An Information-theoretic Approach to Prompt Engineering Without Ground Truth Labels

Anonymous ACL submission

Abstract

Pre-trained language models derive substantial linguistic and factual knowledge from the massive corpora on which they are trained, and prompt engineering seeks to align these models to specific tasks. Unfortunately, existing prompt engineering methods require significant amounts of labeled data, access to model parameters, or both. We introduce a new method for selecting prompt templates *without labeled examples* and *without direct access to the model*. Specifically, over a set of candidate templates, we choose the template that maximizes the mutual information between the input and the corresponding model output. Across 8 datasets representing 7 distinct NLP tasks, we show that when a template has high mutual information, it also has high accuracy on the task. On the largest model, selecting prompts with our method gets 90% of the way from the average prompt accuracy to the best prompt accuracy and requires no ground truth labels.

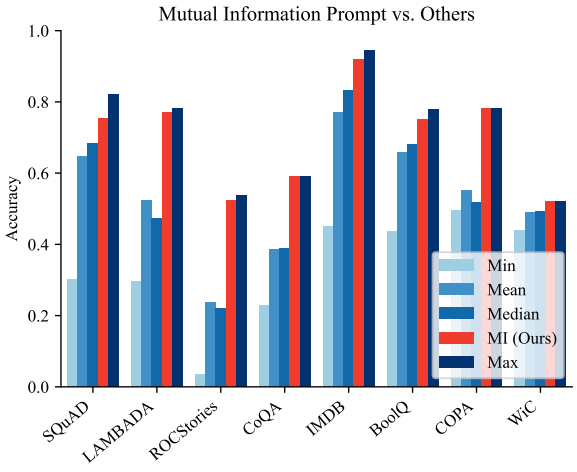

Figure 1: Performance of template selected by our maximum mutual information method (MI) compared to the the worst, mean, median, and best prompt on GPT-3 Davinci (175B). Our method performs at almost oracle levels, without labels or access to model weights.

1 Introduction

It is well-known that large pre-trained language models (LMs) learn substantial linguistic (Liu et al., 2019; Amrami and Goldberg, 2018) and factual world knowledge (Petroni et al., 2020; Bosselut et al.; Bouraoui et al.; Zuo et al., 2018), achieving state-of-the-art performance on classic NLP tasks like closed-book question-answering, sentiment analysis, and many other tasks (Radford et al., 2019; Devlin et al., 2019; Raffel et al., 2019). The largest models can do this in a few-shot way–being trained only with generic, semi-supervised objectives and "taught" tasks with just instructions and a few examples of the task provided via a natural language "prompt" in the context window (Brown et al., 2020). This suggests that pre-training equips them to potentially do many tasks that can be formulated as natural language generation, if only they can be primed in the right way.

Such priming is not a trivial task. The few-shot learning breakthrough can give the impression that if the LM is given a sensible prompt, the model will "understand" what is meant and perform well on the task if it has the capacity. However, LMs can generate substantially different probability distributions–and thus text–given two distinct prompts that appear semantically invariant (e.g., alternative ordering of options, lexical changes like capitalization, and general rephrasing (Zhao et al., 2021; Lu et al., 2021)). This can lead to surprisingly high variance in performance from prompt to prompt. Clearly, some prompts are better than others for aligning a model to a task.

Prompt engineering is a nascent field that aims to find such aligning prompts (Reynolds and McDonell, 2021). While "prompt" refers to any language passed to the model via the context window, a *template* refers to a NL scaffolding filled in according to raw data, resulting in a prompt. Thus, prompt engineering includes finding high-quality templates (i.e., those with high test accuracy). Gen-

erally, this is done by validation set accuracy optimization: a template is chosen from a set of candidates given their performance on a set of labeled examples. In order to do this reliably, many labeled examples are needed. These can be challenging to procure for some tasks and impossible for others. Some recent methods optimize prompts using backpropagation, which requires access to model weights. By using mutual information, our method allows prediction of a prompt's performance without labels or access to model parameters.

Mutual information (MI) is a metric that quantifies the shared information between two random variables (see Section 3.2). We demonstrate that the mutual information between a prompt and a language model's output can serve as a useful surrogate for the test accuracy of a template. To justify this method, we generate a diverse set of 20 templates per dataset and show that for each template, mutual information and accuracy are highly correlated. These results are strongest on the largest models we study and hold across eight datasets representing seven NLP tasks; our method chooses prompts that, on average, get 90% of the way from mean accuracy to maximum accuracy and even selects the best prompt on three of eight datasets.

This suggests that, across a variety of NLP tasks, mutual information can be used to select one of the best prompts from a set of candidate prompts, even without making use of model weights or ground truth labels. In the following pages, we outline each step of our general method for generating and evaluating templates so that it can easily be ported to any other task. Code is available at [URL removed for anonymity - code attached with submission].

2 Related Work

The promise of language models and the challenge of aligning them has given rise to the field of "prompt engineering", which seeks to construct the best prompt given a task and a language model (Liu et al., 2021a). The best performance on prompt engineering is often achieved using backpropagation in continuous prompt embedding space (Lester et al., 2021; Li and Liang, 2021; Gu et al., 2021; Liu et al., 2021b; Zhang et al., 2021) in contrast to generating a discrete set of prompts by hand and testing them. While optimizing in continuous prompt space via backprop allows for similar performance to model-tuning (at least at higher model sizes) (Lester et al., 2021), not all models are publicly available. Thus, these methods are only feasible for those who have direct access to the model and can perform backprop on it. Prompts optimized in continuous space are also not interpretable in natural language, making it harder to transfer insights from prompts that work well for one task to another task. These methods also require labeled examples, while ours does not.

Other selection protocols not based on gradient flow can include cross-validation or minimum description length, as in (Perez et al., 2021). These methods yield prompts that perform marginally better than average in terms of test accuracy.

Mutual information has been used in n-gram clustering, part-of-speech tagging, probing classifiers, and LM training objective reframing (Brown et al., 1992; Stratos, 2019; Voita and Titov, 2020; Kong et al., 2019). Ours is the first work of which we're aware to apply MI to prompt engineering.

3 Methods

At the most abstract, our method is as follows (see Appendix A for a more thorough description):

1. Generate a set of K prompt templatizing functions.
2. Playground a couple of examples to ensure that templates give roughly expected output.
3. Estimate mutual information for each template given a set of inputs $\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_N \sim X$.
4. Choose template(s) based on mutual information and perform inference.

We find it useful to unify all the tasks we study within a single framework, which we describe in Section 3.1. We also justify our use of mutual information as a surrogate for prompt quality and specify how we estimate it in Section 3.2.

3.1 Task Definition

In order to demonstrate our method's widespread applicability and general effectiveness, we validate it across many datasets and tasks. This requires us to estimate mutual information and accuracy, and this is most straightforward in the case where, given a context, a language model produces just one probability distribution $P(\mathbf{t}_n|\text{context} =$

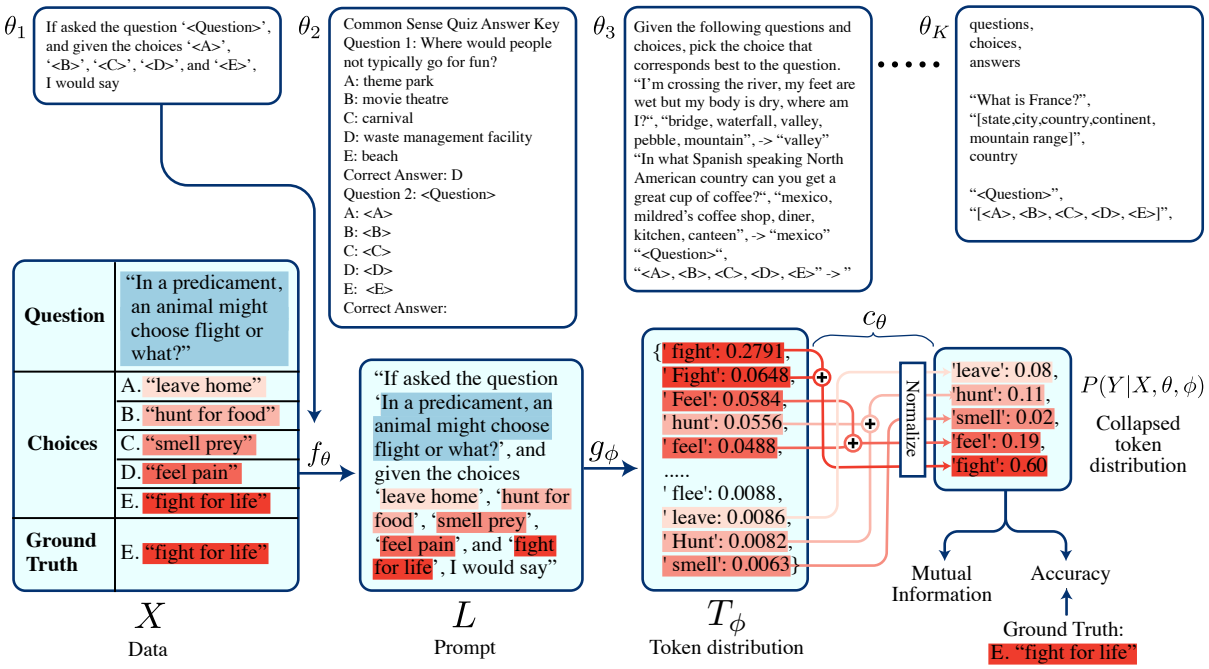

Figure 2: We choose $\theta \in \{\theta_i\}_{i=1}^{K}$ and templatize a sampled instance from the dataset X. We pass this prompt through the language model via g_ϕ, yielding a probability distribution over the model's tokens T_ϕ. The collapsing function c_θ sums the weight given to each token corresponding to each possible answer $y \in Y$ and normalizes, giving a probability distribution $P(Y|\mathbf{x}_i)$, which we can use to estimate mutual information or obtain a guess for y_i.

$\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_{n-1}$). This is in contrast to other experimental setups that use multi-token sampling methods (e.g., beam search). Any NLP task is tractable in this framework so long as the output space consists of a set of options that each start with a unique token. In this case, the language model can "give" an answer by assigning probability to tokens that begin giving each of these answers (invariant to lexical variation like capitalization and leading/trailing spaces). While, for open-ended tasks, this method might artificially inflate accuracy if the model starts to give a wrong answer that happens to start with the same token as the correct one, we find that this difference is small and does not affect our results.[1] Irrelevant tokens (with which none of the desired answers begin) are ignored, and the resulting collapsed probabilities are normalized. We term this approach *One-token Response* (OTR). Although our method isn't limited to OTR tasks, we choose tasks that can be cast as OTR tasks for simplicity and to reduce computational expense. Many NLP tasks fit within this framework, although a few do not (e.g., machine translation and summarization). This basic approach is in common use (Brown et al., 2020), but we formalize it for clarity below.

Generally, the OTR framework casts a natural language task as a classification problem with raw data input $\mathbf{x}_i \in X$ and output $P(Y|\mathbf{x}_i)$, a probability distribution over targets. In order to use a language model ϕ for this task, a templatizing function $f_\theta : X \to L$ is needed to map raw data into natural language prompts. $g_\phi : L \to T_\phi$ maps prompts to a probability distribution over T_ϕ, the token set represented by the model tokenizer. Finally, a collapsing function $c_\theta : T_\phi \to P(Y|\mathbf{x}, \theta, \phi)$ yields an estimate of $P(Y|X)$:

$$P(Y|\mathbf{x}, \theta, \phi) = c_\theta(g_\phi(f_\theta(\mathbf{x}))), \mathbf{x} \in X \quad (1)$$

We also refer to $P(Y|\mathbf{x}, \theta, \phi)$ as $P(Y|f_\theta(\mathbf{x}))$.

The above pipeline can be specified in many ways using different θ and ϕ (see Figure 2), which will result in different accuracies. Our ultimate aim is to select the best θ given ϕ. Whereas past prompt engineering methods rely on scores calculated by comparing model answers and ground truth, our method selects θ by maximizing mutual information, which requires no ground truth labels.

[1] Our open-ended datasets are SQuAD, LAMBADA, and ROCStories, and none of these seemed more likely than ROCStories to exhibit this issue. We reran our experiment on ROCStories by sampling with temperature 0 until reaching a space, and only counted responses as accurate if they exactly matched the corresponding ground truth labels. Results were virtually unchanged: accuracy decreased by only 0.03 on average, and the correlation between mutual information and test accuracy increased by 0.04, from 0.68 to 0.72.

3.2 Mutual Information

Mutual information is a measure of the amount of shared information between two random variables (Cover and Thomas, 2006); in other words, it is the reduction in entropy that is observed in one random variable when the other random variable is known.

We expect mutual information to serve as a good criterion for comparing prompts. Previous work has shown that large networks trained with cross-entropy loss are calibrated (e.g., a 60% confidence corresponds to a 60% chance of the model being correct) when in the early-stopped ($\sim$ 1 epoch) regime (Ji et al., 2021), but become miscalibrated in the overfit regime (Nakkiran and Bansal, 2020). According to (Brown et al., 2020), GPT-3 was trained for a different number of epochs on each corpus in its training data. We calculate it was trained for an average of 1.57 epochs, so we have reason to believe that GPT-3 is generally well-calibrated. Thus, we postulate that a prompt that elicits a very confident response (high mutual information) from the language model is more likely than a less confident prompt to score well.

We denote the mutual information between random variables X and Y as $I(X;Y)$ and the entropy of X as $H(X) = -\int_{\mathbf{x}\in X} P(\mathbf{x}) \log(P(\mathbf{x}))d\mathbf{x}$. The mutual information between X and Y is defined as $D_{\text{KL}}(P_{(X,Y)}||P_X \otimes P_Y)$, and can be rewritten as $H(Y) - H(Y|X)$ (the reduction in entropy in Y given knowledge of X).

Using the OTR framework, we fix a model ϕ and generate a diverse set of K prompt templatizing functions $f_{\theta_1}, f_{\theta_2}, ..., f_{\theta_K}$ along with their corresponding collapsing functions c_{θ_k}. Treating $f_\theta(X) := \{f_\theta(\mathbf{x}), \mathbf{x} \in X\}$ as a random variable, we can calculate $I(f_\theta(X); Y)$ and use it as a criterion for selecting prompt templatizing functions with which to do inference.

We hypothesize that a θ_i with higher mutual information will align a language model to a task better than a θ_j with lower mutual information. Formally, we select $\hat{\theta} = \text{argmax}_\theta\{I(f_\theta(X); Y)\}$.

Mutual information is estimated as:

$$I\left(f_\theta(X); Y\right) = H(Y) - H(Y|f_\theta(X)) \quad (2)$$

where each term is estimated in expectation using draws $\mathbf{x}_i \sim X$ and Equation 1 as follows:

$$H(Y) \approx H\left(\frac{1}{N}\sum_{i=1}^{N} P(Y|f_\theta(\mathbf{x}_i))\right) \quad (3)$$

Dataset	Task	$\lvert Y\rvert$	Base Acc.	Size N_{all}
SQuAD	Open Book QA	$\lvert T_\phi\rvert$	~ 0	16K
LAMBADA	Cloze	$\lvert T_\phi\rvert$	~ 0	5K
ROCStories	Cloze	$\lvert T_\phi\rvert$	~ 0	52K
CoQA	Closed Book QA	5	0.2	9K
IMDB	Sentiment Analysis	2	0.5	50K
BoolQ	Reading Comprehension	2	0.5	16K
COPA	Choice of Positive Alternatives	2	0.5	1K
WiC	Word in Context	2	0.5	5K

Table 1: All datasets used in our experiments. $|Y|$ is the size of the label space and N_{all} is the size of the dataset we sample from (after any modifications).

$$H(Y|f_\theta(X)) \approx \frac{1}{N}\sum_{i=1}^{N} H(P(Y|f_\theta(\mathbf{x}_i))) \quad (4)$$

Thus, the marginal entropy $H(Y)$ is the entropy of the mean of the conditional distributions, and the conditional entropy $H(Y|f_\theta(X))$ is the mean of entropies of the individual conditional distributions.

This definition gives us another reason to expect that mutual information will work well. Since mutual information is the marginal entropy minus the conditional entropy, maximizing mutual information is equivalent to maximizing marginal entropy and minimizing conditional entropy. Thus, MI is high for templates that are, on average, less biased towards any given answer (high marginal entropy) and templates with outputs the model is confident about (low conditional entropy). These attributes are desirable in constructing prompts, and we postulate that maximizing mutual information will yield a well-aligned template.

Looking at it another way, by the data processing inequality (Cover and Thomas, 2006), $I(f_\theta(X); Y) \leq I(X; Y)$. Thus, $I(f_\theta(X); Y)$ gives a lower bound for $I(X; Y)$, and the highest mutual information is the tightest lower bound. The prompt corresponding to this lower bound preserves the most information between X and Y.

4 Experimental Setup

4.1 Datasets

We validate the efficacy of our prompt engineering method with experiments on eight well-known NLP datasets[2]–SQuAD2.0 (Rajpurkar et al., 2018),

[2]Datasets are listed in descending order here and throughout the paper, first by $|Y|$, and then by method performance.

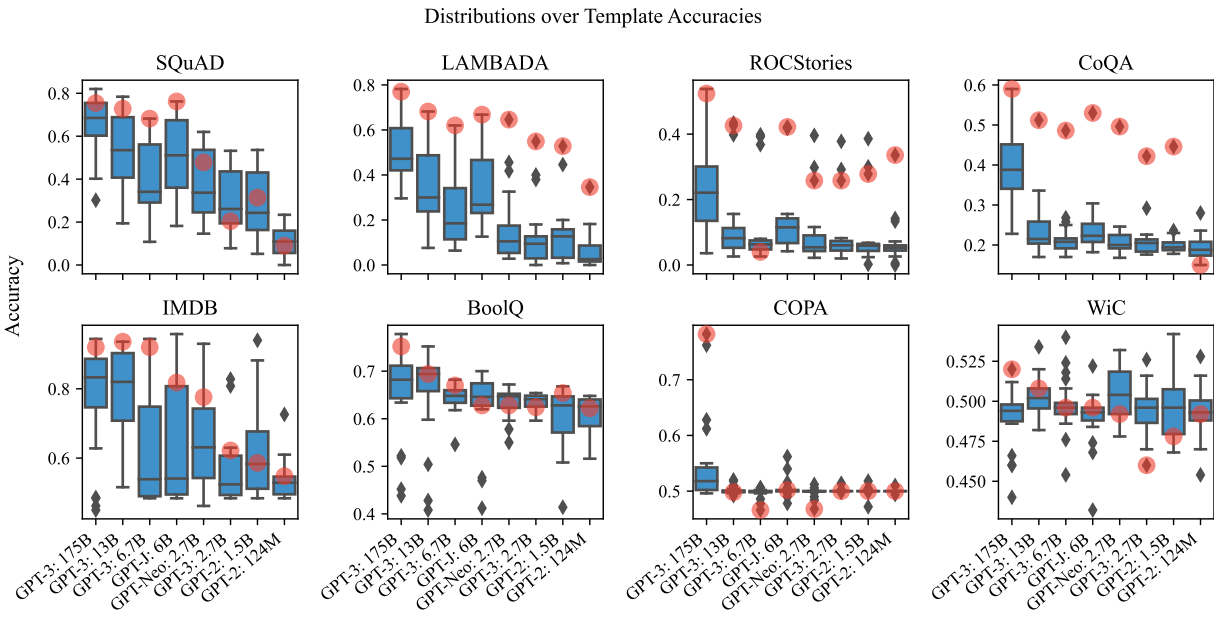

Figure 3: Distributions of accuracies over $K = 20$ templates for each model/dataset pair, compared to the prompts selected with MI (translucent red dots).

LAMBADA (Paperno et al., 2016), ROCStories (Mostafazadeh et al., 2016), CoQA (Talmor et al., 2018), IMDB (Maas et al., 2011), BoolQ (Clark et al., 2019), COPA (Gordon et al., 2012), and WiC (Pilehvar and Camacho-Collados, 2018))–that span seven unique NLP tasks (see Table 1). We used a random sample of $N = 500$ samples from each dataset for our experiments.[3] For ROCStories, which consists of a set of five sentence stories, we randomly masked a word from each story in order to use the data for masked word prediction (cloze).

We made minor changes to two of the datasets in order to cast the associated tasks into OTR. For the SQuAD dataset, we dropped all questions that did not have a one word answer, and for the CoQA dataset, we dropped all questions that had answer choices that started with a shared first word (e.g, the dog, the cat, the monkey). Both of these changes were to decrease ambiguity about which option the model was choosing given its output token distribution for a single token.

4.2 Models

We assess the performance of our method on eight models ranging in size from 124 million to 175 billion parameters. Specifically, we use two sizes of GPT-2 (Radford et al., 2019) (124M, 1.5B), the largest GPT-Neo (Black et al., 2021) model (2.7B), GPT-J (Wang and Komatsuzaki, 2021) (6B), and the four sizes of GPT-3 (Brown et al., 2020) (Ada, Babbage, Curie, and Davinci). We assume (as in (Perez et al., 2021)) that these named models are the four largest models in (Brown et al., 2020), with parameter counts 2.7B, 6.7B, 13B, and 175B respectively. Each model was trained in a generative manner to do next-token prediction.

[3]We sampled from the train sets of CoQA and SQuAD; the train and validation sets of WIC, COPA, and BoolQ; the full datasets of ROCStories and IMDB; and the test set for LAMBADA.

5 Results

In this section, we analyze our experiments. First, we look at our method's ability to select high-accuracy prompts across models and datasets (Section 5.1). Next, we correlate template mutual information and accuracy in Section 5.2. In Section 5.3 we explore the robustness of MI and use ensembling to improve it. Finally, we compare the tranferability of prompt templates selected with MI from model to model in Section 5.4.

5.1 Template Selection Performance

We first define baselines against which we compare our approach. Other prompt engineering methods generally require either access to model weights, labeled data (validation set selection), or both (backprop/continuous prompt embedding methods). Our method does not require these, so we instead compare to random and oracle baselines. A random template selection method would give us the average accuracy of our template set (in expectation),

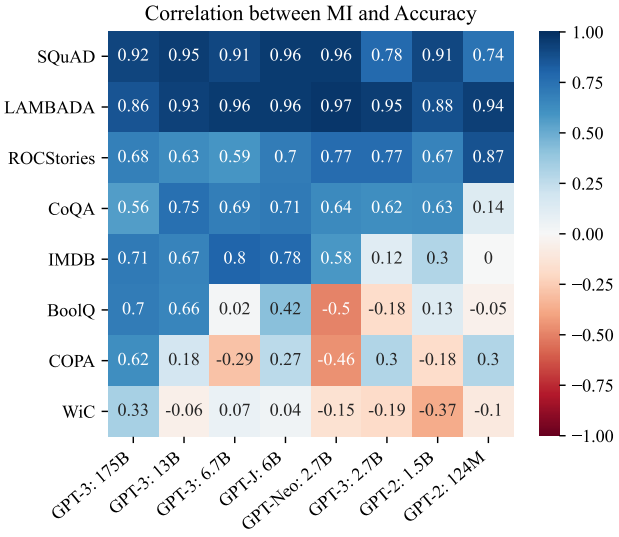

Figure 4: Correlations are more consistently high across all tasks for the largest models, suggesting that our method is most useful at those model sizes.

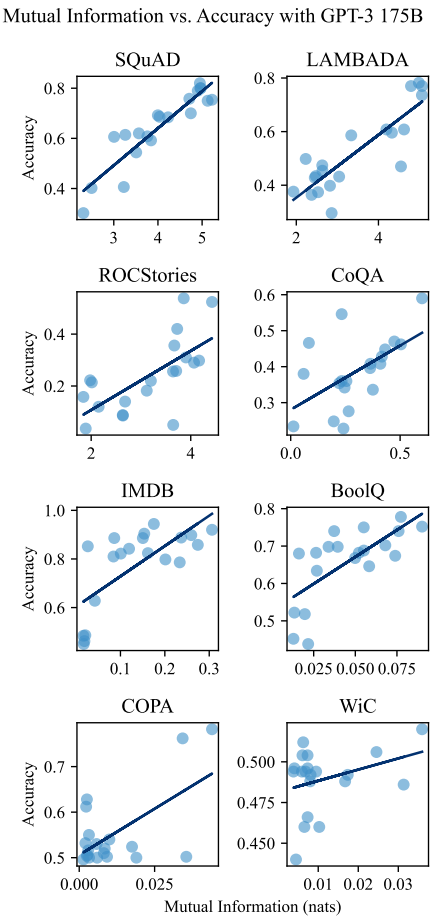

Figure 5: Each dot represents a template and its average mutual information and accuracy over $N = 500$ task instances. Linear best fit (by mean standard error) lines are included to show overall trends.

while an oracle selection method would give us the best accuracy every time. To understand how our MI method compares to these two baselines for each dataset, refer to Figure 1, where we analyze performance on GPT-3 175B. On each of the eight datasets, mutual information selects a prompt template that outperforms both the mean and median accuracies (random baseline performance). In three of the eight datasets, mutual information selects the best (highest accuracy) template from the 20 proposed (equivalent to oracle performance).

Given our method's promising performance with GPT-3 175B, it is natural to ask how it performs with smaller models. Figure 3 shows the accuracy distributions over prompt templates for each dataset/model pair. With every model, MI gives above-average performance on several datasets. Although MI is more likely to select a high accuracy template for larger models, it is a good criterion even for smaller models on all but two datasets, COPA and WiC. Note that, for these two datasets, none of the templates do significantly better than chance (∼50%) besides the largest model on COPA, which is in line with previous work.[4] Thus, we observe that mutual information performs best when there is a high-signal prompt to select from, and worse when all prompts are low-signal.

When considering all datasets but these, MI selects an above average prompt 83% of the time for all models; for the largest two models, MI selects an above average template 100% of the time (even including WiC and COPA).

5.2 Correlation between Template Mutual Information and Accuracy

In Section 5.1, we see how the MI selected template does in terms of accuracy compared to all other templates. We have not discussed, however, how generally mutual information and accuracy are correlated, except that the highest MI template tends to have anomalously high accuracy. Here, we establish that their correlation is high across all templates for the largest models. Each of the $K = 20$ templates has two corresponding measures: average accuracy and average mutual information. We can use these pairs to correlate MI and accuracy via Pearson's R.

We see in Figure 4 that the correlations are surprisingly high for the majority of models and

[4] Our template's best accuracy is 54% for WiC, and 78.2% for COPA, which is similar to previous work (WiC: (Brown et al., 2020) - 49.4%, (Perez et al., 2021) - 54.1%; COPA: (Brown et al., 2020) - 92.0%, (Perez et al., 2021) - 84.8%).

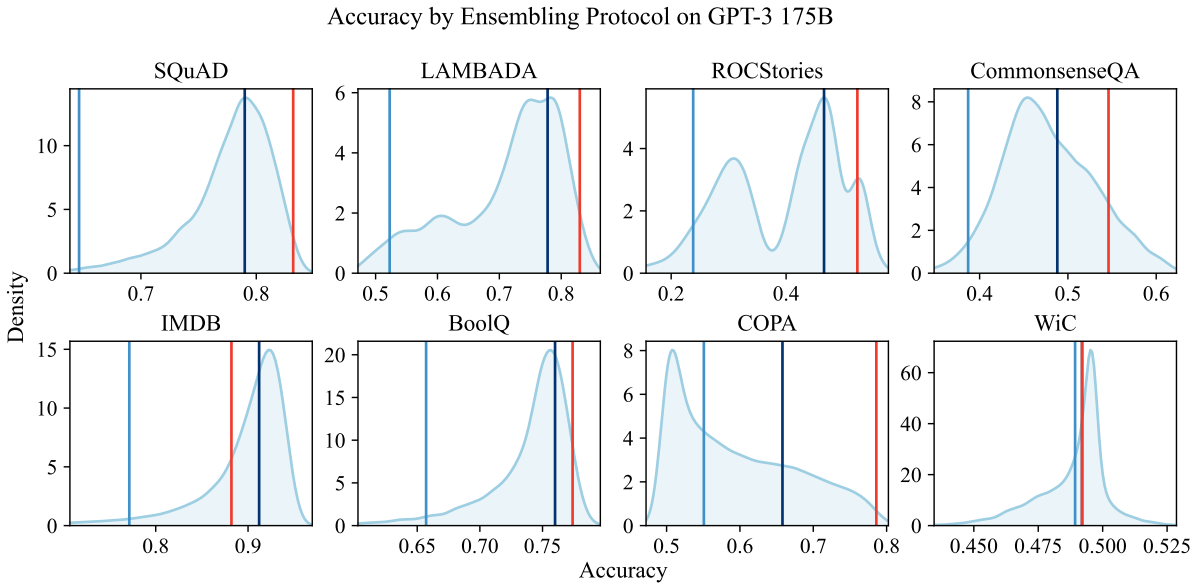

Figure 6: For each dataset the KDE plot represents accuracy over each of the $\binom{20}{5}$ ensembles of 5 templates from the 20 templates associated with the dataset. Each plot also includes lines representing the average accuracy of all single templates for the dataset, the accuracy of the ensemble of all 20 templates, and the accuracy of the ensemble of the top 5 templates chosen by MI. In only one case does all-20 beat top-5-MI, and it does so at 4× the cost.

datasets. For SQuAD, LAMBADA, ROCStories, and CoQA, this pattern holds across all model sizes; for the remainder, results are good on larger models and are much less reliable on smaller models. Overall, this is evidence that as mutual information increases, so does accuracy. In other words, mutual information can be used to make an educated guess about accuracy without having to use any ground truth labels, especially on larger models.

5.3 Method Robustness and Ensembling

We now explore the robustness of our method. To do this, we consider the question: what if we had included a different subset of templates, especially not including the top MI template? Figure 5 contains average mutual information/accuracy data for all $K = 20$ prompt templates on GPT-3 175B (similar plots for other models are found in Appendix C). For SQuAD, LAMBADA, ROCStories, CoQA, BoolQ, and IMDB, the results are robust; the top few prompt templates (by MI) are all high performers. For COPA and WiC, the performance is more brittle, and excluding the top-MI template would have resulted in a large drop in accuracy. This attests, first of all, to the utility of generating a diverse slate of templates as recommended in Appendix A, but also to the risk that outliers could compromise the effectiveness of the method.

A comprehensive discussion of remedies for outliers is beyond the scope of this paper, but it is an important concern. Considering the strength of MI/accuracy correlations, one simple approach is to ensemble the top 5 MI templates.

To compare this principled top-5 ensemble to other possible ensembles of templates, we do the following for each dataset: First, we take all $\binom{20}{5}$ subsets of 5 templates from all 20 templates; second, we calculate the accuracy of each ensemble, and plot this distribution's kernel density estimate, which models the p.d.e. of the random variable "accuracy of 5 random templates ensembled together"; lastly, we compare the accuracy of the top-5-MI templates with the accuracy of the ensemble of all 20 templates and the average accuracy of all templates (equivalent to the average accuracy of the 20 points in each scatterplot in Figure 5). The results are shown in Figure 6.

We would expect both the full and the MI ensembles to beat the average accuracy across templates. Surprisingly, we found that the top-5 mutual information ensemble does at least as well as the full ensemble in all but one case, IMDB, where the difference is just 0.03. Two reasons to use mutual information are, then, that 1) the MI ensemble gets as good or better a result as ensembling all prompt templates and 2) at a fourth of the experimental cost.

In short, ensembling by MI is a cheap and effec-

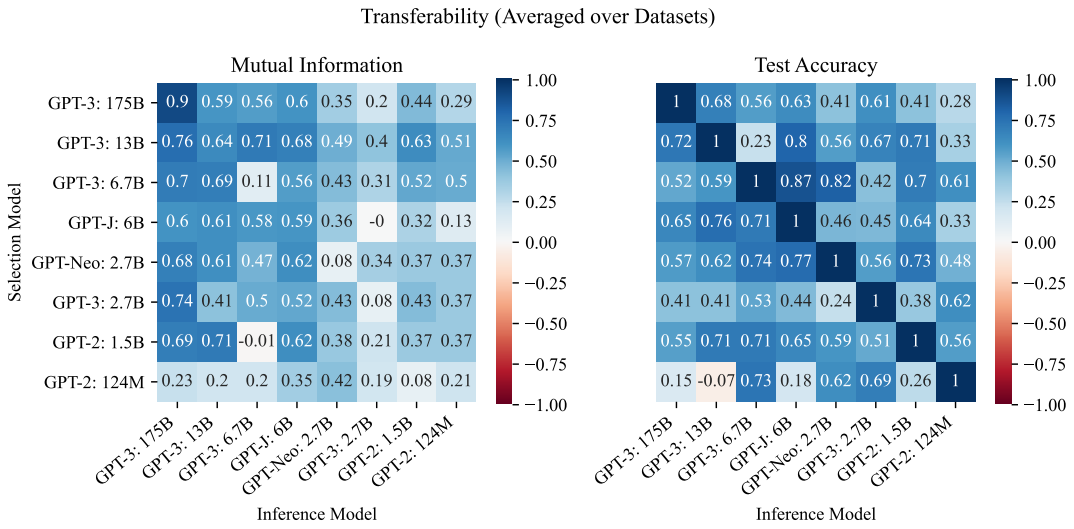

Figure 7: For each model/dataset pair, accuracies are normalized linearly so that 0 is the average prompt accuracy and 1 is the highest test accuracy. Using the prompt chosen by either MI or test accuracy on each selection model, average performance across datasets is reported for each inference model.

tive way to guard against anomalous high mutual information/low accuracy templates.

5.4 Transferability across Models

Finally, we explored how well-chosen templates generalize between models. There are several reasons for doing this. First, model transfer can be useful if more powerful models can only be used selectively (because of cost or access), either for prompt template selection or for inference, and other models must be used in the rest of the workflow. Additionally, studying model transfer can shed light on the universality of template quality across models of different sizes and training regimes.

Concretely, we choose templates by maximizing either test accuracy (oracle) or mutual information (our method) using a selection model ϕ_s, and then calculate test accuracy using a different inference model ϕ_i. We calculate absolute test accuracy and then normalize it such that 0 and 100 correspond to the average and maximum scores across templates for a model/dataset pair. We average our results across datasets and present the results in Figure 7.

MI performance is best when the largest model (GPT-3 175B) is used as both the selection and inference model: on average, MI scores 90% on this normalized scale. Additionally, performance is most consistently high when the largest models are used either for selection or inference. But the vast majority of transfer scores are well above 0 (only one negative average gain out of 64 transfer permutations), suggesting that transfer is often reasonable, and that similar templates work across models given a task.

Overall, we have observed that prompt selection by mutual information is surprisingly effective across a variety of datasets and model sizes. This method works best on larger models and for tasks that the LM is capable of performing. Given the high diversity of tasks that we have explored, we expect this method to transfer well to many other NLP tasks, including ones where there are few or no ground truth labels.

6 Conclusion

In this paper, we introduce a method for selecting prompts that effectively align language models to NLP tasks. Over a set of candidate prompts, our method selects the template that maximizes the mutual information between the input and the model output. We demonstrate that 1) mutual information is highly correlated with test accuracy and 2) selecting a prompt based on mutual information leads to significant accuracy gains over random choice, approaching oracle performance on GPT-3 175B, and it does so across model sizes and tasks.

Whereas other methods rely on ground truth labels and/or direct model access, ours requires neither. Many applications characterized by lack of computational resources, limited model access (e.g., inference only), and lack of ground truth data prohibiting testing of candidate prompts become feasible with our method.

References

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

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

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.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-based *n*-gram models of natural language. *Computational Linguistics*, 18(4):467–480.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *arXiv*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *CoRR*, abs/1905.10044.

Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory 2nd Edition (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience.

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

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 394–398, Montréal, Canada. Association for Computational Linguistics.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. PPT: Pre-trained Prompt Tuning for Few-shot Learning.

Ziwei Ji, Justin D. Li, and Matus Telgarsky. 2021. Early-stopped neural networks are consistent.

Lingpeng Kong, Cyprien de Masson d'Autume, Wang Ling, Lei Yu, Zihang Dai, and Dani Yogatama. 2019. A mutual information maximization perspective of language representation learning.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. pages 4582–4597.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. 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.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. pages 1–46.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. GPT Understands, Too.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James F. Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *CoRR*, abs/1604.01696.

Preetum Nakkiran and Yamini Bansal. 2020. Distributional generalization: A new kind of generalization. *CoRR*, abs/2009.08092.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and

Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. *CoRR*, abs/1606.06031.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True Few-Shot Learning with Language Models. (Cv):1–21.

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.

Mohammad Taher Pilehvar and José Camacho-Collados. 2018. Wic: 10, 000 example pairs for evaluating context-sensitive representations. *CoRR*, abs/1808.09121.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. pages 1–53.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm.

Karl Stratos. 2019. Mutual information maximization for simple and accurate part-of-speech induction.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *CoRR*, abs/1811.00937.

Elena Voita and Ivan Titov. 2020. Information-Theoretic Probing with Minimum Description Length. pages 183–196.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable Prompt Makes Pre-trained Language Models Better Few-shot Learners. pages 1–18.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate Before Use: Improving Few-Shot Performance of Language Models.

Yukun Zuo, Quan Fang, Shengsheng Qian, Xiaorui Zhang, and Changsheng Xu. 2018. Representation Learning of Knowledge Graphs with Entity Attributes and Multimedia Descriptions. *2018 IEEE 4th International Conference on Multimedia Big Data, BigMM 2018*, pages 2659–2665.

A Prompt Engineering Process

In this section, we will step through our method in detail. Again, note that this method uses no ground truth labels and does not require gradient updates or access to the model parameters. Given a task that can be represented in natural language with the OTR framework, the only requirements for our approach are a) several candidate prompt templates and b) some instances (X) on which to do inference.

1. **Generate a set of K prompt templatizing functions with corresponding collapsing functions.** Each prompt template function f_{θ_k} should take in an input from the dataset and output a prompt ready for processing by the language model. Each template must also have a collapsing function c_{θ_k} that takes the language model output and produces a distribution over targets. Prompt template functions should be chosen to be as diverse as possible to increase the probability of finding a range of low- to high-quality prompts. For example, we use templates that frame input from datasets as test questions, back and forth dialogue between friends, Python code, test answer banks, etc. A sample of the prompt templates used in this work is provided in Appendix B. A good resource for coming up with prompt template function ideas is the OpenAI API examples collection[5].

2. **Playground.** For each chosen f_{θ_k}, calculate $g_\phi(f_{\theta_k}(x))$ for a few dataset samples. Do not look at associated ground truth labels for these samples. Simply check to ensure that g_ϕ puts high probability on the tokens one would expect given f_{θ_k} that could be reasonably collapsed by c_{θ_k} into $P(Y)$. For example, on the BoolQ reading comprehension task, the language model predicts the answer to a yes/no question related to a corresponding passage. Given this task, we would expect the highest probability to be on tokens like "Yes" or "No". A poor prompt template, on the other hand, might put the highest probability on unrelated tokens like "I", "think", or "\n". Revise or replace any template that fails to put high probability mass on the tokens expected.

3. **Estimate mutual information for each template f_{θ_k}.** Choose how many data points N to use for estimating mutual information for each template function. A higher N will allow for estimation of mutual information based on a more representative sample of the dataset at the cost of more language model computation. Sample N samples from your dataset. Since we do not require any Y labels, one could even choose the X's on which you desire to do inference (as we do). Then, for each sample x and each template f_{θ_k}, calculate $P(Y|f_\theta(x))$ using Equation 1. Use the output to estimate mutual information for each prompt template with Equation 2.

For all of our experiments, c_θ takes in a distribution of tokens $g_\phi(f_{\theta_k}(x))$ and a mapping between the set of possible ground truth labels for $f_{\theta_k}(x)$ and model vocabulary T_ϕ. For a sentiment analysis task, that mapping would be from the ground truth labels "positive" and "negative" to the expected tokens "positive" and "negative" respectively. If a given prompt template for sentiment analysis was phrased as a yes/no question, the mapping for that prompt template would be from "positive" and "negative" to "yes" and "no" respectively. Our c function returns a probability over Y (target label space), and the highest probability label is treated as the prediction. To keep things simple, the values in our map are always single tokens. See examples in Appendix B.

4. **Choose prompt template(s) to use for inference based on mutual information.** For choosing a single prompt template to use for inference, select the template with highest estimated mutual information. With an increased computational budget, one could also ensemble the top p prompt templates, as we describe in Section 5.3.

5. **Use chosen prompt template(s) to perform inference** Use chosen prompt template(s) $f_{\hat{\theta}}$ to calculate $c_{\hat{\theta}}(g_\phi(f_{\hat{\theta}}(x))$ for each dataset sample. Inference can be done with the language model used for estimating mutual information or a smaller model if cost is prohibitive (for information on performance statistics with this approach, see Figure 7).

B Template Examples

The following are example template f_θs provided for each dataset. We include the highest accuracy template, but all templates used can be found at [GitHub URL removed for anonymity. Our code is included in a ZIP file in our submission]. In blue, we highlight the data that is filled in from X; in red, we highlight the area where we ask the

[5] beta.openai.com/examples

model to predict the next token; everything that is not highlighted is static from instance to instance.

We also include the token sets used in the collapsing functions, if applicable.

B.1 SQuAD

Mutual Information: 4.950, Accuracy: 0.820

TASK: Answer the questions below using the phrasing from the context.

CONTEXT: As of the census of 2000, there were 197,790 people, 84,549 households, and 43,627 families residing in the city. The population density was 3,292.6 people per square mile (1,271.3/km). There were 92,282 housing units at an average density of 1,536.2 per square mile (593.1/km). The racial makeup of the city was 38.3% White, 57.2% African American, 0.2% Native American, 1.3% Asian, 0.1% Pacific Islander, 1.5% from other races, and 1.5% from two or more races. Hispanic or Latino of any race were 2.6% of the population.

QUESTIONS:
1) In 2000, how many families lived in Richmond?
Answer: "43,627"

2) What percentage of the Richmond population of 2000 was Pacific Islander?
Answer: "

Collapsing token sets: None, all tokens are considered.

B.2 LAMBADA

Mutual Information: 4.984, Accuracy: 0.782:

Fill in blank:
Alice was friends with Bob. Alice went to visit her friend ____. -> Bob
"I would speak to you privately," Bowen said, casting a glance around at the others milling about.

The worry in her eyes deepened, but she nodded hesitantly and awaited Bowen's directive.

He led her through the great hall, annoyance biting at him when he saw no place where people weren't congregated. He stepped outside the back of the keep, where, finally, he spied an area near the bathhouses, where it was quiet and ____. ->

Collapsing token sets: None, all tokens are considered.

B.3 ROCStories

Mutual Information: 3.859, Accuracy: 0.538

Fill in the blank for the following sentences.
"Marissa loved _____ pokemon go game. It is the biggest thing right now. She had done so much more walking since she started playing it. She walked all day and evening sometimes. She walked almost 10 miles in two days." -> "Marissa loved

Collapsing token sets: None, all tokens are considered.

B.4 CoQA

Mutual Information: 0.600, Accuracy: 0.590

Instructions: For each question below, choose the answer from the answer bank corresponding to the question that best answers the question.

Question 1 Answer Bank: ladybug, bunny, goldfish, leopard, caterpillar

Question: What animal would be most dangerous for a human to encounter in the wild?

Answer: leopard

Question 2 Answer Bank: wrong, pleasure, encouragement, depression, relief

Question: If you're still in love and end up stopping being married to your partner, what emotion are you likely to experience?
Answer:

Collapsing token sets: {A: 'wrong', B: 'pleasure', C: 'encouragement', D: 'depression', E: 'relief'}

B.5 IMDB

Mutual Information: 0.175, Accuracy: 0.944):

P1: How was the movie?
P2: John Cassavetes is on the run from the law. He is at the bottom of the heap. He sees Sidney Poitier as his equal and they quickly become friends, forming a sort of alliance against a bully of a foreman played by Jack Warden.
As someone who has worked in a warehouse myself when I was younger, I can tell you that the warehouse fights, complete with tumbling packing cases and flailing grappling hooks are as realistic as it gets. I've been in fights like these myself, although no one got killed.

The introduction of Sidney Poitier's widow is a variation on Shakespeare's Shylock "Do I not bleed?" This is an anti racist film, which, at the time, was much needed.
All the three principle characters - Warden, Cassavetes and Poitier - are superb, with Warden the most outstanding of the three.
P1: Would you say your review of the movie is negative or positive?
P2: I would say my review review of the movie is

Collapsing token sets: {Positive: 'positive', Negative: 'negative'}

B.6 BoolQ

Mutual Information: 0.077, Accuracy: 0.778

Given the passage and question, please answer the question with yes or no.
"'Turn on red – In Canada, left turn on red light from a one-way road into a one-way road is permitted except in some areas of Quebec, New Brunswick, and Prince Edward Island. Left turn on red light from a two-way road into a one-way road is permitted in British Columbia but only if the driver turns onto the closest lane and yields to pedestrians and cross traffic.'", "'Can you turn left on red in canada?'" -> "'Yes'"
"'Pyruvic acid – Pyruvic acid (CHCOCOOH) is the simplest of the alpha-keto acids, with a carboxylic acid and a ketone functional group. Pyruvate (/paruvet/), the conjugate base, CHCOCOO, is a key intermediate in several metabolic pathways.'", "'Is pyruvic acid and pyruvate the same thing?'" -> "'

Collapsing token sets: {True: 'Yes', False: 'No}

B.7 COPA

Mutual Information: 0.044, Accuracy: 0.782

For the following premises, choose the alternative that is either a cause or result of the premise, and justify your answer.
Premise: The man broke his toe. What was the CAUSE of this?
Alternative 1: He got a hole in his sock.
Alternative 2: He dropped a hammer on his foot.
Answer: Alternative 2. Getting a hole in your sock would not break your toe, unless there is additional information. Dropping a hammer (which is a heavy object), on the other hand, would almost certainly break your toe. Thus, the best answer is Alternative 2.

Premise: I tipped the bottle. What happened as a RESULT?
Alternative 1: The liquid in the bottle froze.
Alternative 2: The liquid in the bottle poured out.
Answer: Alternative 2. Tipping a bottle causes liquid to fall out, not to freeze. Freezing is caused by being placed in a cold place. Pouring out (Alternative 2) is correct because it makes the most sense.

Premise: I knocked on my neighbor's door. What happened as a RESULT?
Alternative 1: My neighbor invited me in.
Alternative 2: My neighbor left his house.
Answer: Alternative 1. When you knock on a neighbor's door, it is likely that if they are home they will answer and invite you in. It does not make much sense, however, that a neighbor would leave their house without explanation. Therefore, Alternative 1 is the best result of the premise.

Premise: My foot went numb. What happened as a RESULT?
Alternative 1: I put my shoes on.
Alternative 2: I shook my foot.
Answer: Alternative

Collapsing token sets: {Alternative 1: '1', Alternative 2: '2'}

B.8 WiC

Mutual Information: 0.036, Accuracy: 0.520

Classify whether the following two sentences' use of the word has the same meaning or not.

Word: bright
Usage 1: He is a bright child
Usage 2: The sun is very bright today
Meaning: different

Word: didacticism
Usage 1: The didacticism of the 19th century gave birth to many great museums.
Usage 2: The didacticism expected in books for the young.
Meaning:

Collapsing token sets: {Same: 'same', Different: 'different'}

C Additional Figures

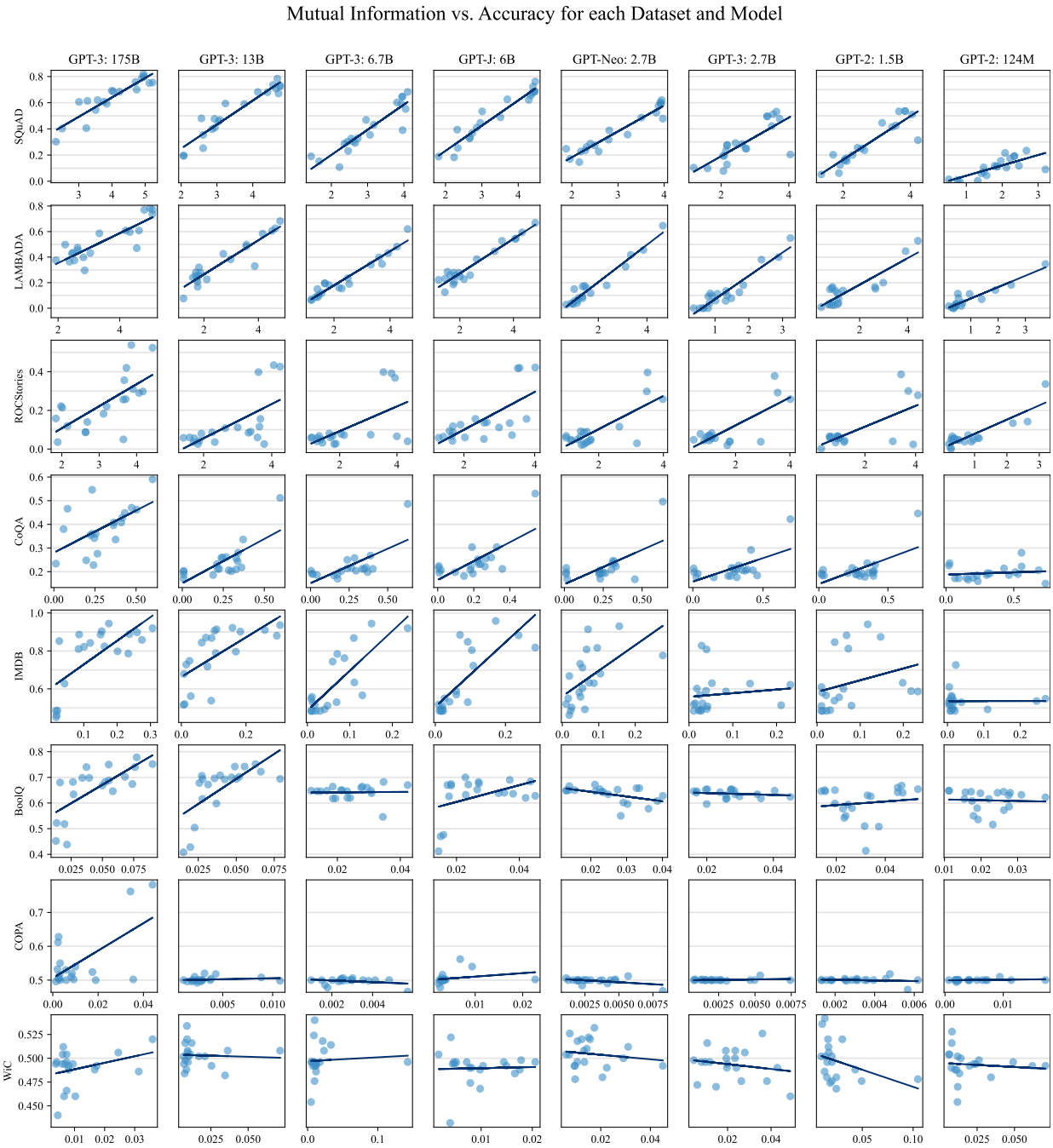

Figure 8: Mutual information plotted against accuracy per prompt for each dataset using GPT-3 175B with linear best fit (by MSE) lines to show overall trends