Describing Differences between Text Distributions with Natural Language

Ruiqi Zhong [1] **Charlie Snell** [1] **Dan Klein** [1] **Jacob Steinhardt** [1]

Abstract

How do two *distributions* of text differ? Humans are slow at answering this, since discovering patterns might require tediously reading through hundreds of samples. We propose to automatically describe the differences by "learning a natural language hypothesis": given two distributions D_0 and D_1, we search for a description that is more often true for D_1, e.g., "*is military-related.*" To tackle this problem, we fine-tune GPT-3 to propose descriptions with the prompt: "[samples of D_0] + [samples of D_1] + *the difference between them is* __." We then re-rank the descriptions by checking how often they hold on a larger set of samples with a learned verifier. On a benchmark of 54 real-world binary classification tasks, while GPT-3 Curie (13B) only generates a description similar to human annotation 7% of the time, the performance reaches 61% with fine-tuning and re-ranking, and our best system using GPT-3 Davinci (175B) reaches 76%. We apply our system to describe distribution shifts, debug dataset shortcuts, summarize unknown tasks, and label text clusters, and present analyses based on automatically generated descriptions.

1. Introduction

What inputs trigger a neuron in my deep learning model? How are the train and test distributions different for my application? How did public opinions on Twitter change from last year to this year? These questions have significant scientific, economic, and social consequences. However, discovering new patterns sometimes requires scanning over thousands of examples, intractable for humans. An automated solution would be far more scalable.

To address this, we develop a method to discover the differences between two distributions and describe them with natural language. We reduce the above questions to "learning a natural language hypothesis" (Section 2): given two text distributions D_0 and D_1, we search for a natural language hypothesis s that is more often true for samples from D_1 than samples from D_0. For instance:

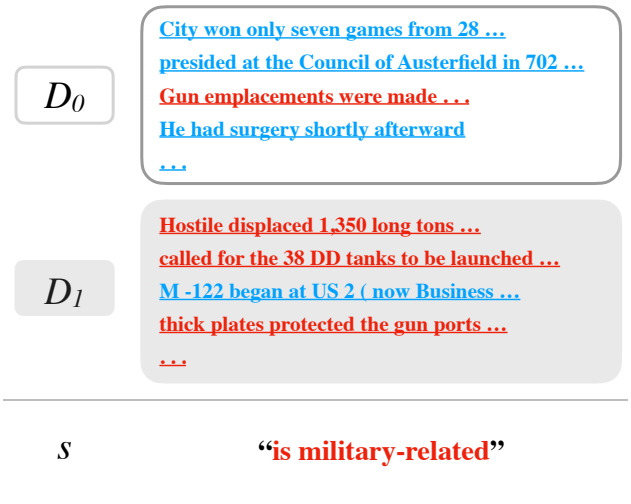

Figure 1. Given two distributions (top), our system automatically discovers their differences and describes them with natural language (bottom). Grey/white background represents D_0/D_1 and red/blue represents whether a sample matches the description s.

- We can describe what triggers an artificial neuron by setting D_1 to be inputs that trigger it and D_0 for other inputs. s could be "*is military-related*" (Figure 1).
- We can describe the differences between the train and test distributions by setting them to be D_0 and D_1. A possible s would be "*is longer in sentence length.*"
- We can describe how public opinions shifted by setting D_0/D_1 to be the opinions from last year/this year. s could be "*is optimistic about the pandemic.*"

We develop a new method to learn a natural language hypothesis. We first prompt GPT-3 Davinci (175B) (Brown et al., 2020) with samples from each distribution and ask it to propose candidate hypotheses s (Section 3.1). However, since GPT-3 has a limited context size, this prompt can only

[1]Computer Science Division, University of California, Berkeley. Correspondence to: Ruiqi Zhong <ruiqi-zhong@berkeley.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

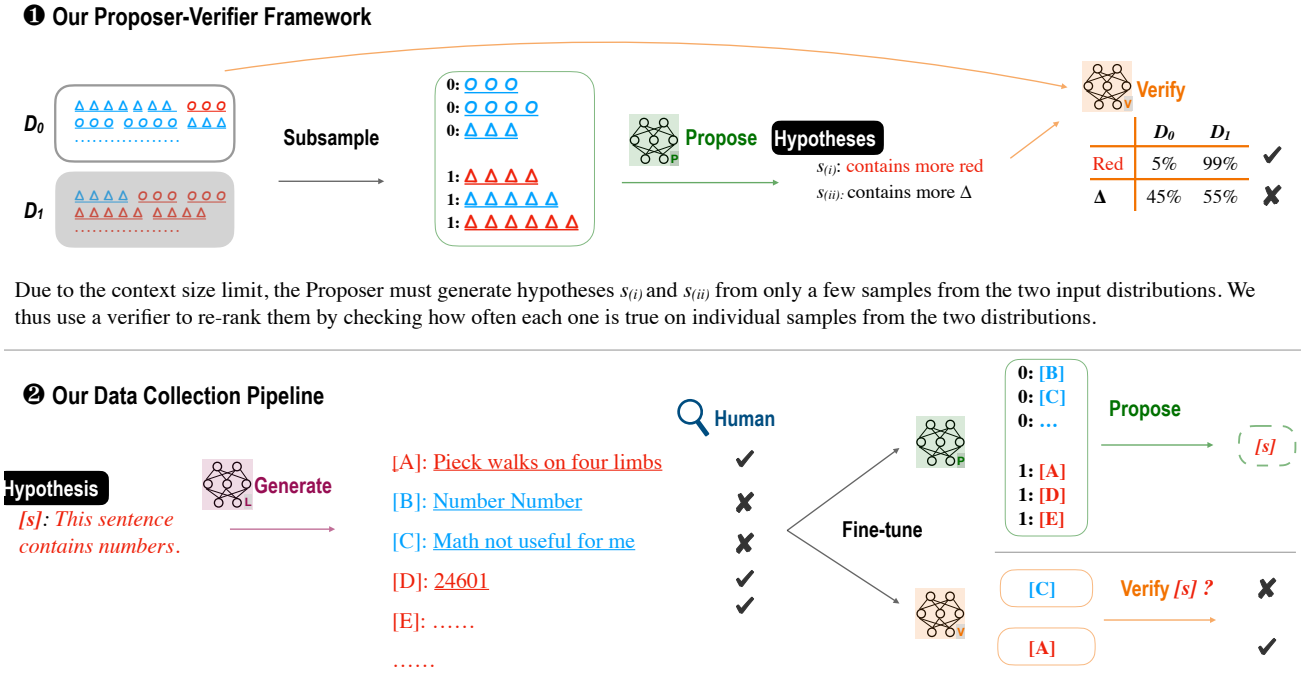

Figure 2. Our architectural framework (**top**) and data collection pipeline (**bottom**). Section 3 describes them in detail.

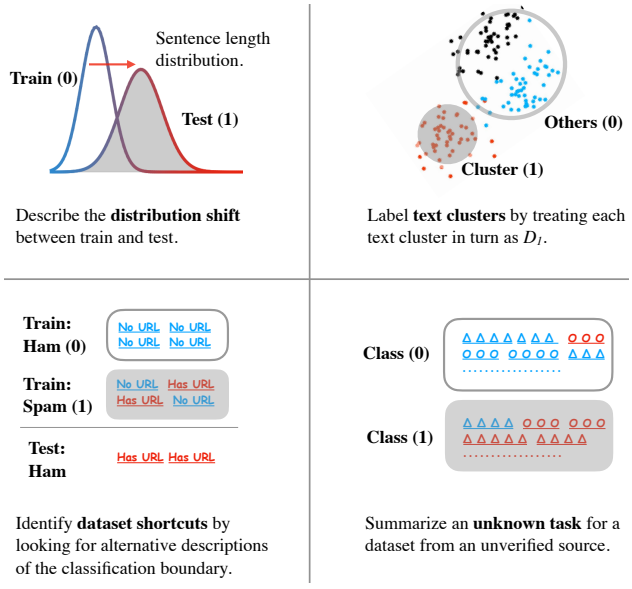

Figure 3. We reduce a wide range of applications to learning a natural language hypothesis and present our analyses in Section 5.

contain a few samples rather than the whole distributions. Therefore, we re-rank the candidates with a verifier that checks how often they hold on a larger set of samples (Section 3.2). We visualize our framework at the top of Figure 2 and the prompts at the top of Figure 4.

Since GPT-3 is not optimized to propose hypotheses, we can improve it through fine-tuning. However, no corpus exists for this task yet. Therefore, we developed a new data collection pipeline (Section 3.3) with three stages: 1) we curated a list of hypotheses s, 2) we asked GPT-3 to generate samples that satisfy s, and 3) we asked annotators to judge whether they indeed satisfy s. Then we fine-tuned the proposer to predict s based on samples that satisfy s and samples that do not (Section 3.4). We visualize our data collection and fine-tuning method at the bottom of Figure 2.

We benchmark our system on 54 real-world binary classification datasets (Zhong et al., 2021), each annotated with natural language descriptions for the positive class. For each binary task, we treat the positive/negative class inputs as D_1/D_0 and compare the top-5 descriptions by our system to the human annotation. While the descriptions by GPT-3 Curie (13B) are similar to the annotations only 7% of the time, the performance reaches 61% with fine-tuning and verifier re-ranking, and our best system using GPT-3 Davinci (175B) reaches 76% (Section 4).

We then check whether the intended uses of existing classi-

fication datasets agree with the descriptions by our system (Section 5). Our system correctly recognizes that the subjectivity analysis (SUBJ) dataset (Pang & Lee, 2004) was constructed by contrasting movie reviews with plot summaries; however, many recent papers (Bragg et al., 2021; Zhong et al., 2021; Gao et al., 2021; Min et al., 2021) were unaware of this fact and used SUBJ for zero/few-shot subjectivity classification. Our system also recognizes several dataset shortcuts. For example, it rediscovered that negations, such as the use of "not/never", is spuriously correlated with the contradiction class in MNLI (Gururangan et al., 2018); for another example, models trained on the SMS Spam classification dataset (Gómez Hidalgo et al., 2006) always consider hyperlinks to be spam. Our system can also describe distribution shifts and text clusters (Section 5), and Figure 3 visualizes all our applications. We conclude with future applications in other modalities (e.g., vision) and research fields (e.g., social science) in Section 7.[1]

2. Learning a Natural Language Hypothesis

Let $\mathcal{X}$ be the set of all text inputs. A natural language hypothesis h is parameterized by a natural language string s and is a mapping from two inputs to a boolean:

$$h_s : \mathcal{X} \times \mathcal{X} \to \{0, 1\}, \tag{1}$$

where $h_s(x_1, x_0) = 1$ means x_1 is more s than x_0. For example, if s is "*is longer in sentence length*," then $h_s(x_1, x_0) = 1$ means x_1 is longer than x_0. The semantics of h_s is defined as

$$h_s(x_1, x_0) \stackrel{\text{def}}{=} \mathbf{1}[\text{humans consider } x_1 \text{ more } s \text{ than } x_0], \tag{2}$$

which our paper operationalizes by taking majority vote among crowdworkers.[2] We call both s and h_s "hypotheses" but write s when using it as a string and h_s as a function.

Let D_0 and D_1 be two distributions over $\mathcal{X}$, and $\mathcal{H}$ be the space of all valid natural language hypotheses. We search for h in $\mathcal{H}$ to maximize its "classification accuracy" CA,

$$\text{CA}(h) \stackrel{\text{def}}{=} \mathbb{E}_{x_0 \sim D_0, x_1 \sim D_1}[h(x_1, x_0))]. \tag{3}$$

Intuitively, given two random samples from each distribution $x_0 \sim D_0$ and $x_1 \sim D_1$, h should classify where each x comes from as accurately as possible. Therefore, our task falls under the standard formulation of statistical machine learning, where we learn a hypothesis h by optimizing a statistical objective (CA) over a hypothesis space $\mathcal{H}$.

Compared to traditional statistical learning, learning a natural language hypothesis poses two new challenges.

Search. Searching in a discrete string space is hard. Section 3.1 addresses this by proposing h_s with a neural network based on samples from D_0 and D_1.

Verify. Computing $h_s(x_1, x_0)$ requires human annotations, which can be expensive. Section 3.2 addresses this by approximating human responses with a neural network.

3. Method

We prompt GPT-3 to propose hypotheses based on a small set of samples (Section 3.1) and use UnifiedQA to verify each hypothesis on a larger set of samples (Section 3.2). Then, we design a data collection pipeline (Section 3.3) to further fine-tune the proposer and the verifier (Section 3.4). Our methods can be visualized in Figure 2.

3.1. Hypothesis Proposer

Our goal is to generate a list of plausible hypotheses based on samples from D_0 and D_1. We do so by prompting GPT-3, a language model that can generate textual completions based on a prompt. We construct a "proposer prompt" by concatenating several samples from D_1, several from D_0, and the instruction "*Compared to group 0, each sentence from group 1 __*" (Figure 4, the 1st row). Since GPT-3 has a context size limit of 2048, we select 5 samples x from each distribution.

Without controlled decoding, a typical prompt completion would be "*is more positive, while sentences from group 0 are ungrammatical.*" However, such a completion is undesirable, since 1) the verifier now needs to check two statements at the same time, namely, whether samples from D_1 are positive and samples from D_0 are ungrammatical, and 2) the second half of the completion describes a population-level property of "group 0", while our verifier only checks hypotheses on individual x. To produce a single hypothesis about individual x, we forbid GPT-3 to decode tokens like "*group*" and terminate the generation with token "," or ".".

Additionally, D_0 and D_1 might overlap, and even an optimal hypothesis h^* cannot fully separate them. As a result, the proposer prompt might contain samples from D_1 that do not satisfy h^*, thus confusing the proposer. Therefore, we choose samples that are representative of their differences to prompt GPT-3. To find those samples, we fine-tune RoBERTa-Large (Liu et al., 2019) to predict whether each sample comes from D_0 or D_1 and keep the top-p percentile samples with the highest confidence. For the top-5, 20, and 100th percentile, we construct proposer prompts with ten different random sets of samples and generate two completions for each set. In total we obtain $3 \times 10 \times 2 = 60$ hypotheses. We re-rank them in the next section.

[1] Appendix G discusses details about code and data release.

[2] More broadly, however, there is no canonical method to interpret natural language. See Section 7 for more discussion.

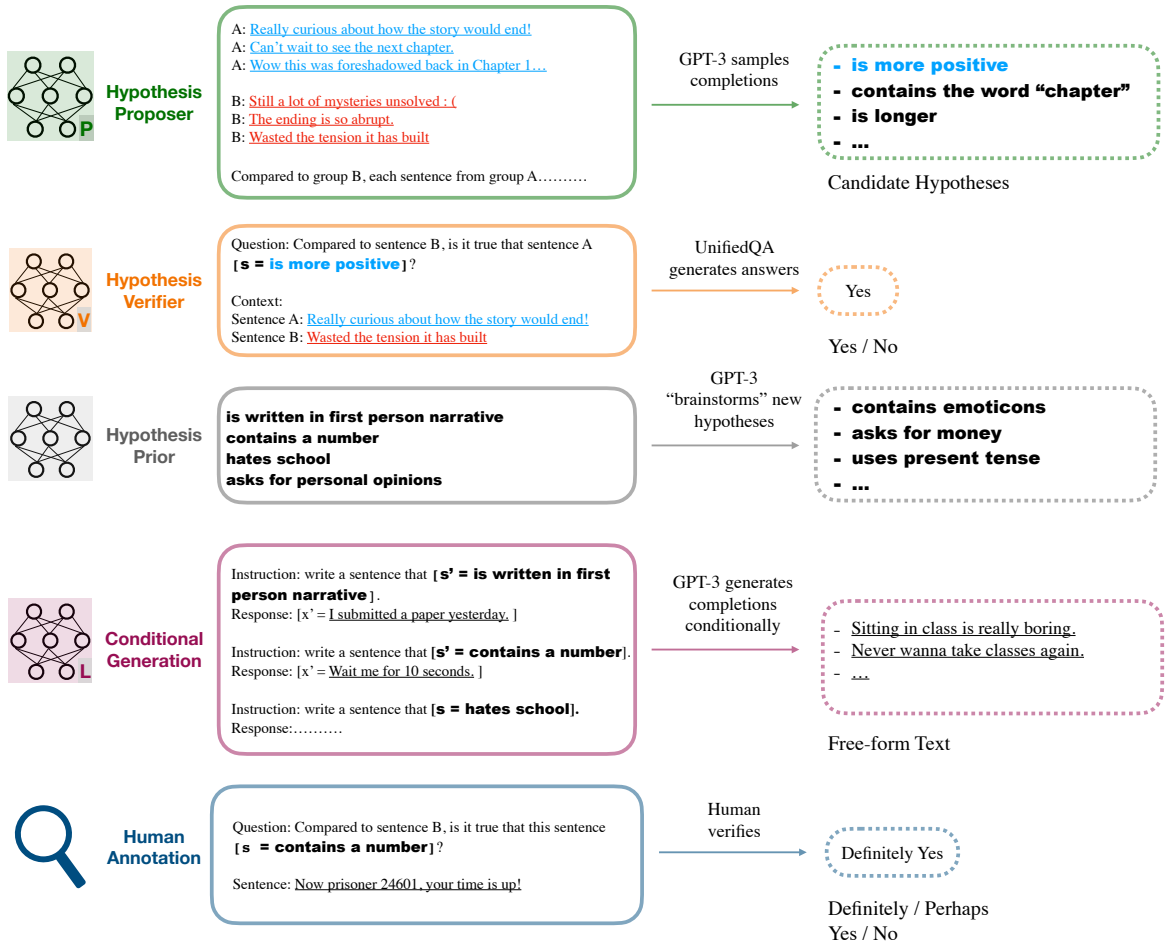

Figure 4. The prompt template for all components in our system. All text datapoints x are underlined and hypotheses s bolded.

3.2. Hypothesis Verifier

Ideally, we should re-rank h_s based on its classification accuracy $\text{CA}(h_s)$, defined in Equation (3). However, it involves computing $h_s(x_1, x_0)$, which requires expensive human annotations (Equation (2)). We therefore approximate it with a verifier neural network V:

$$\hat{h}_s(x_1, x_0) \overset{\text{def}}{=} \frac{1}{2}(V(s, x_1, x_0) - V(s, x_0, x_1) + 1). \quad (4)$$

Here $V(s, x_1, x_0) = 1$ if it predicts that x_1 is more s than x_0 (0 otherwise); then we subtract the baseline $V(s, x_0, x_1)$ obtained by swapping the position of x_0 and x_1, and finally normalize the quantity within $[0, 1]$.

We implement our verifier with UnifiedQA (Khashabi et al., 2020), a question answering model based on T5 (11B) (Raffel et al., 2019). UnifiedQA generates an answer a given a question q and a context c. As shown in the 2nd row of Figure 4, our context c is a pair of sentences A (sampled from D_1) and B (sampled from D_0). The question q is then "*Is it true that sentence A* ***is more positive***?", where in general the bolded part is a hypothesis s generated by the proposer. Then we define $V(s, x_1, x_0) = 1$ if UnifiedQA outputs "yes" and 0 if it outputs "no".

We now use $V(s, x_1, x_0)$ to compute $\text{CA}(\hat{h}_s)$ for each candidate s and re-rank them. To save computation, we estimate $\text{CA}(\hat{h}_s)$ with 400 random pairs of (x_1, x_0) rather than using the entire datasets. Finally, we output the top-5 hypotheses to describe how D_1 and D_0 differ.

3.3. Collecting Data for Supervision

Since GPT-3 and UnifiedQA are not specifically trained to propose or verify hypotheses, we can improve them by fine-tuning (Zhong et al., 2021). However, since no corpus exists yet for these tasks, we need to collect a new dataset to fine-tune our models.

To fine-tune the proposer, we want data where the output is a hypothesis s and the input prompt contains five samples that are more s and five that are less s. To fine-tune the verifier, we want tuples (s, x_1, x_0) where x_1 is more s than x_0. Thus

for both cases, we want a set of hypotheses s, and for each of them, two groups of samples where one group is more s than the other. We designed our data collection pipeline accordingly: we curated a set of hypotheses s, asked GPT-3 to generate samples that do (not) satisfy s, and asked humans to filter out failed generations.

Curating Hypotheses. We curated a pool of 302 hypotheses by hand with the help of GPT-3 (Brown et al., 2020). Concretely, we started the pool by brainstorming ten hypotheses ourselves; then, we sampled five hypotheses from the pool and prompted GPT-3 with their concatenation, as visualized in the 3rd row of Figure 4. Whenever GPT-3 completed the prompt with a hypothesis different from our existing ones, we added it to the pool.

Our curated hypotheses ranges from shallow ("*contains the word "yay" at the end of the sentence*") to topical ("*loves school*") to more complex social and linguistic cues ("*supports universal healthcare*," "*is written in first person*"). To make later conditional generation and human annotation easier, we removed any comparatives from s, e.g., removing the word "*more*" from "*loves school more.*"

Conditional Generation. We refer to samples that satisfy s as "positive" and others as "negative". For example, given $s =$ "*loves school*", a positive sample could be "*My advisor is really helpful and I learned a lot.*" Both positive and negative samples are necessary to fine-tune our models.

To generate positive samples, we prompted GPT-3 as visualized in the 4th row of Figure 4: we curated a set of hypotheses s' and their positive samples x' by hand, concatenated them with the target hypothesis s, and asked GPT-3 to generate a sample x. Sometimes, however, x satisfies s due to trivial word overlaps, e.g., $x =$ "*I love school*" satisfies $s =$ "*loves school*." We curated a list of forbidden output tokens for each hypothesis s by hand to prevent this.

We created negative samples for s by using positive samples for other hypotheses. If s is highly specific, e.g., "*talks about microwaves*," a random sample is unlikely to satisfy it. Therefore, we treat the positive samples of any other hypotheses as the negative samples for s. However, for s like "*uses past tense*", a random sample can satisfy it with non-trivial probability. Therefore, we wrote contrast hypotheses such as "*uses future tense*" and used their positive samples as the negative samples for s. Hence, our pool expanded to 352 hypotheses after including newly written ones, and we asked GPT-3 to generate 15 positive samples for each.

Verifying with Human Annotations. Some samples x from the conditional generation step do not actually satisfy the hypothesis s. To filter out samples that fail, for each (s, x) pair, we recruited turkers[3] to verify whether x satisfies s, as visualized in the 5th row of Figure 4. We collected three annotations for each (s, x) pair and determined the ground truth by majority vote. Finally, for each s, if fewer than five x's passed the turker vote, the authors wrote additional examples by hand.

Thus, for each of the initial 302 hypotheses, we obtained at least five positive and five negative samples for it. We next use these to fine-tune our models.

3.4. Fine-tuning

Proposer. For each of the 302 hypotheses s, we finetuned GPT-3 to generate s based on five positive and five negative samples. We used batch size 20 and a small learning rate of 0.05 to prevent memorizing the target. We fine-tuned for two epochs, each using a different set of subsamples to construct the prompt.

Verifier. Given s and a positive/negative sample x_1/x_0, our verifier should predict that $V(s, x_1, x_0) = 1$ and $V(s, x_0, x_1) = 0$. To create a fine-tuning dataset, we randomly sampled 30 positive-negative pairs of (x_1, x_0) for each s. We fine-tuned UnifiedQA on this dataset for 250 steps with batch size 32 and learning rate 5e-5. To improve out-of-distribution robustness, we averaged the fine-tuned model weights with UnifiedQA (Wortsman et al., 2021).

4. Benchmarking Performance

On a benchmark of 54 real-world binary classification tasks, we show that 1) both re-ranking and fine-tuning are effective, and 2) larger proposers and verifiers are better.

Dataset. The evaluation set of Zhong et al. (2021) aggregated 54 diverse binary text classification tasks, each annotated with one or multiple[4] natural language descriptions s^* for the positive class. These tasks include topic classification, grammaticality classification, stance classification, etc. For each task, we asked our systems to describe how the positive class samples differ from the negative class samples and compared the top-5 descriptions the human annotations.

For now, we assume that the annotations s^* are "correct" (i.e., the best description to separate the positive and negative classes). We will see later that our outputs are sometimes better than s^*.

Evaluated Systems. We conjectured that using a larger proposer, a fine-tuned proposer, and a verifier for re-ranking all improve the generated descriptions. Therefore, we evaluated the following five systems, which all use the verifier from Section 3.4 unless otherwise mentioned. ①: our hy-

[3]We recruited turkers located in the U.S. with $> 98\%$ HIT acceptance rate and paid them \$0.04 per HIT; we estimate our pay rate to be \$18/hrs based on how fast the authors perform this task.

[4]On average 2.2.

	① best	② smaller	③ no tune	④ no verifier	⑤ memo
(A)	31	22	11	4	5
(B)	10	11	6	0	5
(C)	7	10	10	6	21
(D)	6	11	27	44	23

Table 1. We evaluated each of the five systems as described in Section 4. ① largest fine-tuned proposer + verifier, ② smaller proposer size, ③ no fine-tuining, ④ no re-ranking, and ⑤ using the memorization proposer. Better systems have larger numbers in row (A). Using a larger proposer, a fine-tuned proposer, and a verifier all improve the generated descriptions. We report the p values in Appendix B.

pothetically best system, which uses the fine-tuned GPT-3 Davinci (175B) as the proposer. ②: a smaller proposer size (fine-tuned Curie, 13B). ③: no fine-tuning (zero-shot Curie, 13B). ④: no fine-tuning (zero-shot Curie, 13B), and no verifier for re-ranking. We also evaluated ⑤, a "memorization proposer", where the proposer only generates the hypotheses we curated in Section 3.3; this ablation makes sure that the fine-tuned proposer's performance is not simply due to memorizing its training set. If all our conjectures hold, we should find that ① > ② > ③ > ④ and ② > ⑤.

Automatic Evaluation. We first evaluated our systems using the automatic metric BERTscore (Zhang et al., 2019), which approximates the similarity between two natural language texts. For each binary task, we computed the BERTscore between every pair of the human annotations and the top-5 descriptions; then, we chose the highest score among all pairs and averaged it across 54 tasks.

Using this metric, we indeed found that ① (0.930) > ② (0.927) > ③ (0.907) > ④ (0.899), and ② (0.927) > ⑤ (0.916), which validated our conjectures. However, all these numbers are high, the differences are small, and it is hard to interpret what they imply for the quality of our descriptions.[5] Therefore, we additionally evaluated our systems by hand.

Manual Evaluation. We evaluated the top-5 descriptions generated for each of the five systems on the 54 binary tasks (total 1350) by hand. To avoid biases against any of the five systems, the authors were blind towards which system generated each description. We compared the systems' generated descriptions $\hat{s}$ to human annotations s^* and rated their similarity with four levels:

(A), if $\hat{s}$ has mostly the same meaning as one of the human annotations s^*; e.g., "*is related to sports*" = "*is about sports*."

(B), if $\hat{s}$ is close but different; e.g., "*is about sports team*" $\approx$ "*is about sports*."

(C), if $\hat{s}$ is highly correlated but has different meaning; for example, "*people needs shelter*" is correlated with "*there is an earthquake*."

(D), if $\hat{s}$ is unrelated to s^*.

[5]Appendix A runs a sanity check to make sure that the scores, though not very informative, robustly rank system ① over ④.

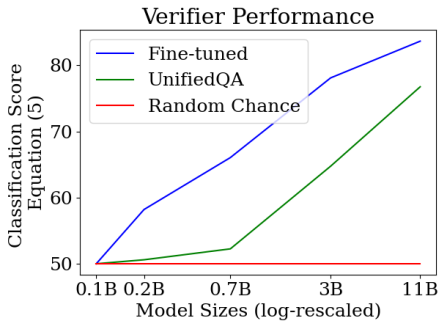

Figure 5. We compared verifiers of various sizes and UnifiedQA out of the box by evaluating their binary classification performance, using the metric $\text{CA}(\hat{h}_{s^*})$ explained in Equation (5). We find that fine-tuning and larger model sizes improve the performance.

For each system, we find the highest rating among the top-5 descriptions and count them across 54 tasks. We find that for row (A), ① > ② > ③ > ④ and ② > ⑤, validating our conjectures. Adding numbers from row (A) and (B), we find that while GPT-3 Curie (13B) only generates a description close to human annotation 7% of the time, the performance reaches 61% with fine-tuning and re-ranking, and our best system using GPT-3 Davinci (175B) reaches 76%. In the appendix, we also present the top-1 performance of our system in Table 2, example human annotations, descriptions by our systems, and their ratings in Table 3.

Due to resource constraints, we did not systematically investigate whether the verifier is still effective after fine-tuning. Nevertheless, our qualitative analyses find that the fine-tuned proposer sometimes still generates completely unrelated hypotheses, repeats the hypothesis in the training set, or "rants" [6] based on a specific text sample. The verifier helps rule them out. Finally, the proposer has a limited context size and can only generate hypotheses conditioned on five samples, losing information about the entire distribution; the verifier does not have this fundamental limitation.

Comparing Verifiers. We next evaluate different choices of the verifier. To test a verifier, we check whether it can reliably separate the two classes when given the gold annotation h^*. More precisely, we compute

[6]E.g., "*contains the word "turned", which indicates that the weather turned to a certain state*"

$$\frac{1}{2}\mathbb{E}_{x_0\sim D_0, x_1\sim D_1}[V(s^*, x_1, x_0) - V(s^*, x_0, x_1) + 1], \quad (5)$$

which is equivalent to the classification accuracy $\mathrm{CA}(\hat{h}_{s^*})$.

We conjectured that larger and fine-tuned verifiers are better, so we compared our fine-tuned verifier in Section 3.4 with smaller ones and UnifiedQA out of the box, averaging $\mathrm{CA}(\hat{h}_{s^*})$ across all 54 tasks. Figure 5 visualizes the results. UnifiedQA performs decently, while additional fine-tuning improves the performance. Still, $\mathrm{CA}(\hat{h}_{s^*})$ is much lower than 1, implying that re-ranking is imperfect and automatic evaluation by approximating $\mathrm{CA}(h_s)$ might not yet be feasible. Nevertheless, these problems may be alleviated in the future: the current state of the art models are at least 25x larger than our verifier (Rae et al., 2021), and the curve in Figure 5 predicts that their performance will be higher.

5. Application

We applied our system to summarize training tasks, debug dataset shortcuts, describe distribution shifts, and label text clusters. All italicized quotes in this section are verbatim generations from our system.

Summarizing Training Tasks. Human descriptions can be imperfect even for widely-used binary classification datasets. For example, the subjectivity analysis (SUBJ) dataset (Pang & Lee, 2004) was proposed as classifying between subjective vs. objective texts, and several works (Bragg et al., 2021; Zhong et al., 2021; Gao et al., 2021; Min et al., 2021) have used it to test zero/few-shot subjectivity classification. However, our system generates descriptions "*is a plot summary of a film*" for the "objective" class and "*is a quote from a film review*" for the "subjective" class. We therefore re-read Pang & Lee (2004) carefully, which says (edited for brevity)

> To gather subjective sentences, we collected 5000 movie review snippets from www.rottentomatoes.com. To obtain (mostly) objective data, we took 5000 sentences from plot summaries available from www.imdb.com.

Therefore, our system's descriptions were in fact more accurate. We conjecture that similar problems will become increasingly prevalent as the trend of aggregating datasets continues (Mishra et al., 2021b; Sanh et al., 2021): as datasets come from heterogeneous sources, it is a management challenge to characterize the task of every dataset accurately. Our system may help here.[7]

Debugging Dataset Shortcuts. Datasets frequently contain unintended shortcuts. For example, the task of natural language inference (NLI) is to verify whether a `hypothesis`[8] is an entailment or a contradiction given a premise. The popular MNLI (Williams et al., 2018) dataset contains a spurious correlation between contradictions and negations ("not", "never", etc.), and some models learn to predict a contradiction whenever these expressions occur, regardless of the premise (Gururangan et al., 2018).

If we know what shortcuts are present, we can apply fixes like group DRO (Sagawa et al., 2019a). But how do we find them in the first place? We used our system to look for (alternative) descriptions of the differences between the two classes. We fed the `hypotheses` from the entailment class and those from the contradiction class to our system, which responded with "*contains a negative statement*"/"*has a negative verb*," revealing the spurious shortcut.

We also applied our system to a popular spam classification dataset (Gómez Hidalgo et al., 2006). We fed sentences from the two classes to our system, which tells us that the spam group "*has a higher number of hyperlinks*." To test whether such URLs influence downstream classifiers, we fed ten of our research communication messages with URLs to a RoBERTa-Large (Liu et al., 2019) model fine-tuned on this dataset (99% in-distribution accuracy). All 10 messages with URLs were classified as spam and were all classified as non-spam after removing the URLs.

Describing Distribution Shifts. We applied our system to describe distribution shifts for natural language tasks. For example, in contrast to MNLI, the SNLI dataset (Bowman et al., 2015) is based on image captions; therefore, our system says that SNLI "*describes a picture*." Naik et al. (2018) constructed another NLI dataset to stress test models' numerical reasoning ability; therefore, our system says that it "*contains a higher number of number words*." To take a different task, TwitterPPDB (Lan et al., 2017) and QQP[9] are both paraphrase detection datasets; the former is constructed by tweets while the latter is constructed by Quora questions; therefore, the system says that the former "*talks about a news story more*" while the latter "*contains a question*."

Labelling Text Clusters. Unsupervised algorithms generate semantically meaningful text clusters; however, researchers usually need to manually examine each of them to identify its semantics (Chang et al., 2009). Our system can automatically describe a text cluster by treating it as D_1 and all others as D_0.

[7] Of course, if our system can already perfectly verify the dataset descriptions by performing the task, then we might not need those datasets for training in the first place. However, even an imperfect AI system can help correct some human mistakes.

[8] This is an NLI-specific concept; we use a special font to distinguish it from "hypothesis" (Section 2) in our paper.

[9] https://www.kaggle.com/c/quora-question-pairs

Figure 6. For each text cluster (dot), we collect human annotations to compute $\text{CA}(h_s)$ for the descriptions by our expert (x-axis) and the top-5 by our system (y-axis). Our system is on par with the expert most of the time.

We compared our system to an expert on their ability to describe clusters. To create the clusters, we used RoBERTa-Base to embed the test set of wikitext-2 (Merity et al., 2016) (9992 sentences) and use the approach of Aharoni & Goldberg (2020) to create 64 clusters. We randomly selected ten of them for evaluation; for each of them, one of our authors read through 20 samples and wrote a natural language description s^*; we then asked him to read the top-5 descriptions by our system and pick the one $\hat{s}$ that he considered to be the best. We evaluated this author's performance by $\text{CA}(h_{s^*})$ and our system's performance by $\text{CA}(h_{\hat{s}})$, where we collected MTurks' annotations to compute $h_s(x_0, x_1)$.

Averaged across all clusters, our system achieves CA=0.8 while the expert achieves 0.77. Figure 6 shows the results for each cluster, and we found that our system is at least on par with the expert most of the time.

Discussion. In all the above applications, our system only informs the decisions of the stakeholders, who have the ultimate responsibility to decide if subjectivity can be approximated by "being review like", if specific correlations are bugs, or if the distribution shift is severe enough to take action. Our system also needs to improve to handle these applications robustly. For example, in the SPAM classification application, our verifier cannot verify whether a hyperlink exists as reliably as a rule-based classifier, while the 16x larger proposer does the heavy lifting. We hope scaling up can alleviate this problem in the future.

6. Related Work

Prompting and Zero-Shot Learning. Checking whether a hypothesis holds for a piece of text can be formulated as a Natural Language Inference (Bowman et al., 2015) or a Question Answering (Clark et al., 2019) task. Recent large pre-trained models can generalize to hypotheses significantly outside the training set (Khashabi et al., 2020), which allows us to re-rank candidate hypotheses. We expect future verifiers to be stronger as model sizes and the number of fine-tuned tasks grow (Wei et al., 2021; Sanh et al., 2021).

Our paper does not search for prompts to improve target task accuracy as in Shin et al. (2020); Mishra et al. (2021a); Rubin et al. (2021), which typically assume the target task is known or do not enforce prompt interpretability. Nevertheless, cross-pollination of ideas might be helpful.

To propose hypotheses, another plausible strategy is to find a continuous prompt (Li & Liang, 2021) first and then decode it to natural language by adding perplexity constraints (Song et al., 2020). However, Khashabi et al. (2021) suggests that this might be hard, given that soft prompts are not unique and heavily depend on initialization.

AI Safety and Scalable Oversight. Machine learning algorithms often fail on input patterns that are rare during train time. Typical examples include out-of-distribution samples (Hendrycks et al., 2021), unforeseen adversaries (Kang et al., 2019), spurious shortcuts (Sagawa et al., 2019b), and their interactions with the target population (Hardt et al., 2016; Hashimoto et al., 2018). Our system can monitor the differences between the train and test distribution to inform decision-makers. More broadly, we hope our automatically generated descriptions can help humans scalably oversee complicated machine learning systems (Amodei et al., 2016).

Learning a Predictor as Explanation. It is not new to discover statistical relationships in data by interpreting a learned hypothesis. Given real-valued features and a target variable, economists frequently run linear regressions and analyze the effect of each feature by interpreting the learned weights (Draper & Smith, 1998), sometimes adding sparsity constraints to focus on more important ones (Pati et al., 1993; Abbasi-Asl & Yu, 2020). Decision tree with a small list of if-then statements can also extract interpretable rules, e.g., to predict strokes (Letham et al., 2015). In comparison, our work focuses on discovering patterns in structured data (e.g., text) rather than real vectors; we also learn a natural language description, which might be easier for humans to understand, rather than a mathematical expression.

7. Discussion

Directions for Improving. Besides increasing model sizes (Section 4), our method would also benefit from: 1) running the proposer on different sets of samples and ensembling their outputs (Min et al., 2021), 2) using a proposer with a larger context size (Kitaev et al., 2020), 3) using a verifier with a symbolic component for numerical computations (Cobbe et al., 2021), and 4) using a retriever to verify information from external sources (Nakano et al., 2021). Additionally, Appendix E interprets our method under a unifying

probabilistic framework and discusses future directions using cycle consistency and self-supervision.

We currently evaluate only 54 distribution pairs by hand, which is time-consuming and small in scale. This might prevent future researchers from validating new methods quickly and reliably. We hope that automatic metrics more discriminative than Zhang et al. (2019) will help in the future, and that the number of distribution pairs for evaluation will increase as the community continues pooling datasets together (Mishra et al., 2021b; Sanh et al., 2021).

Inherent Ambiguities in Natural Language. Classical statistical analyses usually study mathematical hypotheses, whose meaning is uncontroversial and never changes; for example, people from different cultures and eras would all agree on what the number "7" means. However, there is no canonical way to interpret a natural language hypothesis: for example, Sap et al. (2021) finds that annotators with different social backgrounds disagree on the meaning of "*this sentence is offensive*." Future systems need to consider the listeners' background to prevent biases and ambiguities.

Expressiveness of the Descriptions. Our work only considers descriptions in the form of short natural language sentences. However, a single short sentence is sometimes insufficient to capture the multifaceted differences between two distributions, and hence multiple different descriptions are plausible. Currently, the user can choose to examine an arbitrary number of our systems' proposed hypotheses, sorted by Equation (5). Future work may consider logical compositions (e.g., conjunctions) of multiple hypotheses, hence making the descriptions more expressive.

Theoretically, natural language descriptions could be more expressive than those demonstrated in this paper, given that most of the human knowledge is communicated through natural language. However, they can still be limited, since humans can know more than they can verbalize (Polanyi's paradox, (Polanyi & Sen, 2009)).

Broader Applications. Our paper only considers text distributions, but language can also describe other modalities, such as vision (Radford et al., 2021), sound (Barchiesi et al., 2015), smell (Kiela et al., 2015), taste (Nozaki & Nakamoto, 2018), or motor sensations (Thomason et al., 2016). In principle, our framework can adapt to any experience humans can describe through language.

Our framework can also help answer broader scientific questions, for example: what does individual neuron represent in a deep learning model (Hernandez et al., 2022), how people from different parties discuss shooting events (Demszky et al., 2019), how people with different psychological signatures write (Boyd & Pennebaker, 2015), or how search queries change over time (Gentzkow et al., 2019).[10] We hope our method can help humans scalably discover new patterns in big data and complex systems.

Acknowledgement

The first author is funded by NSF-Simons Theorinet Grant (NSF Award #2031985). We thank OpenAI for providing inference and fine-tuning access to GPT-3 Davinci. We thank the Berkeley NLP group, the Steinhardt Group, and the anonymous reviewers for their feedbacks on the paper. We thank Dong Yang for training a new proposer by fine-tuning T5, and the TPU Research Cloud (TRC) program for providing computational resources.

References

Abbasi-Asl, R. and Yu, B. Structural compression of convolutional neural networks, 2020.

Aharoni, R. and Goldberg, Y. Unsupervised domain clusters in pretrained language models, 2020.

Almeida, T., Hidalgo, J. M. G., and Silva, T. P. Towards sms spam filtering: Results under a new dataset. *International Journal of Information Security Science*, 2(1):1–18, 2013.

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Barbieri, F., Camacho-Collados, J., Espinosa Anke, L., and Neves, L. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1644–1650, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.148. URL https://aclanthology.org/2020.findings-emnlp.148.

Barchiesi, D., Giannoulis, D., Stowell, D., and Plumbley, M. D. Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Processing Magazine*, 32(3):16–34, 2015.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://aclanthology.org/D15-1075.

[10] Zeng & Wagner (2002) note that the volume of searches or web hits seeking information related to a disease may be a strong predictor of its prevalence.

Boyd, R. L. and Pennebaker, J. W. Did shakespeare write double falsehood? identifying individuals by creating psychological signatures with text analysis. *Psychological science*, 26(5):570–582, 2015.

Bragg, J., Cohan, A., Lo, K., and Beltagy, I. Flex: Unifying evaluation for few-shot nlp. *Advances in Neural Information Processing Systems*, 34, 2021.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J., and Blei, D. Reading tea leaves: How humans interpret topic models. *Advances in neural information processing systems*, 22, 2009.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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)*, pp. 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300.

Cobbe, K., Kosaraju, V., Bavarian, M., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Demszky, D., Garg, N., Voigt, R., Zou, J. Y., Gentzkow, M., Shapiro, J. M., and Jurafsky, D. Analyzing polarization in social media: Method and application to tweets on 21 mass shootings. In *NAACL*, 2019.

Draper, N. R. and Smith, H. *Applied regression analysis*, volume 326. John Wiley & Sons, 1998.

Gao, T., Fisch, A., and Chen, D. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3816–3830, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.295. URL https://aclanthology.org/2021.acl-long.295.

Gentzkow, M., Kelly, B., and Taddy, M. Text as data. *Journal of Economic Literature*, 57(3):535–74, 2019.

Gómez Hidalgo, J. M., Bringas, G. C., Sánz, E. P., and García, F. C. Content based sms spam filtering. In *Proceedings of the 2006 ACM symposium on Document engineering*, pp. 107–114, 2006.

Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S., and Smith, N. A. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 107–112, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2017. URL https://aclanthology.org/N18-2017.

Hardt, M., Megiddo, N., Papadimitriou, C., and Wootters, M. Strategic classification. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*, pp. 111–122, 2016.

Hashimoto, T., Srivastava, M., Namkoong, H., and Liang, P. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pp. 1929–1938. PMLR, 2018.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8349, 2021.

Hernandez, E., Schwettmann, S., Bau, D., Bagashvili, T., Torralba, A., and Andreas, J. Natural language descriptions of deep visual features. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2201.11114.

Kang, D., Sun, Y., Hendrycks, D., Brown, T., and Steinhardt, J. Testing robustness against unforeseen adversaries. *arXiv preprint arXiv:1908.08016*, 2019.

Khashabi, D., Min, S., Khot, T., Sabharwal, A., Tafjord, O., Clark, P., and Hajishirzi, H. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1896–1907, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.171. URL https://aclanthology.org/2020.findings-emnlp.171.

Khashabi, D., Lyu, S., Min, S., Qin, L., Richardson, K., Singh, S., Welleck, S., Hajishirzi, H., Khot, T., Sabharwal, A., et al. Prompt waywardness: The curious case of discretized interpretation of continuous prompts. *arXiv preprint arXiv:2112.08348*, 2021.

Kiela, D., Bulat, L., and Clark, S. Grounding semantics in olfactory perception. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 231–236, 2015.

Kitaev, N., Kaiser, L., and Levskaya, A. Reformer: The efficient transformer. *ArXiv*, abs/2001.04451, 2020.

Lan, W., Qiu, S., He, H., and Xu, W. A continuously growing dataset of sentential paraphrases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1224–1234, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1126. URL https://aclanthology.org/D17-1126.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Letham, B., Rudin, C., McCormick, T. H., and Madigan, D. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3), Sep 2015. ISSN 1932-6157. doi: 10.1214/15-aoas848. URL http://dx.doi.org/10.1214/15-AOAS848.

Li, X. and Roth, D. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P11-1015.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models, 2016.

Mihaylova, T., Karadzhov, G., Atanasova, P., Baly, R., Mohtarami, M., and Nakov, P. SemEval-2019 task 8: Fact checking in community question answering forums. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 860–869, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/S19-2149. URL https://aclanthology.org/S19-2149.

Min, S., Lewis, M., Hajishirzi, H., and Zettlemoyer, L. Noisy channel language model prompting for few-shot text classification. *arXiv preprint arXiv:2108.04106*, 2021.

Mishra, S., Khashabi, D., Baral, C., Choi, Y., and Hajishirzi, H. Reframing instructional prompts to gptk's language. *arXiv preprint arXiv:2109.07830*, 2021a.

Mishra, S., Khashabi, D., Baral, C., and Hajishirzi, H. Cross-task generalization via natural language crowdsourcing instructions. 2021b.

Naik, A., Ravichander, A., Sadeh, N., Rose, C., and Neubig, G. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 2340–2353, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1198.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Nozaki, Y. and Nakamoto, T. Predictive modeling for odor character of a chemical using machine learning combined with natural language processing. *PloS one*, 13 (6):e0198475, 2018.

Pang, B. and Lee, L. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pp. 271–278, Barcelona, Spain, July 2004. doi: 10.3115/1218955.1218990. URL https://aclanthology.org/P04-1035.

Pati, Y., Rezaiifar, R., and Krishnaprasad, P. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pp. 40–44 vol.1, 1993. doi: 10.1109/ACSSC.1993.342465.

Polanyi, M. and Sen, A. *The tacit dimension*. University of Chicago press, 2009.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Rubin, O., Herzig, J., and Berant, J. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*, 2021.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *ArXiv*, abs/1911.08731, 2019a.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019b.

Sanh, V., Webson, A., Raffel, C., Bach, S. H., Sutawika, L. A., Alyafeai, Z., Chaffin, A., Stiegler, A., Scao, T. L., Raja, A., Dey, M., BARI, M. S., Xu, C., Thakker, U., Sharma, S. S., Szczechla, E., Kim, T., Chhablani, G., Nayak, N. V., Datta, D., Chang, J., Jiang, M. T.-J., Wang, H., Manica, M., Shen, S., Yong, Z. X., Pandey, H., Bawden, R., Wang, T., Neeraj, T., Rozen, J., Sharma, A., Santilli, A., Févry, T., Fries, J. A., Teehan, R., Biderman, S. R., Gao, L., Bers, T. G. O., Wolf, T., and Rush, A. M. Multitask prompted training enables zero-shot task generalization. *ArXiv*, abs/2110.08207, 2021.

Sap, M., Swayamdipta, S., Vianna, L., Zhou, X., Choi, Y., and Smith, N. A. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. *arXiv preprint arXiv:2111.07997*, 2021.

Shin, T., Razeghi, Y., Logan IV, R. L., Wallace, E., and Singh, S. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4222–4235, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.346. URL https://aclanthology.org/2020.emnlp-main.346.

Song, C., Rush, A., and Shmatikov, V. Adversarial semantic collisions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4198–4210, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.344. URL https://aclanthology.org/2020.emnlp-main.344.

Thomason, J., Sinapov, J., Svetlik, M., Stone, P., and Mooney, R. J. Learning multi-modal grounded linguistic semantics by playing" i spy". In *IJCAI*, pp. 3477–3483, 2016.

Wang, Y., Mishra, S., Alipoormolabashi, P., Kordi, Y., Mirzaei, A., Arunkumar, A., Ashok, A., Dhanasekaran, A. S., Naik, A., Stap, D., et al. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *arXiv preprint arXiv:2204.07705*, 2022.

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*, 2018.

Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. *ArXiv*, abs/2109.01652, 2021.

Williams, A., Nangia, N., and Bowman, S. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://aclanthology.org/N18-1101.

Wortsman, M., Ilharco, G., Li, M., Kim, J. W., Hajishirzi, H., Farhadi, A., Namkoong, H., and Schmidt, L. Robust fine-tuning of zero-shot models. *arXiv preprint arXiv:2109.01903*, 2021.

Yin, W., Hay, J., and Roth, D. 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)*, pp. 3914–3923, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1404. URL https://aclanthology.org/D19-1404.

Zeng, X. and Wagner, M. Modeling the effects of epidemics on routinely collected data. *Journal of the American Medical Informatics Association*, 9(Supplement_6):S17–S22, 2002.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pp. 649–657, Cambridge, MA, USA, 2015. MIT Press.

Zhong, R., Lee, K., Zhang, Z., and Klein, D. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 2856–2878, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.findings-emnlp.244.

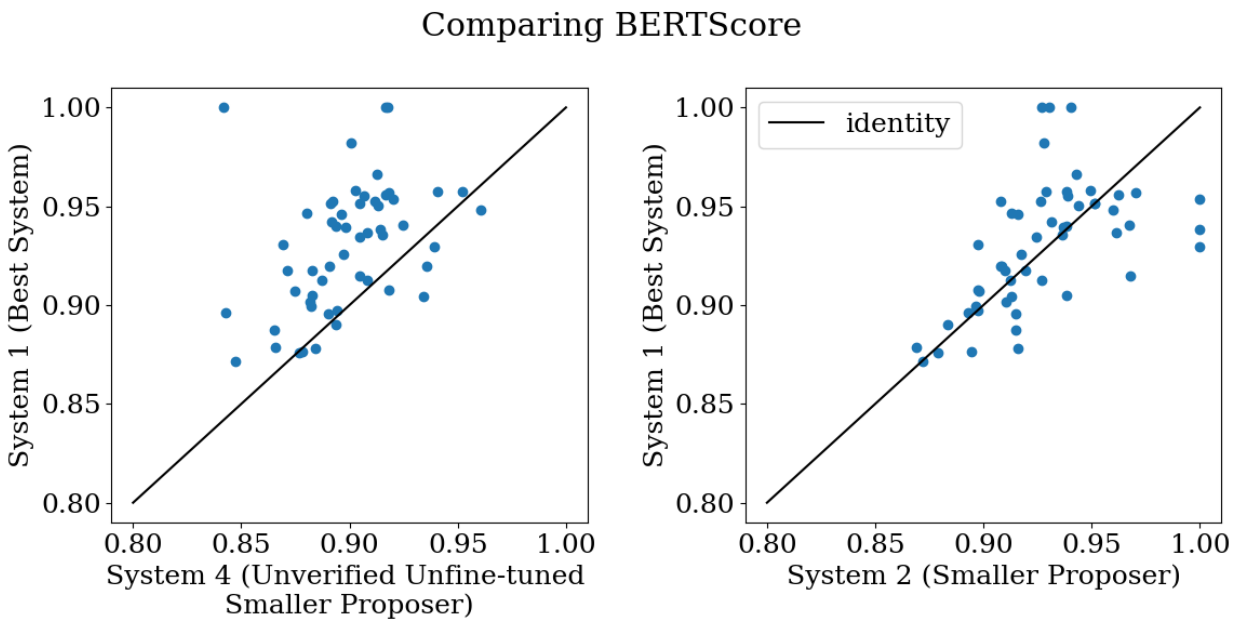

Figure 7. We compare System ① and ④ with BERTScore (Zhang et al., 2019) on the left and ① and ② on the right. Each dot represents a binary task the y/x value is the performance of a system-generated hypothesis evaluated by BERTScore. Our best system ① is clearly outperforming the worst ④ (left), but the difference between the 1st and the 2nd system becomes hard to tell (right).

A. Using BERT-score for Evaluation

We generate scatter plots to compare our best system ① with the worst system ④ and our second best system ② in Figure 7 to double-check that we used the metric correctly. Despite the the small absolute difference (3%) in the reported numbers, BERTScore does robustly tell the difference between system system ① and ④. On the other hand, however, it has trouble discriminating our first and second best system: after squinting at the results hard enough, we find that ① outperforms ② by 0.3 points on average; across binary tasks, ① outperforms ② more than 0.5 points for 46% of the time, while ② outperforms ① by more than 0.5 points 31% of the time. Therefore, BERTscore does agree that ① is better than ②. Nevertheless, we felt that this metric is not discriminative and interpretable enough, so we had to rely on human evaluation (Section 4).

B. Top-K Performance

We calculate the performance of the top-K descriptions by our system according to our manual evaluation, where K ranges from 1 to 5. Table 2 shows the results.

	① Best	② Smaller	③ No Fine-tune	④ No Re-rank	⑤ Memorize
A	13/26/28/30/31	14/17/21/21/22	6/ 8/10/10/11	2/ 3/ 3/ 3/ 4	2/ 3/ 5/ 5/ 5
B	19/14/13/11/10	10/10/8/10/11	5/ 6/ 6/ 6/ 6	1/ 0/ 0/ 0/ 0	5/ 5/ 5/ 5/ 5
C	16/ 8/ 7/ 7/ 7	17/14/12/12/10	7/ 9/10/10/10	2/ 3/ 5/ 5/ 6	16/19/19/21/21
D	6/ 6/ 6/ 6/ 6	13/13/13/11/11	36/31/28/28/27	49/48/46/46/44	31/27/25/23/23

Table 2. Similar to Table 1, ① represents our best system with the largest fine-tuned proposer, ② with a smaller fine-tuned proposer, ③ without fine-tuning, ④ without re-ranking, and ⑤ with the memorization proposer. For each task, we choose the top-K descriptions according to the verifier, and find the highest human rating among the top-K; we then count how often each rating occurs across 54 binary tasks. We report K from 1 to 5 separated by "/" in each cell. Notice that only row (A) is guaranteed to increase as k increases, since we are counting the frequency of the highest ranking; e.g., using five rather than one description can change the highest rating from (B) to (A), thus decreasing the count of (B).

We report the statistical significance of comparing different systems by their best-of-top-5 descriptions. We first examine how often a system generates a "A" level description across 54 binary classification datasets; as a result ① > ② with $p = 9.3 \times 10^{-3}$, ② > ③ with $p = 3.2 \times 10^{-3}$, ③ > ④ with $p = 2.5 \times 10^{-2}$, and ② > ⑤ with $p = 4.2 \times 10^{-5}$. We next examine how often a system generates a "A" or "B" level description; as a result ① > ② with $p = 9.7 \times 10^{-3}$, ② > ③ with $p = 1.7 \times 10^{-4}$, ③ > ④ with $p = 5.4 \times 10^{-4}$, and ② > ⑤ with $p = 1.6 \times 10^{-6}$.

C. Example Positive Samples in Section 3.3

We list a few example hypotheses along with positive sample text (i.e., GPT-3 generated texts that are likely to satisfy the hypotheses), indxed with bullet points.

Hypothesis: *contains internet emoticons.*

- *:) - I'm happy*
- *:) :D :o :(*
- *(:-I'm so excited to tell you about my plans for the future.): I hope they work out!*

Hypothesis: *contains an acronym or abbreviation that might be used online.*

- *SMH is an acronym for shaking my head. It is used when someone reacts to something funny or shocking.*
- *OMG. I think itś my new BF.*
- *IDK - I don't knowTTYL - talk to you later.*

Hypothesis: *is about school life.*

- *Entering the courtyard of the student center, I had a feeling that there was something different about today.*
- *The first day of school is always nerve-wracking. You're the new kid, and you have to make friends fast.*
- *I came to this country in the summer of 2010. I attended a public high school in my new country and was really scared because I did not know what to expect from it.*

Hypothesis: *is about money*.

- *It's so easy to think about all of the things that you could do with your life if you had more disposable income, but it's hard work to make more than enough. I'm not sure what my goal is when it comes to my finances, but I know that whatever happens in the future has got to be better than this.*
- *There's a recession going on and cash flow is scarce.*
- *Money is very important in today's society. It lets you do what you want to do and nothing can stop you.*

Hypothesis: *describes cooking.*

- *If you can perfect a recipe, it's amazing. I love being able to read through a recipe and have the confidence that I have all of the ingredients listed.*
- *I love to watch the way that each ingredient has their own value, and how they all come together to make a dish.*
- *One rainy night, I decided to make spaghetti.*

D. Example Descriptions and Their Ratings

For each binary task, we present the human annotation, the best descriptions from the top-5 descriptions by system ①, and our similarity rating in Table 3.

Human Annotations	Descriptions by Our System	Rating
is religious	*is religious*	(A)
is against feminism	*is a criticism of feminism*	(A)
is about math or science	*is about science*	(B)
asks about a location	*asks about a location*	(B)
contains a good movie review	*praises the film*	(A)
is offensive	*is a Twitter hate-rant*	(C)
is related to computer science	*is a description of a computer-based system*	(B)
is against environmentalist	*is a denial of climate change science*	(C)
is against Hillary	*is a criticism of Hillary Clinton*	(A)
is pro-choice	*advocates for abortion rights*	(A)
is about research in statistics	*presents a research on a statistical topic*	(A)
is related to infrastructure	*mentions natural disaster*	(D)
is about entertainment	*is related to the entertainment industry*	(B)
is environmentalist	*shows an environmental concern*	(A)
is related to health	*is about the topic of "health"*	(A)
contains irony	*is sarcastic in tone*	(A)
supports hillary	*is a positive sentence about Hillary Clinton*	(A)
contains a definition	*is about learning something new*	(B)
is related to terrorism	*is about terrorism*	(A)
expresses a need for water	*is about water shortage*	(A)
involves crime	*is describing clashes*	(C)
is related to sports	*is about sports*	(A)
is related to a medical situation	*is related to the topic of health*	(B)
describes a situation where people need food	*is about the situation of food shortage*	(A)
is pro-life	*can be categorized as a pro-life message*	(A)
contains subjective opinions	*is a review of a movie*	(D)
asks for an opinion	*is asking for help*	(D)
is more related to computers or internet	*is about computer*	(B)
expresses need for utility, energy or sanitation	*contains a word related to electricity*	(C)
is sports related	*is about a topic related to sports*	(A)
asks for a number	*contains a question ...*[*]	(A)
describes a situation where people need to evacuate	*describes a situation involving evacuation*	(A)
is a more objective description of what happened	*is a plot summary of a film*	(D)
is physics research	*is about a physics research*	(A)
is about world news	*is a news article on a country*	(C)
looks more like business news	*deals with economic news*	(A)
describes a situation where people need shelter	*is about earthquake*	(C)
is a spam	*is a "spam" SMS*	(A)
contains grammar errors	*is grammatically incorrect*	(A)
asks about an entity	*contains a word that rhymes with "tree"*	(D)
is about math research	*is about a mathematics research paper*	(A)
supports feminism	*is in support of feminism*	(A)
asks for factual information	*is a request for immigration related questions*	(D)
is more political	*is about politics*	(A)
is against religion	*has a negative connotation towards religion*	(A)

Table 3. For each binary task, we present the human annotation, the best descriptions from the top-5 descriptions by system ①, and our similarity rating, with (A) being the highest (Section 4).

[*]: "*contains a question that can be answered with a number*"; truncated from the column to save sapce.

E. A Unifying View

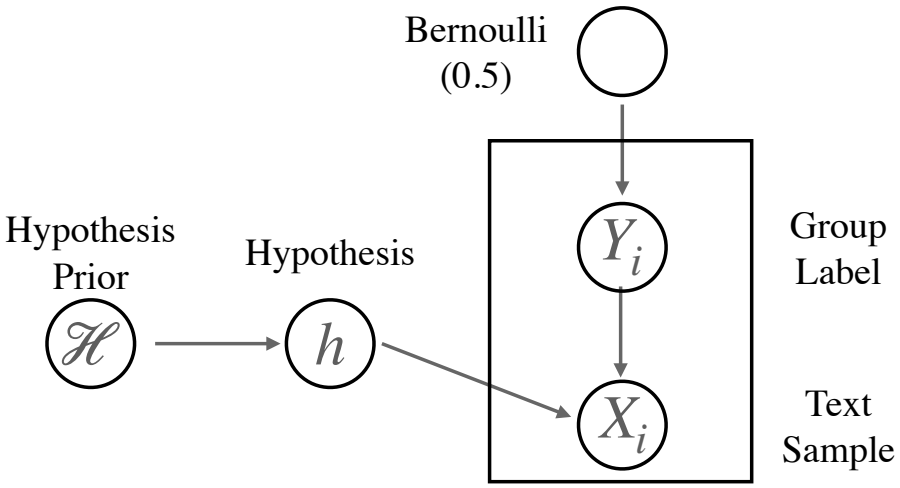

Figure 8. A unifying graphical model interpretation of our framework, where the verifier, the proposer, and the conditional generator can be all written as posterior estimators.

We present a unifying graphical model for the hypothesis h, the samples $X_{1...K}$, and the group labels $Y_{1...K}$ (Figure 8), where $Y_i \in \{0, 1\}$ indicating whether X_i is from distribution D_0 or D_1. Although we did not implement it in our paper, we find it helpful as a mental model to generate future research directions. The graphical model factorizes as:

$$p(h, X_{1...K}, Y_{1...K}) = p(h) \prod_{i=1}^{K} p(X_i|Y_i, h)P(Y_i). \tag{6}$$

Under this framework, the goal of generating a natural language hypothesis becomes posterior estimation:

$$p(h|X_{1...K}, Y_{1...K}) \propto p(h) \prod_{i=1}^{K} p(Y_i|X_i, h). \tag{7}$$

The verifier can also be written as $\hat{p}(Y|X, h)$, the proposer as $\hat{p}(h|X_{1...5}, Y_{1...5})$, the conditional generator as $\hat{p}(X|Y, h)$, and the hypothesis space as a prior $\hat{p}(h)$,[11] all of which can be directly approximated by a fine-tuned language model. To fine-tune these approximators, it suffices to obtain the complete data h, X_*, and Y_*. Our work only fine-tuned the verifier and the proposer, but the conditional generator $\hat{p}(X|Y, h)$ and $\hat{p}(h)$ can also be fine-tuned. We only supervised $\hat{p}$ through querying human about $p(Y|X, h)$, but other forms of queries are also possible. Finally, it is not necessary to follow the recipe in our paper to generate the complete data: we could alternatively first generate X and h, and then generate Y accordingly. Human supervision is also not strictly necessary to generate the complete data: we can purely sample data from some approximators to fine-tune other ones, thus achieving self-supervision through cycle consistency.

F. Original Sources of the Binary Tasks

The 54 binary tasks are from Maas et al. (2011), Yin et al. (2019), Barbieri et al. (2020), Zhang et al. (2015), Yin et al. (2019), Warstadt et al. (2018), Almeida et al. (2013), Pang & Lee (2004), Li & Roth (2002), Mihaylova et al. (2019), and an abstract classification dataset[12].

[11] which our paper defines through manual curation of the hypothesis and modelled as a uniform distribution during inference.

[12] https://www.kaggle.com/abisheksudarshan/topic-modeling-for-research-articles?select=Train.csv

G. Notes on Code and Model Release

We release our code and data with the following link https://github.com/ruiqi-zhong/DescribeDistributionalDifferences.

We cannot directly share our GPT-3 based proposer, since it has to be accessed through the OpenAI API using our own key. To make it easier for other researchers to use our system, we trained another proposer by fine-tuning T5 (Raffel et al., 2019) on a mixture of 1) our collected data, and 2) a large dataset Wang et al. (2022) to learn to follow task instructions. Additionally, we implemented the ensembling approach mentioned in Section 7. Though we have not rigorously benchmarked the new proposer, it seems to be roughly comparable to the proposer based on GPT-3 Davinci (175B parameters), and it can be openly shared, downloaded, and run locally.