Towards Coding Social Science Datasets with Language Models

Anonymous ACL submission

Abstract

Researchers often rely on humans to code (label, annotate, etc.) large sets of texts. This is a highly variable task and requires a great deal of time and resources. Efforts to automate this process have achieved human-level accuracies in some cases, but often rely on thousands of hand-labeled training examples, which makes them inapplicable to small-scale research studies and still costly for large ones. At the same time, it is well known that language models can classify text; in this work, we use OpenAI's GPT-3 as a synthetic coder, and explore what classic methodologies and metrics (such as intercoder reliability) look like in this new context. We find that GPT-3 is able to match the performance of typical human coders and frequently outperforms humans in terms of intercoder agreement across a variety of social science tasks, suggesting that language models could be a useful tool to the social sciences.

1 Introduction

The analysis of textual data–from sources such as open responses to surveys, social media posts, newspaper articles, legislative transcripts, etc.–has become increasingly important for researchers across a variety of disciplines. In the social sciences, for example, analysis of free-form text is used to gather information not easily obtained from traditional closed-ended survey analysis or observation. Traditionally, researchers interested in quantitative content analysis of text have hired and trained (mostly) undergraduate students to *code* the material by assigning numbers, labels, and/or categories to segments of text describing attributes and content of interest. However, such human coding is slow, expensive, often unreliable, and requires extensive time in training and norming. Given variability in experience and perception among coders, researchers hire multiple people to evaluate the same texts, and then calculate intercoder agreement as a measure of confidence that they have collectively identified the things the researchers hope to glean from these texts.

While such an approach works somewhat well for small amounts of text, it is infeasible as a means to analyze the scale of text available in an increasingly digital, information-rich world. To address this problem, researchers have developed a number of supervised machine learning (SML) models to code text in the place of humans. While many of these models perform well, they (like the use of human coders) require extensive time and expense as researchers label thousands of examples as training data, tune hyperparameters, etc. This means that SML models work well for large datasets, but often do not scale down to smaller uses.

Language models (LMs), such as GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), BERT (Devlin et al., 2019) and others, offer an alternative. It is well-known that language models can analyze text and classify it, and it is not our purpose to simply present social-science themed results to that effect. Rather, in this paper we ask: if we consider language models as serious tools of the social science, can we analyze their output with tools and metrics common to the social sciences, and will the results be similar?

In this paper, we show that one such LM, GPT-3 (Brown et al., 2020), is able to perform coding tasks at or exceeding the level of lightly-trained human coders with only 0-3 exemplars (examples of text labeled with a code), upholding the broader trend of effective transfer in NLP. GPT-3 maintains this coding proficiency across a variety of tasks (sentiment, attributes of text, or classification), difficulties (number of possible codes, objective versus subjective, etc.), and co-domains (ordinal versus nominal codes). This suggests that this same model and general method could successfully be used for many other such coding tasks.

Our main contributions are (1) demonstrating that large, pre-trained language models can be used as reliably as human coders on arbitrarily-sized datasets across diverse domains; (2) introducing and exploring social science metrics in the context of language models; and (3) proposing new social science coding tasks as benchmark problems to assess language model quality.

2 Related Work

Because human coding is time-consuming, costly, and still subject to imprecision and variability (Soroka, 2014), many scholars seek automated alternatives. Dictionary-based methods (Roberts and Utych, 2020; Young and Soroka, 2012) work best in cases where clearly defined sets of words indicate the presence of particular content in the text, as opposed to more subtle patterns. They also struggle with generalization (Barberá et al., 2021; Grimmer and Stewart, 2013). This is especially discouraging, given that developing and validating them is expensive (Muddiman and Stroud, 2017).

Therefore, researchers have increasingly turned to supervised machine learning (SML) methods as an alternative, such as naive bayes, random forests, and SVMs (Grimmer and Stewart, 2013; Barberá et al., 2021). Some authors use active learning (Hillard et al., 2008; Collingwood and Wilkerson, 2012; Miller et al., 2020), or dictionary-SML ensemble approaches (Dun et al., 2021). Unfortunately, all of these require a large dataset for training. Typically, this training data is hand-generated by human coders, meaning that SML methods do not completely negate the time and expense of human coders. For instance, (Collingwood and Wilkerson, 2012) find that 100 labeled examples results in a 10 percentage-point drop in accuracy compared to 1000 labeled examples.

In contrast, we leverage the few- and zero-shot capabilities of language models to almost entirely eliminate the need for hand-coded labels. Some researchers have used pre-trained language models such as BERT (Devlin et al., 2019), BART (Lewis et al., 2020), RoBERTa (Liu et al., 2019b), XLNet (Yang et al., 2019), and ELMo (Peters et al., 2018) in automated content analysis. However, to our knowledge, this is the first in-depth comparison between human coders and a language model coder in a few-shot learning regime.

It is easy to compare our approach to SML in terms of cost, since the model we study requires no additional training or labeled data; it is less straightforward to compare performance. It is common in SML classification studies to set rejection thresholds and ignore instances in which a code cannot be confidently assigned (Sebők and Kacsuk, 2021; Karan et al., 2016). In what follows, we report scores for the entire dataset, meaning they cannot be directly compared to this past work.

One critique against work claiming to do few-shot learning is that researchers iterate through many prompts over large validation sets to achieve their results (Perez et al., 2021), essentially overfitting to the dataset and using an entire dataset of exemplars. We avoid this problem by using a very small validation set to test prompts (n=4 per category) and by being transparent about the small amount of experimentation and prompt-engineering done to achieve our results (Section 4.3). We find only minimal (∼5% accuracy boost) gains from prompt engineering.

3 Methodology

Through various data sources metrics, we show that LMs perform coding tasks just as well as humans, and they do so without labeled data. Specifically, we study GPT-3 (Brown et al., 2020), one of the largest available language models (175 billion parameters). This model–along with others comparable in size and training–often generates text that, at least locally, is indistinguishable from that written by a human, seeming to capture a great deal of the ideas, concepts, and relationships present in human-generated text and language, including linguistic and factual knowledge (Liu et al., 2019a; Amrami and Goldberg, 2018; Jiang et al., 2020; Rogers et al., 2020; Petroni et al., 2020; Bosselut et al.; Bouraoui et al.). We leverage these abilities and prompt a language model to simulate a human performing coding tasks. We carefully templatize prompts, parameterizing them by testing candidates on a validation set of labeled social science data, and analyze the predictive distributions for tokens representing codes.

We construct our prompts using a straightforward formula: we provide instructions, categories (if necessary), exemplars (labeled examples of the task), and then the text to classify. We then compute GPT-3's probabilities for the next token over its vocabulary and select the token with the highest probability as the language model's coding choice. For color-coded examples of our prompts, see Figure 1.

These coding tasks are subjective, noisy, and

Using only the following categories
"""
Macroeconomics
Civil Rights, Minority Issues, and Civil Liberties
Health
...
Death Notices
Churches and Religion
Other, Miscellaneous, and Human Interest
"""
Assign the following headlines to one of the categories:
IRAN TURNS DOWN AMERICAN OFFER OF RELIEF MISSION -> International Affairs and Foreign Aid
In Final Twist, Ill Pavarotti Falls Silent for Met Finale -> Arts and Entertainment
Baseball; Incredibly, Yankees Rally in 9th Again and Win in 12 -> Sports and Recreation
House Panel Votes Tax Cuts, But Fight Has Barely Begun ->

(a) CAP Example Prompt - New York Times, 3-exemplars

Are the following descriptions of Republicans extreme or moderate?
-angry, racist, close-minded, homophobic: Extreme
-people, hopeful, educated, agreeable: Moderate
-conservative, white, male, religious:

(b) Pig. Partisans Example Prompt - Positivity, 2-exemplars

Do the following descriptions of Democrats mention personality or character traits?
-accepting, tolerant, intellectual, charitable: Yes, the descriptions mention personality or character traits.
-black, young, female, poor: No, the descriptions do not mention personality or character traits.
-conservative, white, male, religious:

(c) Pig. Partisans Example Prompt - Traits, 2-exemplars

Figure 1: Example Prompts

varying in difficulty, and so, as with many datasets researchers want to code, there is no "ground truth" by which to measure an automated coder's performance. Therefore, we evaluate GPT-3's coding performance using metrics that differ substantially from those used in traditional NLP work, but which are common analytic tools in the social sciences: we calculate various intercoder agreement measures between GPT-3's codes and the codes generated by humans we hired to code the same texts.

3.1 Metrics

We now discuss the three central metrics in our analysis, and outline when each is appropriate.

3.1.1 Intraclass correlation (ICC)

Intraclass correlation is perhaps the most commonly used metric among social scientists to measure the degree of inter-coder agreement among human coders using numerically ordered, (quasi-) continuous values in their coding (e.g., rating a text by some characteristic on a 1-5 scale). In the "PP" coding task that follows, we estimate ICC1k for our human coders and GPT-3 using the methods proposed by (Shrout and Fleiss, 1979). ICC scores are between -1 and 1 and are typically interpreted as follows: < 0.5 = poor inter-coder agreement, $0.5 - .75$ = moderate agreement, $0.75 - 0.9$ = good, and > 0.9 = excellent (Cicchetti, 1994; Koo and Li, 2016).

3.1.2 Joint probability of agreement

For coding tasks in which coders use unordered, categorical data to classify texts (as in the Congressional and New York Times tasks presented below), ICC is not the appropriate metric. Instead, we use two different measures. The first, joint-probability of agreement, measures the probability of any two coders agreeing. In the 2-coder case, where one of the coders is ground truth, this reduces to raw accuracy. Joint probability agreement ranges from 0 to 1. Between two coders, it is calculated as follows: $\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(y_{1,i} = y_{2,i})$, where N is the number of instances being coded, and $y_{1,i}, y_{2,i}$ are the first coder's and the second coder's respective codings of instance i. In the case of K coders, the joint probability agreement is the mean of the pairwise agreements.

3.1.3 Fleiss' kappa

Fleiss' kappa measures the degree to which the proportion of agreement among coders exceeds what would be expected if all coders made their ratings completely at random (Fleiss, 1971; Fleiss et al., 2003). Used specifically to quantify intercoder agreement for categorical data, this measure ranges from -1 to 1. When $\kappa = 0$, it means that the two raters agree at a rate not better than chance. $\kappa < 0$ means increasing agreement worse than chance, and $\kappa > 0$ means increasing agreement greater than chance.

4 Experiments

In general, we show that GPT-3 can effectively perform coding tasks of varying difficulty across several domains, and with at most a few labeled examples. This speaks to the flexibility of GPT-3 as a coder and its ease of use. We show this using data from three datasets: Pigeonholing Partisans (PP), New York Times Headlines (NYT), and Congressional Hearings (Congress).

We chose these datasets to maximize differences in coding tasks as a means of exploring GPT-3's limits. The dimensions they span include:

- **Difficulty:** We expect that some tasks will be easy for the language model to master, e.g., rating positivity (Section 4.1) through sentiment analysis (Radford et al., 2017), and that some will be harder, like subjective tasks (Section 4.1) or tasks with a large number of codes to choose from (Section 4.2.2).

- **Domains:** Section 4.1 explores partisan polarization through descriptions of members of both political parties in the U.S., whereas Section 4.2.2 defines a schema for categorizing newspaper headlines and 4.2.1 does so for summaries of congressional hearings.

- **Category Type:** Ordinal and binary codes are used throughout Section 4.1, while nominal and categorical codes are used in Sections 4.2.1 and 4.2.2.

GPT-3's flexibility in adapting to the range along all of these dimensions is reason to believe that it can readily excel on many coding tasks.

4.1 Pigeonholing Partisans (PP)

We first consider the ability of GPT-3 to act as a coder with data on Americans' stereotypes of Republicans and Democrats (Rothschild et al., 2019). These data, collected in 2016, asked individuals to list four words or phrases that came to their minds when thinking of typical supporters of the Democratic and Republican Parties. This procedure is common in psychological studies of stereotypes (Devine, 1989; Eagly and Mladinic, 1989), and allows survey takers to describe partisans in their own words without being primed by researchers and closed-ended answer choices (Presser, 1989; Iyengar, 1996). This dataset is too small for other kinds of automated coding and an ideal way to consider how well GPT-3 can classify texts without extensive training sets.

To evaluate how well GPT-3 can serve as a coder on these kinds of short, open-ended texts, we recruited 2873 human coders through the survey platform *Lucid* (Coppock and McClellan, 2019) to code a total of 7675 texts, each text being coded at least three times by a random set of coders, and gave them minimal instructions for coding the texts on a number of domains.

Coders rated the texts along five dimensions: (1) positivity (general positive/negative valence), (2) extremity (extreme or moderate quality of the words), and whether the text mentioned (3) character or personality traits, (4) government or policy issues, or (5) social groups. Each of these domains is important to the theoretical ideas of the original orientation of the data collection on partisan

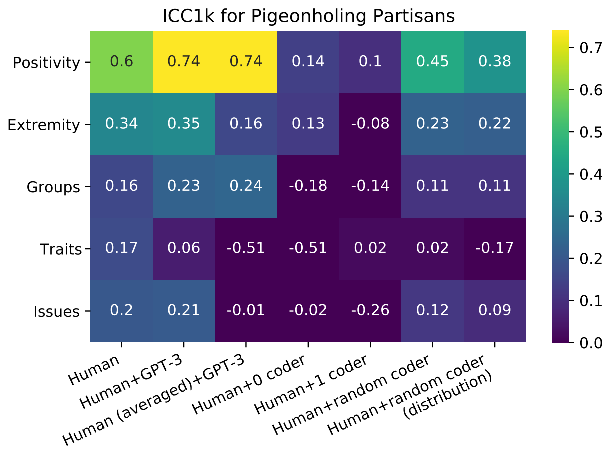

Figure 2: PP ICC1k: Note that including GPT-3 in the class of considered coders increases ICC1k in coding for all attributes except "Traits". The opposite happens when including other, simulated coders.

stereotypes (Rothschild et al., 2019; Busby et al., Forthcoming). While we do not broach this subject in this work, each represents a distinct way of thinking about party attachments and membership that have different political and social consequences.

Then we asked GPT-3 to complete a series of coding tasks on all 7675 texts that are directly analogous those completed by humans. Next, we examined how closely GPT-3 follows individual human coders and human coding in the aggregate, along with how closely humans followed each other. To calculate a correlation statistic, we rely on the probabilities produced by GPT-3 for the attribute in question (probability of extreme, traits, or positive, for example) and the untransformed code from the human respondents. We present these correlations in Figure 3. They suggest that GPT-3 performs above human level in every case but one. That is, for positivity, extremity, groups, and issues, GPT-3 correlates more strongly with each of the human coders than the human coders do with each other. For traits, GPT-3 correlates with the human coders about as well, or slightly lower, than the humans correlate with each other. This is initial evidence that GPT-3 is typically either more reliable or just as reliable a coder as human coders, a remarkable finding given that GPT-3 was provided no more than 2 exemplars in its "training set".

We also consider ICC scores (Fig. 2). As we employ different coders - that is, coders are randomly assigned to texts and not all texts are scored by the same three coders - we use ICC1k, which accounts for this structure.

Our focus here is on the increase or decrease in

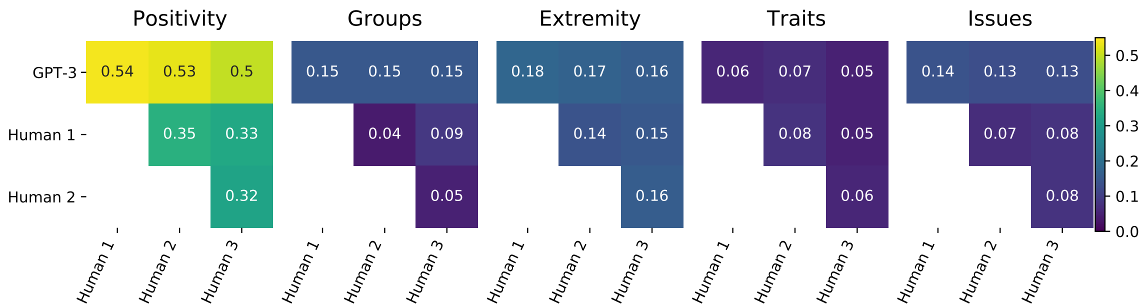

Figure 3: Correlations for PP, calculated with Pearson's R. Other measures of correlation yield similar results. Notice how correlation is higher for GPT-3 and every human than between any two humans. There are only two cells (Humans 1 & 2, 2 & 3 in Traits) strictly greater than any one of GPT-3's correlations with humans.

ICC when GPT-3's codes are added to the three human codes. If GPT-3 improves the reliability of the coding, ICC should improve. If it does not offer this benefit, the ICC score should stay the same or decrease. We also compare adding GPT-3's scores to adding a variety of simulated scores to ensure that the addition of another coder by itself does not drive what we observe: (1) a coder who codes all texts as 0 (lacking the attribute), (2) a coder who codes all texts as 1 (containing the attribute), (3) a coder who codes randomly, and (4) a coder who codes all texts randomly, but with the same overall distribution as GPT-3's predictions. We also consider the ICC values when comparing GPT-3's codes to the average of the human coders (rather than individual coders separately).

The statistics in Figure 2 suggest that adding GPT-3 as a coder improves the overall coding for 2/5 measures (positivity, groups), improves reliability of the coding for 2/5, (extremity, issues), and reduces reliability in 1/5 (traits). Notably, this last area is where human coders correlated the least with each other (see Figure 3) and may represent a fundamentally challenging task.

Another point to note is the stark difference between adding GPT-3 and adding each of the simulated coders (2nd and 3rd columns vs. 4th+). We conclude that GPT-3's outputs do contain real signal and that the boost in ICC is not due to simply adding another coder. Furthermore, since adding GPT-3's outputs to the human outputs generally either increases or maintains ICC across each attribute, we conclude that GPT-3 achieves human or super-human level performance at this task. Importantly, achieving this level of performance required neither coding a large-scale dataset (on the order of tens of thousands or more) nor a large, labeled set of training data for the language model.

4.2 Comparative Agendas Project (CAP)

CAP aims to provide a coherent framework for documenting media and government attention to various policy issues in a comprehensive set of policy domains, without reference to the support or opposition stance or ideological framing of the issue in the source material (Baumgartner et al., 2019). CAP datasets aim to be comprehensive, transparent, and replicable (Bevan, 2019), with many housed at the CAP website (www.comparativeagendas.net). More than 200 scholars have used CAP to test a vast range of empirical political science theories (Walgrave and Boydstun, 2019).

The CAP master codebook includes at least 21 major categories (with others added for some specific applications), and over 200 sub-categories. In order to succeed at this task, GPT-3 must produce a high probability for one of a large, unordered, pre-specified set of tokens that corresponds to the specific content of the input data.

Prior efforts to use dictionary-based and SML approaches to classification in the CAP framework have met limited success (Karan et al., 2016; Hillard et al., 2008; Purpura and Hillard, 2006; Sevenans et al., 2014; Sebők and Kacsuk, 2021). Sebok and Kacsuk (Sebők and Kacsuk, 2021) are able to achieve an 80%+ F1 score on average across categories, but this is reported after culling over 40% of their dataset due to difficulty of classification. We, on the other hand, provide scores given full coverage of the dataset. Reported performance in various approaches is substantially lower than this (accuracies near or below 50%) for dictionary methods, less efficient SMLs, corpora with less training data, or in specific hard-to code categories, which upper limit our average accuracy exceeds. Again, the highest performing outcomes are achieved by setting rejection thresholds (for ambiguous texts or cases where humans or models disagree) and either

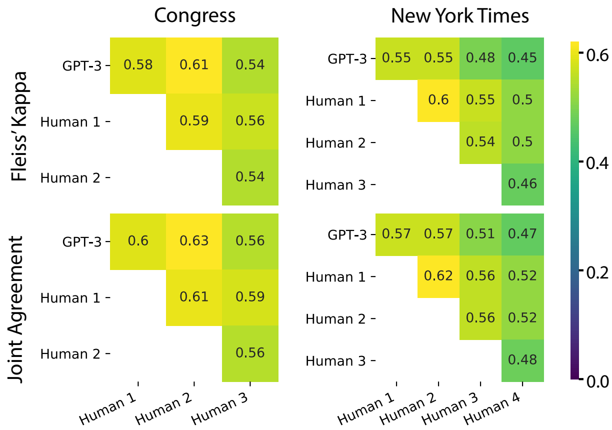

Figure 4: Two measures of GPT-3's agreement with human coders compared with humans' agreement with human coders, across two datasets.

sacrificing coverage or targeting human coders to uncertain cases (Karan et al., 2016; Sebők and Kacsuk, 2021). We achieve our results with complete coverage, a single model, no human disambiguation of difficult cases, and minimal need for labeled training data.

To account for class imbalances and differences in baseline probabilities of different tokens, we normalize the probability distributions in a manner similar to (Zhao et al., 2021). We estimate GPT-3's bias towards a category as the total weight given to each category over a balanced validation set, divide each category probability by GPT-3's bias towards it, and normalize to sum to 1. We found that this produced modest accuracy boosts of 4-5%. If a small validation set is available, we recommend this calibration technique; however, results were qualitatively the same without this calibration.

4.2.1 CAP: Congressional Hearing Summaries (Congress)

The Congressional Hearing corpus contains the *Congressional Information Service* summary of each U.S. Congressional hearing from 1946 to 2010. These summaries were read by human coders and assigned to CAP classifications. GPT-3 is given the full summary text, meaning the coding task is highly comparable between the humans and GPT-3. All results are reported for $n = 326$ texts, which constitutes 16 texts for each category minus 10 for incompleteness in the human codes.

Our comparison of GPT-3's codes to the humans' is in Figure 4. Both our intercoder agreement metrics tell the same story, and imply a finding that holds across metrics: GPT-3 correlates with each human just as well as or better than the humans correlate with each other. Note that the highest joint agreement (.63) and highest Fleiss' kappa (.61) both occur between GPT-3 and Human 2.

Despite there being no real ground truth for this task, we visualize "accuracy" statistics based on the original dataset's single coder (Figure 5). The lack of ground truth is validated by a great deal of human disagreement, as the figure makes clear. We see the accuracy for each coder, with categories sorted in order of GPT-3's accuracy. Interestingly enough, GPT-3 seems to do better at categories that humans do better at, and worse at the categories that humans fail at. Overall, the accuracies were 60% for GPT-3, compared to 63%, 66%, and 55% for the three human coders respectively.

Between the high joint agreement and Fleiss' kappa between GPT-3 and the human coders and the similar accuracies across categories, we believe that GPT-3 has demonstrated performance on-par with humans and SML methods on this dataset.

4.2.2 CAP: *New York Times* Front Page Dataset (NYT)

The second CAP dataset we use is the *New York Times* Front Page Dataset, generated and contributed by Amber Boydstun (Boydstun, 2013). The dataset includes 31034 front page *New York Times* headlines from 1996 - 2006, along with the policy category label assigned by trained human coders. The categories are adapted for media use, and so include 28 primary classification categories. All results are reported for $n = 560$ texts, with 20 sampled from each of the 28 categories.

The original human coders were instructed to read the headline *and the first three paragraphs of the article.* In our work, GPT-3 is only provided the headline, because the full article text is not available in the public data. To control for this difference in available information, we also had three minimally trained human coders complete an identical classification task to GPT-3.

Since the NYT data is in the same structure as the Congress data, we apply the same analyses. For both joint agreement and Fleiss' kappa (Figure 4), GPT-3 correlates with the humans in the range of how they correlate with each other. We also notice a strong trend between GPT-3's accuracy and the humans accuracy per category (Figure 6). Unlike Congress, however, there are 3 categories that the humans all perform much better than GPT-3: "International Affairs and Foreign Aid," "Government Operations," and "Death Notices." On the other

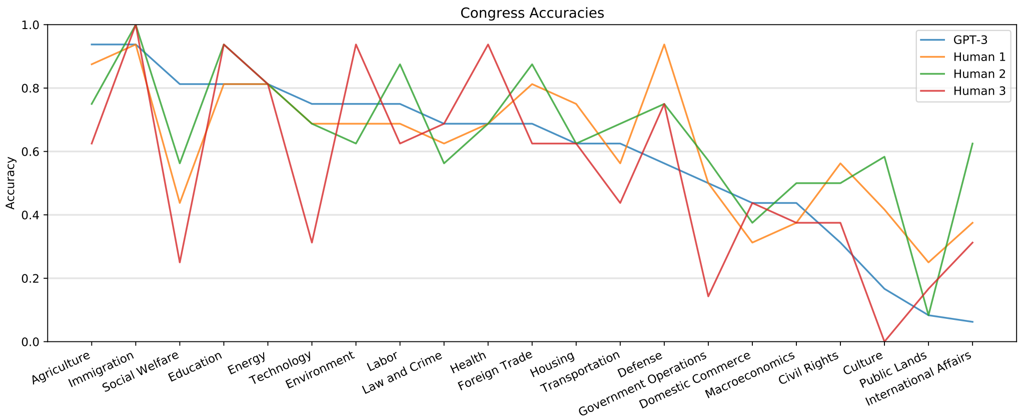

Figure 5: Congress Accuracy by Coder: Treating the original dataset's code as "ground truth", and sorting categories in descending order based on GPT-3's score, note how noisy the performance of the human coders is. There is only 1 category that all humans score strictly better on (International Affairs).

hand, GPT-3 performs significantly better than humans at some of the categories: "Environment," "Health," and "Labor." Despite this discrepancy, GPT-3's total accuracy was 55%, compared to 57%, 59%, 51%, and 45% for the four humans respectively. Overall, we have demonstrated that GPT-3 on average achieves on-par performance with humans for the New York Times dataset (remembering that performance is systematically worse or better depending on category).

4.3 Prompt Engineering

Some elements of prompt engineering seem to matter a great deal, and some seem to matter not at all, or at least not in any reliable way.

As an example of the former, one has to be mindful of where the prompt ends and what next token is being modeled. Since generative language models sample one token at a time, we need to be able to sample a unique first token (usually, a unique first word) for each category we attempt to model. For example, "very positive" and "very negative" both start with the token "very," so it would be impossible for us to compare the two categories with a single token sample. Fortunately, all of our categories started with unique first tokens, but this may not be true for other tasks.

Another choice that impacted our results was the presentation of categories in the question format of the PP data. Specifically, GPT-3 performed significantly worse when asked to respond to a question with the tokens "yes" or "no" than when the choice was between substantive alternatives, such as "extreme" vs "moderate" or "positive" vs. "negative". For the other three attributes, we found that restating the objective after the "yes" or "no" (e.g., "Yes, mentions personality or character traits") substantially helped. These were the only prompt variations attempted for the PP dataset.

Other elements seemed to have minimal impact, like the number and type of exemplars. While we know that more labeled training data significantly improves SML performance (Collingwood and Wilkerson, 2012), it is unclear whether more labeled exemplars to GPT-3 will achieve the same. As shown in Figure 7, we find that one exemplar performs much better than none, but there is little gain in accuracy achieved by providing more than 2 or 3 exemplars. We also conducted extensive experiments testing different classes of exemplars (more or less difficult to classify, in the spirit of active learning), and that also seemed not to matter (See Appendix B for details).

We also tried many variations on the prompt format, including: surrounding categories in quotes; using slashes, pipes, and other delimiters to separate exemplar headlines from their respective categories; providing lists of example headlines for each category in parentheses right next to the category; new lines in specific places making boundaries between exemplars clearer; and other general rephrasing. None of these changes resulted in a marginal accuracy less than 50% or greater than 57%. This demonstrates a relative stability of the information retrieval process, allaying some concerns (though not all) that minor changes in wording or punctuation will radically alter coding accuracy.

For all of our final prompts used, please refer to Appendix A.

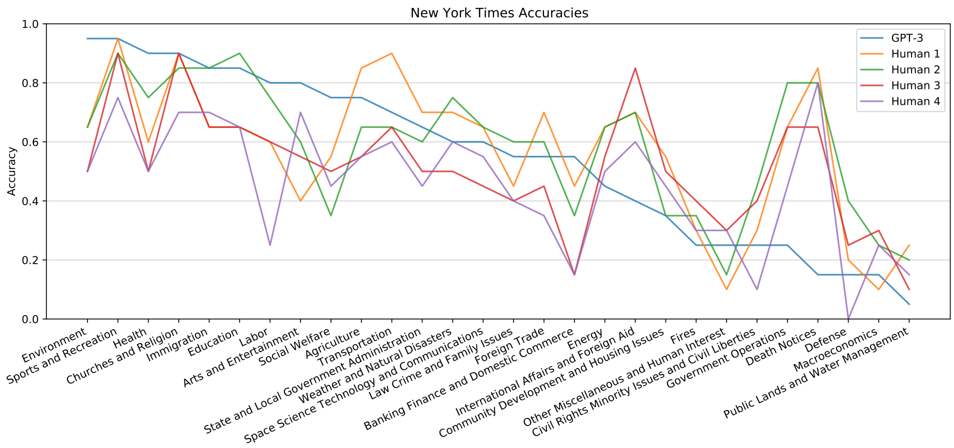

Figure 6: New York Times Accuracy by Coder: Treating the original dataset code as "ground truth", and sorting categories in descending order according to GPT-3's score, note how noisy the humans' coding is. Clearly some areas are easier for human coders (e.g., Death Notices) and some are easier for GPT-3 (e.g., Environment).

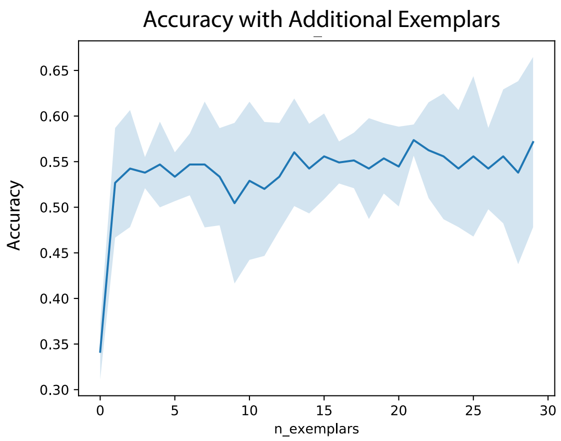

Figure 7: Increasing number of exemplars up to 30 shows no improvement past 2 or 3. This experiment was done on the NYT dataset.

5 Future Work

The results presented in this paper provide encouraging evidence that GPT-3 is able to perform automated coding tasks with effectiveness comparable to that of lightly-trained human coders. However, much work remains in order to bring this possibility to full fruition. Further explorations should be conducted into principled distribution calibration and prompt engineering, in order to capitalize on the full capabilities of LMs. Fine-tuning approaches should also be investigated; perhaps it is possible to refine the model weights such that categorical text continuations become more probable and/or accurate, especially within specialized domains. We recommend the application of these methods to additional datasets, potentially with the assignment of multiple labels for each text, in order to validate the robustness of this technique across multiple research domains. Finally, we propose that future explorations into automated coding via GPT-3 utilize the contextual nature of GPT-3's responses in order to actively simulate the coding behaviors of specific populations. For example, the conditioning prompts used in the current work could be pre-pended with information designed to elicit responses that emulate those of specific demographic groups, thus creating additional fidelity to human coding scenarios.

6 Conclusion

We have demonstrated that LMs can potentially be used to code social science datasets and that they can be analyzed with metrics common in the social sciences. Fine-grained analysis shows that GPT-3 can match the performance of human coders on average across small and large datasets; with both ordinal and categorical codes; and on tasks of varying complexity. In some cases, it even outperforms humans in increasing intercoder agreement scores, often with no more than 3 exemplars.

We hope that these results initiate a two-way dialogue: the social sciences are a rich source of potential applications and benchmarks for LMs, but as LMs play an increasing role throughout sciences–with LMs and humans potentially working side-by-side–it is possible that the field of NLP will need to move beyond traditional notions of accuracy and analyze LMs with methods such as those presented here to ensure their reliability. Harnessing LMs as synthetic coders will open up a new world of possibilities, which is a worthy endeavor indeed.

References

Asaf Amrami and Yoav Goldberg. 2018. Word Sense Induction with Neural biLM and Symmetric Patterns. pages 4860–4867.

Pablo Barberá, Amber E. Boydstun, Suzanna Linn, Ryan McMahon, and Jonathan Nagler. 2021. Automated text classification of news articles: A practical guide. *Political Analysis*, 29(1):19–42.

Frank Baumgartner, Christian Breunig, and Emiliano Grossman. 2019. The comparative agendas project: Intellectual roots and current developments.

Shaun Bevan. 2019. Gone fishing: The creation of the comparative agendas project master codebook. In *Comparative Policy Agendas*, pages 17–34. Oxford University Press.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. COMET : Commonsense Transformers for Automatic Knowledge Graph Construction.

Zied Bouraoui, Jose Camacho-collados, and Steven Schockaert. Inducing Relational Knowledge from BERT.

Amber E Boydstun. 2013. *Making the news: Politics, the media, and agenda setting*. University of Chicago Press.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv:2005.14165*.

Ethan C. Busby, Adam J. Howat, Jacob E. Rothschild, and Richard M. Shafranek. Forthcoming. *The Partisan Next Door: Stereotypes of Party Supporters and Consequences for Polarization in America*. Cambridge University Press.

DV Cicchetti. 1994. Guidelines, criteria, and rules of thumb for evaluating normed and standardized assessment instruments in psychology. *Psychological assessment*.

Loren Collingwood and John Wilkerson. 2012. Tradeoffs in accuracy and efficiency in supervised learning methods. *Journal of Information Technology & Politics*, 9(3):298–318.

Alexander Coppock and Oliver A. McClellan. 2019. Validating the demographic, political, psychological, and experimental results obtained from a new source of online survey respondents. *Research & Politics*, 6(1):1–14.

Patricia G. Devine. 1989. Stereotypes and prejudice: Their automatic and controlled components. *Journal of Personality and Social Psychology*, 56(1):5–18.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lindsay Dun, Stuart Soroka, and Christopher Wlezien. 2021. Dictionaries, supervised learning, and media coverage of public policy. *Political Communication*, 38(1-2):140–158.

Alice H. Eagly and Antonion Mladinic. 1989. Gender stereotypes and attitudes toward women and men. *Personality and Social Psychology Bulletin*, 15(4):545–558.

JL Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*.

Joseph L. Fleiss, Bruce Levin, and Myunghee Cho Paik. 2003. *Statistical Methods for Rates and Proportions*. Wiley-Interscience.

Justin Grimmer and Brandon M Stewart. 2013. Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political analysis*, 21(3):267–297.

Dustin Hillard, Stephen Purpura, and John Wilkerson. 2008. Computer-assisted topic classification for mixed-methods social science research. *Journal of Information Technology & Politics*, 4(4):31–46.

Shanto Iyengar. 1996. Framing responsibility for political issues. *Annals of the American Academy of Political and Social Science*, 546(1):59–79.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Mladen Karan, Jan Šnajder, Daniela Širinić, and Goran Glavaš. 2016. Analysis of policy agendas: Lessons learned from automatic topic classification of croatian political texts. In *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 12–21.

TK Koo and MY Li. 2016. A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of chiropractic medicine*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:1073–1094.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Blake Miller, Fridolin Linder, and Walter R Mebane. 2020. Active learning approaches for labeling text: review and assessment of the performance of active learning approaches. *Political Analysis*, 28(4):532–551.

Ashley Muddiman and Natalie Jomini Stroud. 2017. News values, cognitive biases, and partisan incivility in comment sections. *Journal of communication*, 67(4):586–609.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. Language models as knowledge bases? *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 2463–2473.

Stanley Presser. 1989. Measurement issues in the study of social change. *Social Forces*, 68(3):856–868.

Stephen Purpura and Dustin Hillard. 2006. Automated classification of congressional legislation. In *Proceedings of the 2006 international conference on Digital government research*, pages 219–225.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Damon C Roberts and Stephen M Utych. 2020. Linking gender, language, and partisanship: Developing a database of masculine and feminine words. *Political Research Quarterly*, 73(1):40–50.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Jacob E. Rothschild, Adam J. Howat, Richard M. Shafranek, and Ethan C. Busby. 2019. Pigeonholing partisans: Stereotypes of party supporters and partisan polarization. *Political Behavior*, 41(2):423–443.

Miklós Sebők and Zoltán Kacsuk. 2021. The multiclass classification of newspaper articles with machine learning: The hybrid binary snowball approach. *Political Analysis*, 29(2):236–249.

Julie Sevenans, Quinn Albaugh, Tal Shahaf, Stuart Soroka, and Stefaan Walgrave. 2014. The automated coding of policy agendas: A dictionary based approach (v. 2.0.). In *CAP Conference*, pages 12–14.

Patrick E Shrout and Joseph L Fleiss. 1979. Intraclass correlations: uses in assessing rater reliability. *Psychological bulletin*, 86(2):420.

Stuart Soroka. 2014. Reliability and validity in automated content analysis. In *Communication and language analysis in the corporate world*, pages 352–363. IGI Global.

Stefaan Walgrave and Amber E Boydstun. 2019. The comparative agendas project. *Comparative Policy Agendas: Theory, Tools, Data.*

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Lori Young and Stuart Soroka. 2012. Affective news: The automated coding of sentiment in political texts. *Political Communication*, 29(2):205–231.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models.

A Prompts For Each Task

A.1 Pigeonholing Partisans

- **Positivity**:

  Are the following descriptions of PARTY positive or negative?

  -agreeable, reasonable, understanding, cooperative: Positive
  -angry, bigoted, racist, homophobic: Negative

- **Groups**:

  Do the following descriptions of PARTY mention social groups?

  -Christian, privileged, young, white: Yes, mentions social groups.
  -apathetic, agreeable, pro-environment, political: No, doesn't mention social groups.

- **Traits**:

  Do the following descriptions of PARTY mention personality or character traits?

  -accepting, tolerant, intellectual, charitable: Yes, mentions personality or character traits.
  -black, young, female, poor: No, doesn't mention personality or character traits.

- **Extremity**:

  Are the following descriptions of PARTY extreme or moderate?

  -angry, racist, close-minded, homophobic: Extreme
  -people, hopeful, educated, agreeable: Moderate

- **Issues**:

  Do the following descriptions of PARTY include government or policy issues?

  -aging, religious, accepting, patriotic: No, doesn't include government or policy issues.
  -abortion, medical marijuana, gun control, anti-sexism: Yes, includes government or policy issues.

A.2 CAP

- **Congressional Hearings**:

  Using only the following categories
  """
  Macroeconomics
  Civil Rights
  Health
  Agriculture
  Labor
  Education
  Environment
  Energy
  Immigration
  Transportation
  Law and Crime
  Social Welfare
  Housing
  Domestic Commerce
  Defense
  Technology
  Foreign Trade
  International Affairs
  Government Operations
  Public Lands
  Culture
  """
  Assign the following congressional hearing summaries to one of the categories:
  Extend defense production act provisions through1970. -> Defense
  FY90-91 authorization of rural housing programs. -> Housing
  Railroad deregulation. -> Transportation
  To consider Federal Reserve Board regulations and monetary policies after February 2016 report on monetary policy. ->'

- **New York Times Headlines**

  Using only the following categories
  """
  Macroeconomics
  Civil Rights, Minority Issues, and Civil Liberties
  Health

Agriculture
Labor
Education
Environment
Energy
Immigration
Transportation
Law, Crime, and Family Issues
Social Welfare
Community Development and Housing Issues
Banking, Finance, and Domestic Commerce
Defense
Space, Science, Technology and Communications
Foreign Trade
International Affairs and Foreign Aid
Government Operations
Public Lands and Water Management
State and Local Government Administration
Weather and Natural Disasters
Fires
Arts and Entertainment
Sports and Recreation
Death Notices
Churches and Religion
Other, Miscellaneous, and Human Interest
"""
Assign the following headlines to one of the categories:
IRAN TURNS DOWN AMERICAN OFFER OF RELIEF MISSION -> International Affairs and Foreign Aid
In Final Twist, Ill Pavarotti Falls Silent for Met Finale -> Arts and Entertainment
In Times Sq., a Dry Run for New Yearś 2000 -> Arts and Entertainment
House Panel Votes Tax Cuts, But Fight Has Barely Begun ->'

B Exemplar Types Experiments

We also explored whether some exemplars were better or worse at "teaching" the categories to the model. We considered that for a given category,

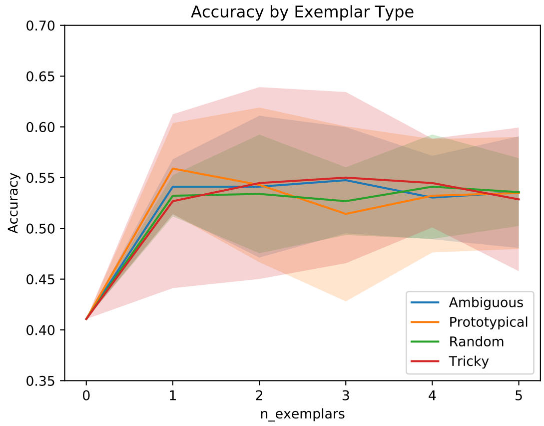

Figure 8: Each class of exemplar considered does an equal amount to help the model's accuracy. This is surprising, and suggests that the model might learn nothing from the exemplars besides the format of the task.

an instance could be a better or worse exemplar. We might define this by a quantity we'll call its *margin*: the difference between (1) the probability the model assigns to the correct category and (2) the highest probability of the probabilities for all the wrong categories. Thus, "prototypical" exemplars would have high positive margin (model guesses right), "ambiguous" exemplars would have margins with very low absolute values (model torn between multiple categories), and "tricky" exemplars would have margins with very high negative values (model guesses wrong). In theory, prototypical exemplars could teach the model about the proper distribution of texts belonging to a category, ambiguous exemplars could teach the model about the boundaries between the distributions of each category, and tricky exemplars could correct the model's prior on categories by flagging common mistakes made in coding texts from that category's distribution.

To answer this question empirically, we first randomly sample 90 candidate exemplars from each category. We then code each with the model given a set of 4 exemplars sampled randomly once and then held constant specifically for this task. Then we sort them by their margin and construct one set of each: prototypical, ambiguous, and tricky exemplars. Finally, we perform 5 trials where we classify 4 instances from each category using an increasing number of these sets of exemplars and measure performance. The results, in Figure 8, demonstrate no discernible signal as to which kind of exemplar is best to present to the model in the context window. This is one bit of evidence that this dimension, of the prototypicality vs. ambiguity vs. trickiness of exemplars, is not at all determinative of a model's performance on a coding task, a dimension which is very important for active learning.