us from fine-tuning models on either dataset since the texts were considered to violate the usage policies.

[18]See Yin et al. (2019); Wankmüller (2022) for further discussion.

[19]We truncate any long Facebook comments to avoid excessive token consumption for GPT-3 Ada and GPT-3 Davinci. The mean comment length is 272 characters with a standard deviation of 604. We thus set the maximum allowed length to 876 characters (272 + 604), resulting in a total of 5 examples out of 100 being truncated.

[20]The endpoint we used for original experiments has been deprecated and replaced with a new API that is capable of chat-styled instruction-tuning. See `https://openai.com/index/gpt-3-5-turbo-fine-tuning-and-api-updates/`.

[21]We do not perform cross-validation on the training data because it is computationally, and sometimes financially, costly when fine-tuning LLMs since k-fold cross-validation requires k separate fine-tuning and prediction runs.

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

[23]`https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/`

[24]Assuming a corpus of 1M documents, each consisting of 100 words, GPT-3 consumes approximately 1000 tokens for every 7.5 documents ($\sim$100 tokens for 75 words), putting the total token consumption at approximately 133.3M. Based on OpenAI's cheapest GPT-4o pricing ($2.5 per 1M tokens), the text alone would cost around $333.25 to input. If used for zero- or few-shot learning, the prompts and exemplars would also add a considerable number of additional tokens. Outputs are charged at a higher rate ($7.5 per 1M tokens). In the simple Twitter example above, the output with the format `TARGET, STANCE` uses 5 tokens. Thus, a million generations will consume 5M tokens, costing an additional $37.50 and bringing the total cost to $370.75. Text generation tasks can consume substantially more tokens. For example, in thread-prediction analysis it takes up to 178 tokens to generate the output for for the longest threads, consuming 36 times as many tokens as the simple prediction task. Given that researchers may experiment with multiple models and learning regimes, including analyses for sensitivity to prompts and exemplars, the true training cost may end up being multiples of the cost to fine-tune and deploy the final model.

[25]For example, Lambda Labs charges $2.58 plus tax for the usage of two A100 GPUs, see `https://lambdalabs.com/service/gpu-cloud`.

[26]See `https://aisnakeoil.substack.com/p/openais-policies-hinder-reproducible`

[27]`https://twitter.com/OfficialLoganK/status/1638332437531971585`

[28]See also https://twitter.com/LeonDerczynski/status/1611390172087652352 for a concrete example.

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.

Ainslie, J., Lee-Thorp, J., de Jong, M., Zemlyanskiy, Y., Lebrón, F., and Sanghai, S. (2023). Gqa: Training generalized multi-query transformer models from multi-head checkpoints.

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.

Backstrom, L., Kleinberg, J., Lee, L., and Danescu-Niculescu-Mizil, C. (2013). Characterizing and curating conversation threads: expansion, focus, volume, re-entry. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 13–22. ACM.

Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2):179–190.

Bail, C. A. (2024). Can Generative AI improve social science? *Proceedings of the National Academy of Sciences*, 121(21):e2314021121. Publisher: Proceedings of the National Academy of Sciences.

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.

Berry, G. and Taylor, S. J. (2017). Discussion Quality Diffuses in the Digital Public Square. In *Proceedings of the 26th International Conference on World Wide Web - WWW '17*, pages 1371–1380, Perth, Australia. ACM Press.

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.

Bianchi, F., Kalluri, P., Durmus, E., Ladhak, F., Cheng, M., Nozza, D., Hashimoto, T., Jurafsky, D., Zou, J., and Caliskan, A. (2023). Easily Accessible Text-to-Image Generation Amplifies Demographic Stereotypes at Large Scale. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, pages 1493–1504, New York, NY, USA. Association for Computing Machinery.

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., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A Statistical Approach to Machine Translation. *Computational Linguistics*, 16(2):79–85.

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.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. (2022). Scaling instruction-finetuned language models.

Dagan, I., Glickman, O., and Magnini, B. (2006). The pascal recognising textual entailment challenge. In Quiñonero-Candela, J., Dagan, I., Magnini, B., and d'Alché Buc, F., editors, *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dai, A. M. and Le, Q. V. (2015). Semi-supervised Sequence Learning. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

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. (2019). Black-Box Models and Sociological Explanations: Predicting High School Grade Point Average Using Neural Networks. *Socius: Sociological Research for a Dynamic World*, 5:237802311881770.

Davidson, T. (2024). Start Generating: Harnessing Generative Artificial Intelligence for Sociological Research. *Socius: Sociological Research for a Dynamic World*, 10. Publisher: SAGE Publications.

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.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. (2023). Qlora: Efficient finetuning of quantized llms.

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.

Dixon, L., Li, J., Sorensen, J., Thain, N., and Vasserman, L. (2018). Measuring and Mitigating Unintended Bias in Text Classification. In *Proceedings of the 2018 Conference on AI, Ethics, and Society*, pages 67–73. ACM Press.

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.

Efron, B. and Tibshirani, R. (1986). Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*, 1(1):54–75.

Egami, N., Hinck, M., Stewart, B., and Wei, H. (2023). Using Imperfect Surrogates for Downstream Inference: Design-based Supervised Learning for Social Science Applications of Large Language Models. *Advances in Neural Information Processing Systems*, 36:68589–68601.

Egami, N., Hinck, M., Stewart, B. M., and Wei, H. (2024). Using Large Language Model Annotations for the Social Sciences: A General Framework of Using Predicted Variables in Downstream Analyses.

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.

Fu, Z., Lam, W., Yu, Q., So, A. M.-C., Hu, S., Liu, Z., and Collier, N. (2023). Decoder-only or encoder-decoder? interpreting language model as a regularized encoder-decoder.

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.

Gilardi, F., Alizadeh, M., and Kubli, M. (2023). ChatGPT outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120. Publisher: Proceedings of the National Academy of Sciences.

Gray, R. and Neuhoff, D. (1998). Quantization. *IEEE Transactions on Information Theory*, 44(6):2325–2383. Conference Name: IEEE Transactions on Information Theory.

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.

Grossmann, I., Feinberg, M., Parker, D. C., Christakis, N. A., Tetlock, P. E., and Cunningham, W. A. (2023). AI and the transformation of social science research. *Science*, 380(6650):1108–1109. Publisher: American Association for the Advancement of Science.

Hacker, P., Engel, A., and Mauer, M. (2023). Regulating ChatGPT and other Large Generative AI Models. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, pages 1112–1123, New York, NY, USA. Association for Computing Machinery.

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.

Heseltine, M. and Clemm von Hohenberg, B. (2024). Large language models as a substitute for human experts in annotating political text. *Research & Politics*, 11(1):20531680241236239. Publisher: SAGE Publications Ltd.

Hofman, J. M., Sharma, A., and Watts, D. J. (2017). Prediction and explanation in social systems. *Science*, 355(6324):486–488.

Howard, J. and Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685 [cs].

Ibrahim, E. I. and Voyer, A. (2024). The Augmented Qualitative Researcher: Using Generative AI in Qualitative Text Analysis.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. (2018). Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713.

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.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2023). Mistral 7b.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2024). Mixtral of Experts. arXiv:2401.04088 [cs].

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.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling Laws for Neural Language Models. arXiv:2001.08361 [cs, stat].

Khattab, O., Singhvi, A., Maheshwari, P., Zhang, Z., Santhanam, K., Vardhamanan, S., Haq, S., Sharma, A., Joshi, T. T., Moazam, H., Miller, H., Zaharia, M., and Potts, C. (2023). DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines. arXiv:2310.03714 [cs].

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.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. (2024). Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, pages 22199–22213, Red Hook, NY, USA. Curran Associates Inc.

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 000312241987713.

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.

Küçük, D. and Can, F. (2020). Stance Detection: A Survey. *ACM Comput. Surv.*, 53(1):12:1–12:37.

Laurer, M., van Atteveldt, W., Casas, A., and Welbers, K. (2024). Less annotating, more classifying: Addressing the data scarcity issue of supervised machine learning with deep transfer learning and bert-nli. *Political Analysis*, 32(1):84–100.

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.

Li, H. (2022). Language models: past, present, and future. *Communications of the ACM*, 65(7):56–63.

Liu, D. M. and Salganik, M. J. (2019). Successes and Struggles with Computational Reproducibility: Lessons from the Fragile Families Challenge. *Socius: Sociological Research for a Dynamic World*, 5:1–21.

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.

Mangrulkar, S., Gugger, S., Debut, L., Belkada, Y., Paul, S., and Bossan, B. (2022). Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Manning, C. D. (2022). Human Language Understanding & Reasoning. *Daedalus*, 151(2):127–138.

Martin, J. H. and Jurafsky, D. (2024). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, Upper Saddle River, NJ, 3rd edition.

Mei, K., Fereidooni, S., and Caliskan, A. (2023). Bias Against 93 Stigmatized Groups in Masked Language Models and Downstream Sentiment Classification Tasks. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, pages 1699–1710, New York, NY, USA. Association for Computing Machinery.

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.

Mohammad, S. M., Sobhani, P., and Kiritchenko, S. (2017). Stance and Sentiment in Tweets. *ACM Trans. Internet Technol.*, 17(3):26:1–26:23.

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

Mullainathan, S. and Spiess, J. (2017). Machine Learning: An Applied Econometric Approach. *Journal of Economic Perspectives*, 31(2):87–106.

Murakami, A. and Raymond, R. (2010). Support or Oppose? Classifying Positions in Online Debates from Reply Activities and Opinion Expressions. In Huang, C.-R. and Jurafsky, D., editors, *Coling 2010: Posters*, pages 869–875, Beijing, China. Coling 2010 Organizing Committee.

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.

Naab, T. K., Ruess, H.-S., and Küchler, C. (2023). The influence of the deliberative quality of user comments on the number and quality of their reply comments. *New Media & Society*, page 14614448231172168. Publisher: SAGE Publications.

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

Nelson, L. K., Brewer, A., Mueller, A. S., O'Connor, D. M., Dayal, A., and Arora, V. M. (2023). Taking the Time: The Implications of Workplace Assessment for Organizational Gender Inequality. *American Sociological Review*, 88(4):627–655.

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.

Noble, S. U. (2018). *Algorithms of Oppression: How Search Engines Reinforce Racism*. NYU Press, New York, NY.

O'Neil, C. (2016). *Weapons of math destruction: How big data increases inequality and threatens democracy*. Broadway Books.

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].

Palmer, A., Smith, N. A., and Spirling, A. (2024). Using proprietary language models in academic research requires explicit justification. *Nature Computational Science*, 4(1):2–3. Publisher: Nature Publishing Group.

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

Pasquale, F. (2015). *The black box society*. Harvard University Press, Cambridge, MA.

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.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. Technical report, OpenAI.

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].

Rathje, S., Mirea, D.-M., Sucholutsky, I., Marjieh, R., Robertson, C., and Bavel, J. J. V. (2023). GPT is an effective tool for multilingual psychological text analysis. Publisher: OSF.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks.

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.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Röttger, P., Kirk, H., Vidgen, B., Attanasio, G., Bianchi, F., and Hovy, D. (2024). XSTest: A Test Suite for Identifying Exaggerated Safety Behaviours in Large Language Models. In Duh, K., Gomez, H., and Bethard, S., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.

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.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423. Publisher: Nokia Bell Labs.

Shazeer, N. (2020). Glu variants improve transformer.

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.

Shugars, S. and Beauchamp, N. (2019). Why Keep Arguing? Predicting Engagement in Political Conversations Online. *Sage Open*, 9(1):2158244019828850. Publisher: SAGE Publications.

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

Sobhani, P., Inkpen, D., and Matwin, S. (2015). From argumentation mining to stance classification. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 67–77.

Sobhani, P., Inkpen, D., and Zhu, X. (2017). A Dataset for Multi-Target Stance Detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 551–557, Valencia, Spain. Association for Computational Linguistics.

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.

Sridhar, D., Foulds, J., Huang, B., Getoor, L., and Walker, M. (2015). Joint Models of Disagreement and Stance in Online Debate. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 116–125, Beijing, China. Association for Computational Linguistics.

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

Strubell, E., Ganesh, A., and McCallum, A. (2019). Energy and Policy Considerations for Deep Learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., and Liu, Y. (2023). Roformer: Enhanced transformer with rotary position embedding.

Sun, C., Qiu, X., Xu, Y., and Huang, X. (2020). How to fine-tune bert for text classification?

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. (2023). Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

TeBlunthuis, N., Hase, V., and Chan, C.-H. (2024). Misclassification in Automated Content Analysis Causes Bias in Regression. Can We Fix It? Yes We Can! *Communication Methods and Measures*, 0(0):1–22. Publisher: Routledge _eprint: https://doi.org/10.1080/19312458.2023.2293713.

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.

Walker, M., Anand, P., Abbott, R., and Grant, R. (2012). Stance Classification using Dialogic Properties of Persuasion. In Fosler-Lussier, E., Riloff, E., and Bangalore, S., editors, *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 592–596, Montréal, Canada. Association for Computational Linguistics.

Wang, Y., Kordi, Y., Mishra, S., Liu, A., Smith, N. A., Khashabi, D., and Hajishirzi, H. (2023). Self-Instruct: Aligning Language Models with Self-Generated Instructions. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

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., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. (2022). Finetuned Language Models are Zero-Shot Learners. In *International Conference on Learning Representations*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. (2023a). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903 [cs].

Wei, J., Wei, J., Tay, Y., Tran, D., Webson, A., Lu, Y., Chen, X., Liu, H., Huang, D., Zhou, D., and Ma, T. (2023b). Larger language models do in-context learning differently. arXiv:2303.03846 [cs].

Weidinger, L., Uesato, J., Rauh, M., Griffin, C., Huang, P.-S., Mellor, J., Glaese, A., Cheng, M., Balle, B., Kasirzadeh, A., Biles, C., Brown, S., Kenton, Z., Hawkins, W., Stepleton, T., Birhane, A., Hendricks, L. A., Rimell, L., Isaac, W., Haas, J., Legassick, S., Irving, G., and Gabriel, I. (2022). Taxonomy of Risks posed by Language Models. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 214–229, Seoul Republic of Korea. ACM.

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. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3912–3921, Hong Kong, China. Association for Computational Linguistics.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3320–3328, Cambridge, MA, USA. MIT Press.

Yuksekgonul, M., Bianchi, F., Boen, J., Liu, S., Huang, Z., Guestrin, C., and Zou, J. (2024). TextGrad: Automatic "Differentiation" via Text. arXiv:2406.07496 [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.

Zhang, J., Chang, J., Danescu-Niculescu-Mizil, C., Dixon, L., Hua, Y., Taraborelli, D., and Thain, N. (2018). Conversations Gone Awry: Detecting Early Signs of Conversational Failure. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1350–1361, Melbourne, Australia. Association for Computational Linguistics.

Zhao, Z., Wallace, E., Feng, S., Klein, D., and Singh, S. (2021). Calibrate before use: Improving few-shot performance of language models. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., and Ba, J. (2022). Large Language Models are Human-Level Prompt Engineers. In *The Eleventh International Conference on Learning Representations*.

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.

Ziems, C., Held, W., Shaikh, O., Chen, J., Zhang, Z., and Yang, D. (2024). Can Large Language Models Transform Computational Social Science? *Computational Linguistics*, pages 1–55.

Zubiaga, A., Kochkina, E., Liakata, M., Procter, R., and Lukasik, M. (2016). Stance Classification in Rumours as a Sequential Task Exploiting the Tree Structure of Social Media Conversations. In Matsumoto, Y. and Prasad, R., editors, *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2438–2448, Osaka, Japan. The COLING 2016 Organizing Committee.

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, 2024). 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, except for 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 20 epochs, respectively, in batches of 16 documents.

We estimate all four models for 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 comment tasks are reported at the bottom of Table A5 and Table A6, respectively. In general, we find that the models with BoW features outperform the embedding representations when evaluated using the strictest joint 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, both models achieve comparable performance using the BoW features for the Twitter task, but the SVM performs better on the Facebook comments.

A.3 Hyperparameters, fine-tuning specifications, and compute infrastructure

One challenge in comparing model performance across different families of models is to set up as consistent hyperparameters and fine-tuning environments as possible. Since the size and architecture of the first and second model families significantly differ, we had to apply slightly different fine-tuning setups. For the encoder-only models, we modify two key parameters. *Batch size* controls how many training examples are used for each gradient update. Although a batch size of 16 or 32 is common when fine-tuning BERT, we chose the batch size of 4 to make the fine-tuning environment for the first and second model families more comparable. Therefore, we applied a relatively small batch size of 4 for the first model family as we did for the second set of models. *Learning rate*

controls how much the model updates its parameters each iteration. We used a learning rate of 2e-5 that Sun et al. (2020) suggests as necessary to help the BERT model converge during fine-tuning.

For the second set of models, we applied a batch size of 1 and performed gradient updates at every fourth batch, averaging loss functions from four consecutive batches, such that our effective batch size is 4 while using an actual batch size of 1. Without this technique, larger models like Llama3-70B consume more GPU memory than consumer-level GPUs can offer. For memory efficiency, we also deployed QLoRA (Dettmers et al., 2023) that combines two separate techniques: 4-bit parameter quantization and Parameter-Efficient Fine-Tuning (PEFT) (Mangrulkar et al., 2022) with Low Rank Adapter (LoRA) (Hu et al., 2021). Quantization refers to techniques to cast model parameters, usually in float32 or bfloat16, to a less precise data type such as int4 (Gray and Neuhoff, 1998). In practice, this means that weights are converted from decimals to integers. This significantly reduces the memory used by the model and speeds up inference with minimal impact on accuracy (Jacob et al., 2018). PEFT is based on the well-known insight underlying techniques like Singular Value Decomposition that large matrices can be approximated by low-rank matrices without much information loss. LLMs can thus be fine-tuned by only a small portion of the parameters (typically less than 1 percent). For our fine-tuning experiments, we used hyperparameters suggested in the QLoRA paper (Dettmers et al., 2023): a learning rate of 2e-4, a dropout of 0.05 (0.1 for Llama3-70B), alpha = 16, and gamma = 64. Due to constraints on hardware availability, we were unable to systematically explore how variation in these parameters impacted performance, although we ran ad-hoc experiments with Llama3-8B that showed quantization had no significant impact on predictive performance.

Each GPT 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.[29] Other hyperparameters are set to defaults used by OpenAI.

Our experiments were implemented using a MacBook Pro laptop with an M1 Max 32-core GPU and 64GB of RAM for the first, encoder-only model family, and NVidia RTX 3090-24GB and A100-40GB GPUs provided by [University anonymized for peer-review] for the second, medium to large encoder-decoder and decoder-only model family. Fine-tuning Llama3-70B, the largest among the open-weights models in this study, was distributed across two GPUs and took approximately 10

hours. Except for the GPT experiments, which were completed using the OpenAI API, all other analyses were completed using the Huggingface libraries (Wolf et al., 2020). Our experiments with GPT-3 models were self-funded (costing approximately $1500) and our GPT-4o analyses were funded by credits provided through OpenAI's Researcher Access Program (costing less than $100 in credits since we did not perform fine-tuning).

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 A3: **Regression models predicting target-specific F1 score for Facebook one-shot experiments**
This table shows the regression coefficients from six models predicting target-specific F1 scores using the length of an exemplar, the stances in the exemplar towards both targets, and the stance interactions. The reference category of the target-specific stances is Neutral/None.

	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 A4: **Regression models predicting target-specific F1 score for Twitter two-shot experiments**

This table shows the regression coefficients from six models predicting target-specific F1 scores using the length of target-specific exemplars, the stance of the exemplar towards each target, and the stance interactions. The reference category of the target-specific stances is Neutral/None.

A.5 Full results

Model	Type	$F1_T$	$F1_S$	$F1_J$	R_T	R_S	R_J	P_T	P_S	P_J
BERT	10	0.46	0.24	0.14	0.5	0.29	0.21	0.6	0.31	0.15
	100	0.43	0.36	0.18	0.58	0.53	0.35	0.34	0.28	0.12
	All	0.85	0.65	0.6	0.85	0.65	0.6	0.85	0.65	0.61
SBERT	10	0.5	0.39	0.21	0.59	0.5	0.34	0.57	0.36	0.28
	100	0.43	0.36	0.18	0.58	0.53	0.35	0.34	0.28	0.12
	All	0.87	0.72	0.65	0.87	0.72	0.65	0.87	0.72	0.66
DeBERTa	0	0.59	0.6	0.44	0.63	0.6	0.43	0.62	0.68	0.54
	10	0.25	0.36	0.05	0.42	0.53	0.18	0.18	0.28	0.03
	100	0.43	0.36	0.18	0.58	0.53	0.35	0.34	0.28	0.12
	All	0.87	0.66	0.57	0.87	0.67	0.6	0.87	0.67	0.61
Flan-T5 XXL	0	0	0.57	0	0	0.61	0	0	0.63	0
	2	0.01	0.58	0.01	0	0.63	0	0.21	0.65	0.04
	10	0.25	0.07	0.01	0.42	0.21	0.08	0.18	0.04	0.01
	100	0.43	0.36	0.18	0.58	0.53	0.35	0.34	0.28	0.12
	All	0.77	0.53	0.41	0.78	0.61	0.49	0.79	0.5	0.38
Mistral-7B	0	0.76	0.63	0.48	0.7	0.64	0.48	0.85	0.66	0.55
	2	0.77	0.62	0.52	0.72	0.63	0.52	0.84	0.64	0.58
	10	0.45	0.28	0.15	0.49	0.29	0.18	0.59	0.35	0.23
	100	0.47	0.48	0.26	0.57	0.5	0.33	0.52	0.48	0.27
	All	0.82	0.64	0.56	0.82	0.64	0.56	0.82	0.64	0.57
Llama3-8B	0	0.69	0.59	0.43	0.68	0.62	0.44	0.83	0.67	0.54
	2	0.76	0.63	0.49	0.71	0.67	0.51	0.87	0.73	0.61
	10	0.58	0.45	0.25	0.59	0.45	0.3	0.58	0.44	0.23
	100	0.44	0.55	0.28	0.59	0.58	0.37	0.76	0.57	0.32
	All	0.84	0.75	0.67	0.84	0.75	0.66	0.85	0.76	0.68
Llama3-70B	0	0.77	0.64	0.52	0.7	0.66	0.53	0.86	0.68	0.56
	2	0.78	0.68	0.54	0.73	0.7	0.56	0.84	0.71	0.59
	10	0.56	0.37	0.22	0.56	0.4	0.27	0.57	0.36	0.22
	100	0.44	0.5	0.25	0.58	0.55	0.35	0.55	0.54	0.39
	All	0.81	0.73	0.62	0.81	0.73	0.61	0.81	0.73	0.62
GPT-3 Ada	0	0.26	0.04	0	0.23	0.02	0	0.73	0.46	0
	2	0.06	0.11	0.03	0.04	0.06	0.02	0.72	0.34	0.38
	10	0.81	0.44	0.3	0.81	0.5	0.41	0.83	0.43	0.25
	100	0.82	0.58	0.46	0.82	0.59	0.48	0.82	0.58	0.47
	All	0.88	0.73	0.65	0.88	0.73	0.65	0.88	0.73	0.66
GPT-3 Davinci	0	0.75	0.7	0.49	0.64	0.7	0.48	0.89	0.72	0.58
	2	0.77	0.68	0.54	0.66	0.68	0.5	0.92	0.7	0.66
	10	0.78	0.43	0.3	0.76	0.47	0.35	0.8	0.45	0.38
	100	0.84	0.69	0.61	0.84	0.69	0.61	0.85	0.69	0.63
	All	**0.89**	**0.76**	**0.69**	**0.88**	**0.76**	**0.69**	0.89	0.76	**0.7**
GPT-4o	0	0.75	0.72	0.53	0.63	0.72	0.48	**0.94**	0.78	0.62
	2	0.76	0.73	0.53	0.64	0.73	0.49	**0.94**	**0.79**	0.63
SVM + BoW	All	0.80	0.56	0.48	0.81	0.57	0.49	0.81	0.56	0.49
SVM + GloVe	All	0.71	0.48	0.34	0.72	0.53	0.40	0.72	0.48	0.33
CNN + BoW	All	0.78	0.59	0.48	0.78	0.59	0.48	0.80	0.59	0.49
CNN + GloVe	All	0.66	0.53	0.38	0.66	0.54	0.38	0.69	0.54	0.42

Table A5: **Twitter results.**
This table shows the held-out scores for each model and learning regime. The scores are split into groups: F1 scores for Target (T), Stance (S), and the joint score (J); Recall scores; and Precision scores. The best model by each metric is shown in bold.

Model	Type	$F1_C$	$F1_T$	$F1_J$	R_C	R_T	R_J	P_C	P_T	P_J
BERT	10	0.38	0.47	0.14	0.4	0.47	0.16	0.41	0.49	0.16
	100	0.42	0.52	0.22	0.44	0.66	0.31	0.54	0.43	0.2
	1000	0.83	0.82	0.7	0.84	0.82	0.72	0.83	0.81	0.69
	All	0.85	0.84	0.73	0.86	0.84	0.74	0.85	0.85	0.73
SBERT	10	0.44	0.52	0.15	0.58	0.66	0.31	0.35	0.68	0.1
	100	0.35	0.52	0.26	0.42	0.66	0.36	0.65	0.43	0.32
	1000	0.84	0.83	0.72	0.84	0.84	0.74	0.84	0.84	0.72
	All	0.85	0.84	0.74	0.86	0.84	0.76	0.86	0.85	0.74
DeBERTa	0	0.44	0.68	0.32	0.43	0.67	0.3	0.69	0.73	0.61
	10	0.44	0.49	0.15	0.59	0.57	0.28	0.35	0.43	0.1
	100	0.53	0.52	0.29	0.56	0.66	0.4	0.58	0.43	0.24
	1000	0.82	0.86	0.71	0.82	0.86	0.73	0.82	0.87	0.71
	All	0.85	0.86	0.74	0.85	0.86	0.75	0.85	0.87	0.74
Flan-T5 XXL	0	0.2	0.26	0.02	0.23	0.34	0.04	0.18	0.81	0.02
	1	0.24	0.33	0.02	0.32	0.38	0.05	0.19	0.82	0.02
	10	0.44	0.52	0.15	0.59	0.66	0.32	0.35	0.43	0.1
	100	0.37	0.52	0.22	0.39	0.66	0.3	0.44	0.43	0.19
	1000	0.62	0.69	0.46	0.69	0.73	0.52	0.64	0.65	0.46
	All	0.75	0.75	0.58	0.78	0.79	0.64	0.77	0.71	0.56
Mistral-7B	0	0.67	0.54	0.39	0.66	0.51	0.35	0.7	0.82	0.71
	1	0.67	0.59	0.42	0.66	0.56	0.38	0.71	0.77	0.67
	10	0.2	0.45	0.08	0.23	0.43	0.09	0.39	0.51	0.13
	100	0.5	0.56	0.26	0.49	0.6	0.29	0.55	0.57	0.24
	1000	0.73	0.73	0.56	0.73	0.73	0.56	0.74	0.73	0.57
	All	0.77	0.77	0.61	0.77	0.77	0.61	0.77	0.77	0.61
Llama3-8B	0	0.78	0.7	0.59	0.77	0.66	0.54	0.81	0.83	0.75
	1	0.78	0.53	0.45	0.77	0.5	0.41	0.81	0.86	0.77
	10	0.45	0.4	0.12	0.45	0.41	0.16	0.46	0.44	0.15
	100	0.52	0.54	0.3	0.5	0.62	0.34	0.6	0.55	0.28
	1000	0.79	0.8	0.66	0.79	0.81	0.67	0.79	0.8	0.67
	All	0.83	0.83	0.69	0.83	0.84	0.7	0.83	0.83	0.69
Llama3-70B	0	0.79	0.78	0.66	0.78	0.76	0.64	0.82	0.85	0.76
	1	0.82	0.81	0.69	0.81	0.8	0.68	0.84	0.87	0.79
	10	0.47	0.53	0.15	0.54	0.63	0.27	0.48	0.48	0.11
	100	0.51	0.52	0.35	0.5	0.63	0.4	0.61	0.45	0.33
	1000	0.79	0.8	0.64	0.79	0.81	0.65	0.79	0.8	0.64
	All	0.82	0.84	0.69	0.82	0.84	0.7	0.82	0.84	0.7
GPT-3 Ada	0	0.05	0.11	0	0.04	0.24	0	0.09	0.08	0
	1	0.05	0.07	0	0.03	0.07	0	0.46	0.43	0
	10	0.46	0.52	0.18	0.58	0.66	0.33	0.38	0.43	0.12
	100	0.71	0.73	0.55	0.74	0.73	0.57	0.74	0.73	0.56
	1000	0.84	0.82	0.71	0.84	0.82	0.71	0.84	0.82	0.72
	All	0.86	0.82	0.73	0.86	0.82	0.73	0.87	0.82	0.74
GPT-3 Davinci	0	0.72	0.77	0.61	0.72	0.76	0.56	0.79	0.84	0.73
	1	0.73	0.64	0.51	0.73	0.62	0.45	0.8	0.83	0.74
	10	0.45	0.51	0.15	0.58	0.64	0.29	0.37	0.43	0.1
	100	0.8	0.84	0.67	0.81	0.84	0.69	0.81	0.84	0.66
	1000	0.88	0.89	0.8	0.88	0.89	0.8	0.89	0.89	0.8
	All	**0.91**	0.9	0.83	**0.91**	0.9	0.83	**0.91**	0.9	0.83
GPT-4o	0	**0.91**	0.9	0.83	**0.91**	**0.91**	**0.84**	**0.91**	**0.91**	**0.85**
	1	**0.91**	**0.91**	**0.84**	**0.91**	**0.91**	**0.84**	**0.91**	**0.91**	**0.85**
SVM + BoW	All	0.71	0.76	0.54	0.72	0.78	0.55	0.71	0.75	0.55
SVM + GloVe	All	0.60	0.64	0.39	0.65	0.69	0.43	0.58	0.61	0.38
CNN + BoW	All	0.67	0.67	0.46	0.70	0.67	0.49	0.68	0.69	0.48
CNN + GloVe	All	0.57	0.61	0.38	0.65	0.62	0.44	0.68	0.67	0.47

Table A6: **Facebook results.**
This table shows the held-out scores for each model and learning regime. The scores are split into three groups: F1 scores for Clinton (C), Trump (T), and the joint score (J); Recall scores; and Precision scores. The best model by each metric is shown in bold.

Length	Position	Model	$F1_C$	$F1_T$	$F1_J$	R_C	R_T	R_J	P_C	P_T	P_J
All	All	Llama3-8B zero-shot	0.74	0.60	0.48	0.73	0.61	0.47	0.79	0.71	0.59
		Llama3-8B instruction-tuned	0.86	0.86	0.75	0.86	0.85	0.75	0.87	0.86	0.77
		Llama3-70B zero-shot	0.85	0.86	0.74	0.85	0.86	0.74	0.85	0.86	0.74
		Llama3-70B instruction-tuned	**0.89**	**0.90**	**0.82**	**0.89**	**0.90**	**0.82**	**0.90**	**0.90**	**0.83**
		GPT-4o zero-shot	0.85	0.85	0.73	0.85	0.84	0.73	0.87	0.85	0.76
		GPT-4o baseline	0.83	0.77	0.64	0.83	0.76	0.66	0.84	0.77	0.70
0	0	Llama3-8B zero-shot	0.81	0.58	0.45	0.80	0.62	0.46	0.86	0.75	0.55
		Llama3-8B instruction-tuned	0.96	0.88	0.84	0.96	0.88	0.84	0.96	0.90	0.87
		Llama3-70B zero-shot	0.88	0.82	0.76	0.88	0.82	0.74	0.89	0.83	0.80
		Llama3-70B instruction-tuned	0.96	0.92	0.89	0.96	0.92	0.90	0.97	0.92	0.92
		GPT-4o zero-shot	0.83	0.69	0.58	0.82	0.70	0.54	0.89	0.71	0.74
		GPT-4o baseline	0.76	0.61	0.42	0.76	0.60	0.42	0.81	0.62	0.60
1	All	Llama3-8B zero-shot	0.81	0.65	0.56	0.81	0.64	0.57	0.83	0.73	0.65
		Llama3-8B instruction-tuned	0.92	0.87	0.83	0.92	0.87	0.83	0.93	0.87	0.83
		Llama3-70B zero-shot	0.85	0.86	0.75	0.85	0.86	0.75	0.85	0.87	0.76
		Llama3-70B instruction-tuned	0.95	0.92	0.90	0.95	0.92	0.90	0.95	0.92	0.91
		GPT-4o zero-shot	0.88	0.85	0.77	0.88	0.85	0.77	0.89	0.87	0.81
		GPT-4o baseline	0.78	0.79	0.61	0.78	0.78	0.63	0.78	0.81	0.65
	0	Llama3-8B zero-shot	0.81	0.66	0.60	0.82	0.66	0.62	0.85	0.75	0.72
		Llama3-8B instruction-tuned	0.92	0.84	0.82	0.92	0.84	0.82	0.93	0.85	0.85
		Llama3-70B zero-shot	0.92	0.86	0.83	0.92	0.86	0.82	0.93	0.88	0.85
		Llama3-70B instruction-tuned	0.96	0.86	0.86	0.96	0.86	0.86	0.96	0.86	0.87
		GPT-4o zero-shot	0.88	0.88	0.81	0.88	0.88	0.80	0.90	0.90	0.87
		GPT-4o baseline	0.77	0.78	0.57	0.78	0.78	0.62	0.78	0.83	0.54
	1	Llama3-8B zero-shot	0.80	0.63	0.53	0.80	0.62	0.52	0.82	0.71	0.65
		Llama3-8B instruction-tuned	0.92	0.90	0.84	0.92	0.90	0.84	0.94	0.90	0.86
		Llama3-70B zero-shot	0.77	0.86	0.66	0.78	0.86	0.68	0.78	0.86	0.66
		Llama3-70B instruction-tuned	0.94	0.98	0.94	0.94	0.98	0.94	0.95	0.98	0.95
		GPT-4o zero-shot	0.88	0.82	0.73	0.88	0.82	0.74	0.88	0.84	0.76
		GPT-4o baseline	0.78	0.77	0.63	0.78	0.78	0.64	0.78	0.79	0.66
2	All	Llama3-8B zero-shot	0.72	0.63	0.45	0.70	0.63	0.43	0.80	0.75	0.66
		Llama3-8B instruction-tuned	0.81	0.87	0.73	0.81	0.87	0.73	0.83	0.88	0.76
		Llama3-70B zero-shot	0.87	0.88	0.79	0.86	0.88	0.78	0.88	0.88	0.81
		Llama3-70B instruction-tuned	0.85	0.91	0.79	0.85	0.91	0.79	0.87	0.91	0.82
		GPT-4o zero-shot	0.87	0.89	0.78	0.87	0.89	0.77	0.87	0.89	0.80
		GPT-4o baseline	0.84	0.80	0.67	0.84	0.79	0.67	0.85	0.80	0.74
	0	Llama3-8B zero-shot	0.82	0.58	0.52	0.82	0.58	0.52	0.84	0.69	0.68
		Llama3-8B instruction-tuned	0.82	0.84	0.72	0.82	0.84	0.72	0.82	0.84	0.73
		Llama3-70B zero-shot	0.86	0.82	0.73	0.86	0.82	0.74	0.86	0.83	0.79
		Llama3-70B instruction-tuned	0.88	0.90	0.82	0.88	0.90	0.82	0.88	0.90	0.83
		GPT-4o zero-shot	0.88	0.92	0.80	0.88	0.92	0.82	0.88	0.92	0.85
		GPT-4o baseline	0.90	0.84	0.76	0.90	0.84	0.76	0.90	0.85	0.82
	1	Llama3-8B zero-shot	0.71	0.68	0.43	0.68	0.68	0.42	0.77	0.80	0.70
		Llama3-8B instruction-tuned	0.77	0.84	0.72	0.74	0.84	0.68	0.82	0.85	0.78
		Llama3-70B zero-shot	0.83	0.92	0.77	0.82	0.92	0.76	0.85	0.92	0.81
		Llama3-70B instruction-tuned	0.84	0.94	0.83	0.82	0.94	0.80	0.87	0.94	0.88
		GPT-4o zero-shot	0.83	0.88	0.74	0.82	0.88	0.72	0.83	0.88	0.80
		GPT-4o baseline	0.81	0.76	0.60	0.80	0.76	0.60	0.81	0.76	0.73
	2	Llama3-8B zero-shot	0.66	0.62	0.42	0.60	0.62	0.36	0.84	0.78	0.76
		Llama3-8B instruction-tuned	0.87	0.92	0.79	0.86	0.92	0.78	0.90	0.94	0.86
		Llama3-70B zero-shot	0.91	0.90	0.85	0.90	0.90	0.84	0.93	0.91	0.89
		Llama3-70B instruction-tuned	0.85	0.88	0.74	0.84	0.88	0.74	0.89	0.90	0.82
		GPT-4o zero-shot	0.90	0.86	0.77	0.90	0.86	0.78	0.92	0.89	0.84
		GPT-4o baseline	0.83	0.79	0.66	0.82	0.78	0.66	0.85	0.81	0.74
3	All	Llama3-8B zero-shot	0.70	0.55	0.44	0.70	0.56	0.44	0.76	0.65	0.55
		Llama3-8B instruction-tuned	0.87	0.88	0.77	0.87	0.88	0.78	0.89	0.88	0.79
		Llama3-70B zero-shot	0.87	0.88	0.78	0.87	0.88	0.78	0.88	0.88	0.80
		Llama3-70B instruction-tuned	0.90	0.90	0.83	0.90	0.90	0.83	0.90	0.91	0.84

(continued)

Length	Position	Model	$F1_C$	$F1_T$	$F1_J$	R_C	R_T	R_J	P_C	P_T	P_J
		GPT-4o zero-shot	0.87	0.88	0.78	0.87	0.88	0.78	0.89	0.88	0.80
		GPT-4o baseline	0.86	0.77	0.68	0.86	0.77	0.69	0.86	0.77	0.71
	0	Llama3-8B zero-shot	0.68	0.66	0.54	0.68	0.68	0.52	0.73	0.78	0.69
		Llama3-8B instruction-tuned	0.88	0.90	0.84	0.88	0.90	0.84	0.90	0.90	0.86
		Llama3-70B zero-shot	0.82	0.92	0.79	0.82	0.92	0.78	0.86	0.92	0.84
		Llama3-70B instruction-tuned	0.90	0.96	0.88	0.90	0.96	0.88	0.91	0.96	0.91
		GPT-4o zero-shot	0.88	0.88	0.81	0.88	0.88	0.80	0.90	0.89	0.84
		GPT-4o baseline	0.86	0.82	0.72	0.86	0.82	0.74	0.87	0.83	0.75
	1	Llama3-8B zero-shot	0.76	0.54	0.43	0.76	0.54	0.46	0.77	0.57	0.53
		Llama3-8B instruction-tuned	0.88	0.88	0.78	0.88	0.88	0.78	0.89	0.88	0.81
		Llama3-70B zero-shot	0.94	0.84	0.81	0.94	0.84	0.82	0.94	0.85	0.82
		Llama3-70B instruction-tuned	0.90	0.88	0.81	0.90	0.88	0.82	0.90	0.88	0.81
		GPT-4o zero-shot	0.86	0.90	0.77	0.86	0.90	0.76	0.88	0.91	0.79
		GPT-4o baseline	0.87	0.78	0.69	0.88	0.78	0.72	0.90	0.79	0.72
	2	Llama3-8B zero-shot	0.69	0.51	0.41	0.68	0.50	0.36	0.80	0.67	0.62
		Llama3-8B instruction-tuned	0.87	0.80	0.71	0.86	0.80	0.70	0.90	0.82	0.75
		Llama3-70B zero-shot	0.90	0.88	0.80	0.90	0.88	0.80	0.91	0.89	0.81
		Llama3-70B instruction-tuned	0.92	0.88	0.84	0.92	0.88	0.84	0.94	0.89	0.87
		GPT-4o zero-shot	0.88	0.88	0.80	0.88	0.88	0.78	0.91	0.89	0.86
		GPT-4o baseline	0.88	0.69	0.64	0.88	0.68	0.64	0.88	0.70	0.66
	3	Llama3-8B zero-shot	0.69	0.51	0.41	0.66	0.52	0.40	0.82	0.67	0.61
		Llama3-8B instruction-tuned	0.87	0.92	0.79	0.86	0.92	0.78	0.88	0.92	0.81
		Llama3-70B zero-shot	0.84	0.88	0.75	0.82	0.88	0.74	0.87	0.88	0.80
		Llama3-70B instruction-tuned	0.87	0.90	0.79	0.86	0.90	0.78	0.90	0.90	0.82
		GPT-4o zero-shot	0.87	0.86	0.76	0.86	0.86	0.76	0.90	0.87	0.80
		GPT-4o baseline	0.83	0.80	0.65	0.82	0.80	0.66	0.86	0.83	0.74
4	All	Llama3-8B zero-shot	0.75	0.54	0.46	0.74	0.54	0.45	0.79	0.70	0.61
		Llama3-8B instruction-tuned	0.89	0.81	0.74	0.89	0.80	0.73	0.89	0.83	0.76
		Llama3-70B zero-shot	0.85	0.82	0.71	0.85	0.82	0.71	0.85	0.83	0.71
		Llama3-70B instruction-tuned	0.92	0.87	0.81	0.92	0.87	0.81	0.92	0.88	0.82
		GPT-4o zero-shot	0.89	0.79	0.70	0.88	0.79	0.70	0.89	0.82	0.75
		GPT-4o baseline	0.85	0.77	0.65	0.85	0.76	0.66	0.85	0.78	0.73
	0	Llama3-8B zero-shot	0.82	0.66	0.59	0.82	0.66	0.58	0.85	0.71	0.68
		Llama3-8B instruction-tuned	0.90	0.86	0.78	0.90	0.86	0.78	0.91	0.88	0.81
		Llama3-70B zero-shot	0.88	0.94	0.82	0.88	0.94	0.82	0.88	0.95	0.85
		Llama3-70B instruction-tuned	0.92	0.94	0.86	0.92	0.94	0.86	0.93	0.95	0.89
		GPT-4o zero-shot	0.90	0.83	0.74	0.90	0.82	0.74	0.91	0.88	0.83
		GPT-4o baseline	0.90	0.79	0.69	0.90	0.80	0.72	0.91	0.84	0.81
	1	Llama3-8B zero-shot	0.76	0.58	0.45	0.76	0.58	0.46	0.77	0.63	0.49
		Llama3-8B instruction-tuned	0.86	0.84	0.72	0.86	0.84	0.72	0.86	0.84	0.73
		Llama3-70B zero-shot	0.79	0.78	0.63	0.80	0.78	0.62	0.81	0.78	0.67
		Llama3-70B instruction-tuned	0.86	0.86	0.75	0.86	0.86	0.74	0.86	0.87	0.79
		GPT-4o zero-shot	0.80	0.78	0.61	0.80	0.78	0.60	0.80	0.79	0.63
		GPT-4o baseline	0.77	0.78	0.66	0.78	0.78	0.66	0.78	0.79	0.73
	2	Llama3-8B zero-shot	0.73	0.42	0.40	0.72	0.42	0.36	0.79	0.68	0.60
		Llama3-8B instruction-tuned	0.86	0.72	0.68	0.86	0.70	0.64	0.86	0.77	0.73
		Llama3-70B zero-shot	0.82	0.69	0.60	0.82	0.68	0.58	0.83	0.73	0.63
		Llama3-70B instruction-tuned	0.98	0.83	0.81	0.98	0.82	0.80	0.98	0.85	0.86
		GPT-4o zero-shot	0.92	0.72	0.69	0.92	0.70	0.64	0.94	0.80	0.79
		GPT-4o baseline	0.86	0.75	0.66	0.86	0.74	0.66	0.87	0.76	0.68
	3	Llama3-8B zero-shot	0.72	0.51	0.47	0.70	0.52	0.44	0.80	0.74	0.68
		Llama3-8B instruction-tuned	0.88	0.85	0.80	0.88	0.84	0.78	0.90	0.87	0.84
		Llama3-70B zero-shot	0.86	0.84	0.75	0.86	0.84	0.74	0.87	0.85	0.76
		Llama3-70B instruction-tuned	0.92	0.90	0.87	0.92	0.90	0.86	0.93	0.91	0.89
		GPT-4o zero-shot	0.89	0.82	0.74	0.88	0.82	0.72	0.90	0.85	0.82
		GPT-4o baseline	0.81	0.82	0.62	0.80	0.82	0.64	0.85	0.83	0.68
	4	Llama3-8B zero-shot	0.75	0.57	0.47	0.68	0.54	0.40	0.86	0.86	0.83

(continued)

Length	Position	Model	$F1_C$	$F1_T$	$F1_J$	R_C	R_T	R_J	P_C	P_T	P_J
		Llama3-8B instruction-tuned	0.95	0.80	0.77	0.94	0.78	0.74	0.96	0.86	0.84
		Llama3-70B zero-shot	0.91	0.85	0.79	0.90	0.84	0.78	0.92	0.91	0.85
		Llama3-70B instruction-tuned	0.93	0.83	0.79	0.92	0.82	0.78	0.95	0.84	0.81
		GPT-4o zero-shot	0.92	0.83	0.80	0.92	0.82	0.78	0.93	0.90	0.88
		GPT-4o baseline	0.91	0.70	0.65	0.90	0.68	0.64	0.92	0.74	0.76
5	All	Llama3-8B zero-shot	0.73	0.67	0.51	0.72	0.66	0.51	0.78	0.74	0.60
		Llama3-8B instruction-tuned	0.82	0.87	0.73	0.81	0.87	0.72	0.85	0.87	0.77
		Llama3-70B zero-shot	0.83	0.87	0.71	0.82	0.87	0.71	0.83	0.87	0.73
		Llama3-70B instruction-tuned	0.86	0.91	0.79	0.86	0.91	0.79	0.88	0.91	0.81
		GPT-4o zero-shot	0.81	0.87	0.72	0.81	0.87	0.72	0.82	0.87	0.74
		GPT-4o baseline	0.83	0.77	0.65	0.83	0.77	0.66	0.83	0.78	0.69
	0	Llama3-8B zero-shot	0.75	0.70	0.54	0.74	0.70	0.56	0.80	0.77	0.62
		Llama3-8B instruction-tuned	0.88	0.94	0.85	0.88	0.94	0.84	0.88	0.95	0.86
		Llama3-70B zero-shot	0.88	0.92	0.81	0.88	0.92	0.80	0.89	0.93	0.84
		Llama3-70B instruction-tuned	0.90	0.98	0.91	0.90	0.98	0.90	0.90	0.98	0.93
		GPT-4o zero-shot	0.90	0.90	0.83	0.90	0.90	0.82	0.90	0.92	0.85
		GPT-4o baseline	0.90	0.82	0.74	0.90	0.82	0.74	0.90	0.83	0.79
	1	Llama3-8B zero-shot	0.75	0.69	0.53	0.74	0.68	0.54	0.78	0.74	0.57
		Llama3-8B instruction-tuned	0.72	0.88	0.66	0.68	0.88	0.62	0.84	0.88	0.80
		Llama3-70B zero-shot	0.79	0.84	0.68	0.78	0.84	0.68	0.80	0.87	0.79
		Llama3-70B instruction-tuned	0.81	0.89	0.75	0.78	0.90	0.72	0.86	0.90	0.83
		GPT-4o zero-shot	0.79	0.89	0.71	0.76	0.90	0.68	0.84	0.91	0.82
		GPT-4o baseline	0.80	0.73	0.63	0.80	0.76	0.66	0.80	0.80	0.70
	2	Llama3-8B zero-shot	0.69	0.69	0.49	0.68	0.68	0.48	0.74	0.76	0.57
		Llama3-8B instruction-tuned	0.81	0.79	0.65	0.80	0.78	0.64	0.84	0.81	0.70
		Llama3-70B zero-shot	0.77	0.86	0.63	0.76	0.86	0.62	0.79	0.88	0.70
		Llama3-70B instruction-tuned	0.89	0.86	0.75	0.88	0.86	0.74	0.91	0.87	0.81
		GPT-4o zero-shot	0.78	0.84	0.66	0.78	0.84	0.66	0.80	0.86	0.69
		GPT-4o baseline	0.78	0.80	0.66	0.78	0.80	0.66	0.78	0.81	0.67
	3	Llama3-8B zero-shot	0.76	0.70	0.55	0.76	0.68	0.54	0.78	0.78	0.63
		Llama3-8B instruction-tuned	0.86	0.90	0.77	0.86	0.90	0.76	0.87	0.91	0.79
		Llama3-70B zero-shot	0.90	0.84	0.76	0.90	0.84	0.76	0.90	0.84	0.78
		Llama3-70B instruction-tuned	0.88	0.94	0.82	0.88	0.94	0.82	0.90	0.94	0.85
		GPT-4o zero-shot	0.86	0.82	0.76	0.86	0.82	0.76	0.86	0.83	0.79
		GPT-4o baseline	0.86	0.70	0.58	0.86	0.70	0.62	0.87	0.71	0.57
	4	Llama3-8B zero-shot	0.68	0.64	0.43	0.66	0.64	0.44	0.79	0.73	0.55
		Llama3-8B instruction-tuned	0.85	0.84	0.75	0.84	0.84	0.74	0.89	0.84	0.81
		Llama3-70B zero-shot	0.80	0.86	0.65	0.80	0.86	0.66	0.84	0.87	0.73
		Llama3-70B instruction-tuned	0.85	0.86	0.74	0.84	0.86	0.74	0.88	0.86	0.81
		GPT-4o zero-shot	0.75	0.88	0.68	0.74	0.88	0.66	0.81	0.89	0.77
		GPT-4o baseline	0.81	0.72	0.56	0.80	0.72	0.58	0.84	0.74	0.61
	5	Llama3-8B zero-shot	0.78	0.60	0.51	0.76	0.60	0.48	0.84	0.71	0.65
		Llama3-8B instruction-tuned	0.83	0.86	0.76	0.82	0.86	0.74	0.88	0.87	0.81
		Llama3-70B zero-shot	0.82	0.88	0.74	0.82	0.88	0.74	0.82	0.88	0.76
		Llama3-70B instruction-tuned	0.88	0.90	0.80	0.88	0.90	0.80	0.89	0.90	0.81
		GPT-4o zero-shot	0.80	0.86	0.71	0.80	0.86	0.72	0.81	0.87	0.74
		GPT-4o baseline	0.83	0.82	0.71	0.84	0.82	0.72	0.83	0.83	0.73

Table A7: **Thread prediction scores.**
This table contains the F1, precision, and recall scores for the thread-prediction models. Length and position denote the number of replies of the thread and the position of a particular text (where 0 equals the original comment). The top set of rows shows the results overall across all comments and replies and the remaining rows show the performance for each thread length, starting with the aggregate score across all texts, followed by scores for texts in a specific position. The Model column lists the models used. The remaining columns are split into three groups: F1 scores for Clinton (C), Trump (T), and the joint score (J); Recall scores; and Precision scores. The best-performing model across all lengths and positions is in bold.