LMPriors: Pre-Trained Language Models as Task-Specific Priors

Anonymous Author(s)
Affiliation
Address
email

Abstract

Particularly in low-data regimes, an outstanding challenge in machine learning is developing principled techniques for augmenting our models with suitable priors. This is to encourage them to learn in ways that are compatible with our understanding of the world. But in contrast to generic priors such as shrinkage or sparsity, we draw inspiration from the recent successes of large-scale language models (LMs) to construct *task-specific priors* distilled from the rich knowledge of LMs. Our method, Language Model Priors (LMPriors), incorporates auxiliary natural language metadata about the task—such as variable names and descriptions—to encourage downstream model outputs to be consistent with the LM's common-sense reasoning based on the metadata. Empirically, we demonstrate that LMPriors improve model performance in settings where such natural language descriptions are available, and perform well on several tasks that benefit from such prior knowledge, such as feature selection, causal inference, and safe reinforcement learning.

1 Introduction

Much of modern-day machine learning is *data-driven*—given training examples, we aim to learn a function that minimizes an objective corresponding to a particular downstream task. This paradigm has led to tremendous success in data-rich domains such as protein structure prediction for drug discovery [1], game playing [2, 3, 4], automating medical diagnoses [5], computational sustainability [6, 7], and climate modeling [8, 9]. However, the recent failures of such algorithms as in shortcut learning and vulnerability to adversarial examples [10, 11, 12, 13, 14] seem to suggest that purely data-driven approaches have a long way to go from becoming truly *intelligent* agents.

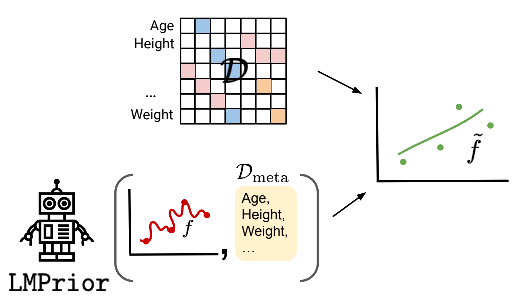


Figure 1: A flowchart of the Language Model Prior (LMPrior) framework. We leverage the rich knowledge base of a pretrained LM to incorporate task-relevant prior knowledge into our learning algorithm f. Our method uses natural language metadata $\mathcal{D}_{\text{meta}}$ to return a specialized learner $\tilde{f}$, whose outputs given the dataset $\mathcal{D}$ are encouraged to remain consistent with both the metadata and real-world knowledge as distilled in the LM.

One facet of intelligence which separates human agents from artificial ones is *prior knowledge about the world* that can be combined with inferences derived purely from data. Consider a prediction setting that aims to determine the length of one's commute time. Although an algorithm may discover a relationship between commute time and favorite color, our intuition tells us that this relationship is most likely spurious. Additionally, an autonomous driving agent may require several expert demonstrations before it learns that it should not veer off a cliff; a generative model may need to see an extremely large number of faces before it learns that earrings should not be placed on someone's head. These failure modes are surprising to us precisely because they violate deeply-ingrained prior beliefs about how the world works. Artificial agents, on the other hand, lack such grounding in real-world contexts and are thus limited in their ability to reason about the semantic relationships between entities present in data. This problem becomes even more pertinent in low-data regimes where our algorithms are prone to overfitting.

Two key observations guide this work. The first is that auxiliary metadata, often in the form of natural language descriptions such as variable names that ground the features in real-world entities, are becoming increasingly more abundant [15]. The second is that in spite of this, most conventional learning algorithms are designed to *ignore* this valuable information. This approach is understandable due to the subjective and qualitative nature of prior information elicited from experts or algorithm designers, combined with the difficulty of scaling up approaches to thousands or millions of variables. Inspired by the recent successes of large-scale pretrained language models (LMs) across a wide range of domains and data modalities [16, 17, 18, 19, 20], we propose to leverage the LM's rich knowledge base as a heuristic for prior knowledge about the world. This provides a pathway for algorithmically, scalably, and repeatedly generating relevant inductive biases from task-specific metadata such as variable names and descriptions. Our framework, which we call Language Model Priors (LMPriors), then serves as a way to construct *task-specific priors* tailored to any learning setting where natural language descriptions of the task are available. We provide an illustrative flowchart of how LMPriors fit into the conventional machine learning pipeline in Figure 1.

Empirically, we demonstrate that our LMPriors are able to perform well on a variety of downstream tasks which benefit from auxiliary sources of information. Concretely, the contributions of our work can be summarized as follows:

1. We introduce LMPriors, a framework for algorithmically incorporating semantically-relevant prior knowledge into learning problems via use of a prior distribution extracted from a LM.
2. We explicitly specify LMPriors for feature selection, causal discovery, and reinforcement learning tasks. Each LMPrior is a mapping from a set of task-specific metadata $\mathcal{D}_{\text{meta}}$ to a learning procedure with a bespoke inductive bias.
3. We show empirically that LMPriors achieve strong performance on feature selection, causal discovery, and safe reinforcement learning tasks, and demonstrate that it can also serve as a useful preprocessing wrapper around existing algorithms to boost their performance.

2 Preliminaries

2.1 Neural Language Modeling

Language modeling seeks to learn a probability distribution $p_{\text{LM}}(\mathbf{x})$ over variable-length sequences of text $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_{|\mathbf{x}|})$, drawn from an underlying distribution $p_{\text{text}}(\mathbf{x})$, such that $p_{\text{LM}}(\mathbf{x}) \approx p_{\text{text}}(\mathbf{x})$. Although several approaches exist for parameterizing $p_{\text{LM}}(\mathbf{x})$, conventional neural LMs posit an autoregressive factorization over $p_{\text{LM}}(\mathbf{x}) = \prod_{i=1}^{|\mathbf{x}|} p_{\text{LM}}(\mathbf{x}_i|\mathbf{x}_{<i})$ and are trained via maximum likelihood [21, 22]. When predicting the next token $\mathbf{x}_i$, the preceding tokens $\mathbf{x}_{<i}$ are known as the *context* or *prompt* $\mathbf{c}$.

Modern LMs are trained on large corpora consisting of billions of tokens over diverse sources of text including encyclopedias, news websites, emails, books, and scientific papers [23]. In order to successfully predict the next token over such a diverse set of contexts, LMs implicitly possess rich knowledge about concepts in the training data. This allows them to solve a startling variety of tasks from simple descriptions of the task itself, a setting known as zero-shot learning [17]. We seek to leverage this rich knowledge base as the foundation of our approach.

Prompt design. Since the largest LMs are currently proprietary[1], we assume black-box access to the underlying LM and avoid cases where our method would need to fine-tune or access internal statistics (such as gradients or embeddings) of the model. Given this assumption, our control over the model's

[1]We note that this status quo is quickly changing with open-source tools such as HuggingFace [24].

predictions relies entirely on our choice of prompt. Effective prompt design is a key challenge when utilizing modern LMs, and one that has been widely studied [25, 26, 27, 28, 29].

2.2 Task-Specific Knowledge in Data-Driven Learning

To motivate our framework, we first consider a generic parameter estimation problem. Given a dataset $\mathcal{D}$ consisting of n data points $\mathbf{x}_i \in \mathcal{X}$ drawn from an underlying distribution $p(\cdot|\theta)$, our goal is to estimate $\theta \in \Theta$. We define a *learning procedure*, or *learner*, as a function $f : \mathcal{X}^n \to \Theta$ to do so. For instance, in a linear regression task where the dataset $\mathcal{D}$ consists of $(\mathbf{x}, \mathrm{y})$ pairs, the learner f may return the solution of a least-squares fit between the $\mathbf{x}$ and $\mathbf{y}$ samples in the dataset. For a probabilistic independence testing problem, where we again have a dataset $\mathcal{D}$ consisting of $(\mathbf{x}, \mathrm{y})$ pairs, the learner f would return a probability of independence between the two variables: $p(\mathbf{x} \perp\!\!\!\perp \mathbf{y})$. In the common empirical risk minimization (ERM) setting, we use a learning procedure with an $f(\mathcal{D}) = \arg\min_{\theta'} \sum_i^n \ell(\theta', \mathbf{x}_i)$ for some loss ℓ.We may even view reinforcement learning (RL) as a sequential instantiation of this problem, where we sequentially observe samples from a Markov Decision Process (MDP) and must estimate the optimal policy—a function of the MDP's parameters.

Challenges in learning. However, several challenges arise when designing an effective learning procedure f. The most common is inaccurate estimation of θ in a low-data setting. In fact, given finite samples without access to the underlying data generating process, we cannot guarantee that our estimate $\hat{\theta}$ will equal the true θ. While procedures such as ERM do guarantee that we will recover the true θ in the infinite data regime (under some regularity conditions) [30], in general there are no meaningful bounds on the number of samples needed for this convergence with modern deep learning architectures. Therefore, we must resort to approximate algorithms with few guarantees. A variety of "no-free-lunch" theorems [31] tell us that when averaged over all possible data generating processes, all predictive algorithms perform equally well. An approach that performs better on some particular distribution of data must make up for it by performing worse on another. Thus to find an effective learning procedure for a particular dataset, we must incorporate some assumptions about the data generating distribution.

Incorporating task-relevant metadata. A key observation is that the above loss-minimization framework actually *discards* task-relevant information. Concretely, f is agnostic to any contextual metadata that may give more information about the dataset $\mathcal{D}$. For example, in a regression setting the variable names and textual descriptions of $\mathbf{x}, y$ are not used—f operates directly on their values. However, such variable names can provide valuable information which we can exploit in our design of f. For example, if we know that the output of a prediction task represents age, we can construct f such that the predictor it produces is always constrained to be non-negative. Similarly if we know that our task is to predict a magnetic field, we may design f so its output is a vector field with zero divergence. In this way the variable names can be used to introduce *task-relevant bias* into f by incorporating auxiliary information that is not present in the dataset $\mathcal{D}$. This should help generalization, as it encourages the learning algorithm to recover f that is consistent with the information we have from the context and *grounds* the learning task in real-world entities. This becomes particularly important in low-data regimes, where f is prone to overfitting [32].

Machine learning practitioners today already incorporate such auxiliary information—they explicitly set prior distributions, choose models known to perform well on similar datasets, and drop a-priori irrelevant features from consideration. We can view this procedure as abstractly utilizing some additional *metadata* $\mathcal{D}_{\text{meta}}$ which consists of variable names, data collection details, and other contextual information not contained in the dataset itself to develop a task-relevant bias to give f. Abstractly, the action of the practitioner $\mathcal{P}_{\text{expert}}$ may be represented as the following functional transformation:

$$\mathcal{P}_{\text{expert}}(\mathcal{D}_{\text{meta}})(f) = \tilde{f}$$

where $\tilde{f}$ is a new learning procedure with a useful task-specific bias. Such metadata is becoming increasingly available, standardized, and descriptive [15]. Given this abundance of metadata, our goal is to develop a procedure which can assist practitioners by automatically constructing a task-relevant bias which can incorporated into a learning procedure f.

3 The LMPrior Framework

From the above observation, we consider how to combine task-relevant natural-language metadata $\mathcal{D}_{\text{meta}}$ into our algorithm f. To do so, we introduce Language Model Priors (LMPriors), a framework for leveraging a pretrained LM as the method to algorithmically interpret $\mathcal{D}_{\text{meta}}$. We emphasize that LMPriors can only handle situations where textual information about $\mathbf{x}$ and y (such as descriptions) are available; without them, we must return to the standard learning setting.

LMPrior as a function transform. We define LMPriors as a family of functions $\mathcal{P}$ which take some relevant metadata $\mathcal{D}_{\text{meta}}$ which is not used by the traditional learning process f. The LMPrior then transforms f to $\tilde{f}$ which exhibits a bias towards outputs which are consistent with the metadata $\mathcal{D}_{\text{meta}}$. In the following section we describe several specific instantiations of LMPriors, describing in each case how the metadata is used to elicit a common-sense judgment which is then incorporated into the learning procedure $\tilde{f}$.

3.1 Task Overview

Feature selection. In a feature selection task, where the goal is to select a subset of the dataset's most informative features while discarding irrelevant ones, the LMPrior acts as a *regularizer*. We assume that the metadata $\mathcal{D}_{\text{meta}}$ consists of all variable names, descriptions of all variables, and a short sentence of context. The goal is to elicit the prior probability that a variable x is predictive of the target y given the variable names and context; we describe the explicit prompt used as a function of the metadata in Figure 3.2. For example, in a setting where our data source has been corrupted by an auxiliary dataset, we would like to filter out those nuisance variables that would hurt f's performance on the original data $\mathcal{D}$. We use the LM to generate the probability that variables are relevant, and remove them from the dataset if the probability is less than a specified threshold τ. This acts as a form of regularization on the subset of features selected for a downstream prediction task.

Reinforcement learning. In reinforcement learning (RL) we face a more general learning task. The input $\mathcal{D}$ is a Markov Decision Process (MDP) consisting of a tuple $(\mathcal{S}, \mathcal{A}, p_0, q, r, \gamma)$, where $\mathcal{S}, \mathcal{A}$ are state and action spaces, p_0 and q are the initial state distribution and dynamics, $r(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and γ is the discount factor. The goal is to find a policy $f : \mathcal{S} \to \mathcal{A}$ which maximizes the expected distribution of rewards under the dynamics. Similarly to how practictioners add in inductive biases to the desired behaviour via reward shaping, the role of the reinforcement learning LMPrior $\mathcal{P}_{\text{RL}}$ is to modify the MDP via reward shaping. We assume that the metadata consists of a mapping from the raw state and action variables to a natural language description, such as a method to convert a set of pixels to a textual description. The metadata also consists of a set of examples of hypothetical (state, action) pairs and judgments of their value. The goal is to elicit a shaped reward including a bonus that should be given to the agent for the current state and action. For example, the common-sense reward awarded should be negative for a self-driving car crashing, or positive for a puzzle-solving agent collecting a key. Note that if we are specifically concerned with possible suboptimality in the original MDP after training with reward shaping, we may use potential-based reward shaping [33], where optimality with respect to $\tilde{r}$ guarantees optimality with respect to r.

Concretely, we combine the metadata into a prompt forcing the LM to classify the state,action pair as `good` or `bad`. We then obtain a new reward function $\tilde{r}(s,a) = r(s,a) + \mathbb{E}_{t \sim p_{\text{LM}}(\cdot|\mathbf{c}(s,a,\mathcal{D}_{\text{meta}}))}\left[\mathbb{1}_{\text{good}}[t] - \mathbb{1}_{\text{bad}}[t]\right]$, where $\mathbf{c}(s,a)$ is the current (state,action)-dependant prompt and $\mathbb{1}_{\text{good}}$, $\mathbb{1}_{\text{bad}}$ are the indicator functions over the output tokens `good` and `bad` respectively. In this work, we study the application of an RL LMPrior to the problem of safe RL: leveraging pre-existing knowledge about the desirability of entering hazardous areas to reduce violations of safety constraints.

Causal discovery. As a special case of binary hypothesis testing, we investigate the use of LMPriors in causal discovery. Here our goal is to elicit the relative prior probability of the possible relationships between two variables $\mathbf{x}$ and $\mathbf{y}$: $\mathbf{x} \to \mathbf{y}$ or $\mathbf{y} \to \mathbf{x}$. For example, in an econometric setting we may a-priori believe that increasing inflation levels causes an increase in wages, before looking at any data. Many recent works have been developed to infer the causal direction from observational data [34, 35, 36]. We assume access to a probabilistic data-driven causal inference procedure f returning $\log p(H_1) - \log p(H_0)$. Here H_0 is the hypothesis that the causal direction is $\mathbf{x} \to \mathbf{y}$ and H_1 the hypothesis that the causal direction is $\mathbf{y} \to \mathbf{x}$. The causal discovery LMPrior $\mathcal{P}_{\text{CD}}$ requires metadata consisting of names and descriptions of $\mathbf{x}$ and $\mathbf{y}$, as well as a sentence of brief context. These are then included in a prompt $\mathbf{c}(\mathcal{D}_{\text{meta}})$ (described explicitly in figure 4.3) designed to elicit either the sentence $\texttt{X} \rightarrow \texttt{Y}$ or $\texttt{Y} \rightarrow \texttt{X}$. The LMPrior then augments f by adding on the prior likelihood:

$$\mathcal{P}_{\text{CD}}(f)(\mathcal{D}) = \log\left(\frac{p_{\text{LM}}(\mathbf{x} \rightarrow \mathbf{y}|\mathbf{c}(\mathcal{D}_{\text{meta}}))}{p_{\text{LM}}(\mathbf{y} \rightarrow \mathbf{x}|\mathbf{c}(\mathcal{D}_{\text{meta}}))}\right) + f(\mathcal{D})$$

In this setting, $\mathcal{P}_{\text{CD}}(f)$ returns the (log) posterior for the most likely causal structure for $\mathbf{x}$ and y.

3.2 Model Architecture and API

Model Details. We use the `Davinci` GPT-3 model for the LM backbone for LMPrior, as it has the largest number of parameters available (175B) and achieves strong performance on a number of benchmarks [17]. We use the `davinci-instruct-beta` variant, and access GPT-3 via the OpenAI API.

Prompt Format. Although we adapt the prompt for each of our downstream tasks, we largely keep its overall format consistent following the best practices in [28]. Specifically, we utilize a template consisting of: (1) a natural language description of the task which contextualizes the following examples in the prompt; (2) a small number of examples instructing GPT-3 with the desired behavior; and (3) an explanation intended to guide GPT-3 with some intuition for the correct answers. The inclusion of the explanation ensures that the context has examples of thoughtful reasoning. It can also serve as a useful tool to understand erroneous predictions, as it indicates some amount of reasoning behind the prediction. We illustrate our prompts in Figures 4.3 and 3.2. We note that we tailor the particular description as well as the provided examples to the task of interest. We outline some more detailed guidelines and empirical findings from formatting the various prompt formats in Appendix A.

Decision Rule. Given the LMPrior's completion to a particular prompt, we can leverage its response as either a "soft" or "hard" decision rule. Concretely, in the feature selection setting, a particular threshold value τ determines the cutoff as to whether certain features will be included in the downstream predictor. For the causal inference task, we utilize the LMPrior's outputs as soft probabilities and combine them with a data-driven likelihood method approach to obtain a posterior belief over the most plausible structure.

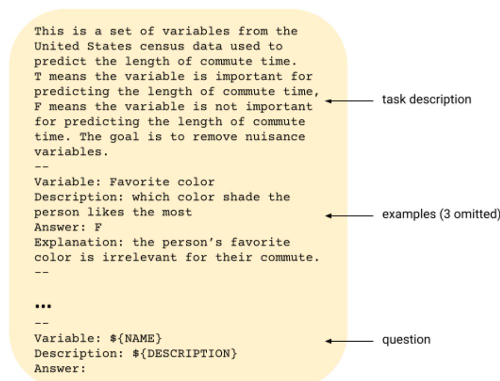


Figure 2: An example of a prompt used in LMPriors for the feature selection task in Section 4.1.2. The prompt $\mathbf{c}$ consists of a textual description of the feature selection task, the variable name, a short description of the variable, and the correct answer followed by an explanation. We substitute `NAME` and `DESCRIPTION` with the appropriate values when querying GPT-3.

4 Empirical Evaluations

In this section, we are interested in empirically answering the following questions:

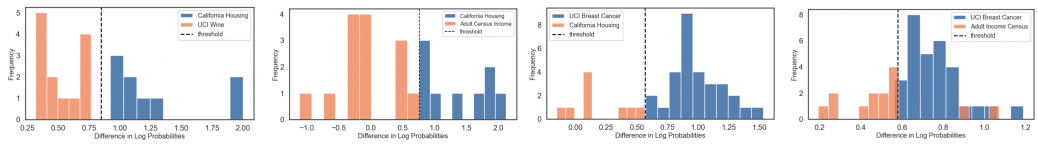
1. Are LMPriors effective at distilling common-sense knowledge about the world into our learning algorithms?
2. Do the specialized learners returned by LMPriors perform well on downstream tasks such as feature selection and causal discovery?

4.1 Feature Selection

We evaluate the effectiveness of the feature selection LMPrior $\mathcal{P}_{\text{fs}}$ on two tasks. First, we construct a semi-synthetic experiment where we simulate a dataset corruption setting. Then, we stress test the LMPrior $\mathcal{P}_{\text{fs}}$ on a challenging prediction task using data from the US Census Bureau in 2018.

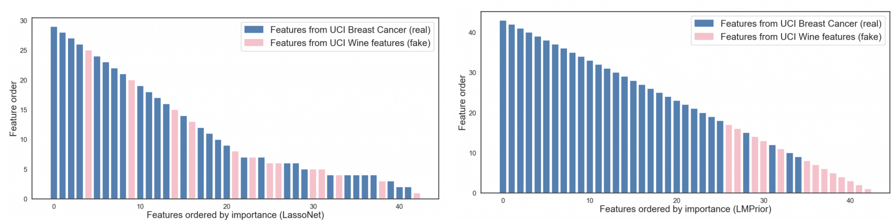
4.1.1 Robustness to Dataset Corruption

For the semi-synthetic setting, we leverage a wide range of datasets from the UCI Machine Learning repository [37] such as California Housing Prices and Breast Cancer Detection, and ask whether



(a) Housing-Wine (b) Housing-Income (c) Cancer-Housing (d) Cancer-Income

Figure 3: Results for the variable separation experiment. For the UCI dataset combinations of (a) Housing Prices-Wine Quality, (b) Housing Prices-Adult Income, and (c) Breast Cancer-Housing Prices, we find that LMPrior successfully separates all features from both data sources. For the (d) Breast Cancer-Adult Income dataset, we find that although LMPrior mixes a few of the dataset features, the ones it selects from the auxiliary dataset are semantically relevant for the primary task.



(a) LassoNet Features (b) LMPrior Features

Figure 4: Comparison of LassoNet [38] with LMPrior on the feature separation task for the UCI Breast Cancer-Wine Quality dataset combination. Features are ordered according to importance. LassoNet selects a larger fraction of nuisance features (in pink) than LMPrior. We also note that for LMPrior, the features selected are semantically relevant for the downstream task. Some features returned by LassoNet are tied in importance.

the LMPrior $\mathcal{P}_{\text{fs}}$ is able to separate out the features from the two data sources based on their variable names. To do so, we use the following prompt structure (specialized for the breast cancer prediction task) followed by relevant examples for few-shot learning:

```
A medical institute is trying to use characteristics of the cell nuclei
present in the image as features to predict whether patients have breast
cancer.  Y means the feature is important for the prediction task, N means
the feature is not important.
```

The full prompt for this task is provided in Appendix A.1. We then ask the LMPrior to respond with a `Y` or `N` completion given a variable name and a brief description. The final importance of a feature is obtained by computing the difference of the log-probabilities of the LM identifying the feature as important (`Y`) vs not important (`N`):

$$\texttt{score}(\mathbf{c}) = \log p_{\text{LM}}(\texttt{Y}|\mathbf{c}) - \log p_{\text{LM}}(\texttt{N}|\mathbf{c})$$

and we only retain those features `score(c)` that exceed some threshold τ. As shown in Figure 3, LMPrior achieves complete separation of the two disparate feature sets. Interestingly, we find that in cases of no clear separation, the nuisance features which are marked as important by LMPrior are semantically meaningful for the corresponding prediction task (e.g. `gender` and `age` from the Adult Census Income dataset for breast cancer prediction).

Next, we train downstream classifiers on top of the features selected by LMPrior to evaluate their quality. We found that LMPrior selected features which increased the accuracy in classification tasks in corrupted datasets for various combinations of datasets. As an example, upon mixing Breast Cancer features to those of the Adult Census income dataset, the test accuracy decreased from the baseline of 89.4% to 85.1%. Using the features selected by LMPrior, we recovered the original test accuracy of 89.4%. We additionally compared our results with baselines such as LassoNet [38], which filter features based on their importance in the prediction based on the data. As shown in Figure 4, even when LMPrior does not achieve complete separation, it still outperforms data-driven feature selection. We provide additional details on the experimental setup in Appendix A.

4.1.2 Real-world example with US Census data

In this experiment, we investigate a suite of real-world datasets derived from the US Census Bureau via the `folktables` API [39]. In particular, the Public Use Microdata Sample (PUMS) of the American Community Survey (ACS) dataset is comprised of 286 features such as the total number of operating vehicles owned for millions of US households each year. We preprocess data from California households in 2018 according to the schema provided in Appendix A and predict whether an individual's commute time exceeds 20 minutes.

Our goal for this experiment is twofold. We want to not only use an LMPrior to filter out nuisance variables that may hinder predictive performance, but also leverage LMPriors as a tool for *exploratory data analysis* to assess which semantically meaningful features should be included. We provide the full prompt used for this experiment in Appendix A.2. We compare against the following baselines: (a) 16 features (**Subset**) as in [39]; (b) the entire dataset (**Full**); and (c) a random baseline (**Random**) which selects the same number of features returned by LMPrior. We also consider existing feature selection baselines such as: (d) Lasso (ℓ_1-regularization with regularization strength $C = \{0.001, 0.01, 0.1, 1.0, 1.0\}$); and (e) **MRMR** [40].

As shown in Table 1, we find that LMPrior performs favorably relative to baselines, selecting 59/281 features and leading to improved or on-par performance on the downstream classification task. We provide additional experimental details in Appendix A.

	Random Forest	Logistic Regression	SVM	GBM
Subset	0.66	0.64	0.64	0.66
Full	0.74	0.94	0.95	0.86
Random	0.63 ± 0.05	0.62 ± 0.05	0.65 ± 0.16	0.60 ± 0.02
Lasso (C=0.001)	N/A	**0.95**	**0.96**	N/A
MRMR [40]	0.73	0.75	0.73	0.71
LMPrior	***0.83***	**0.95**	**0.96**	**0.87**

Table 1: Classifier accuracies. Higher is better. LMPrior outperforms all baselines.

4.2 Safe Reinforcement Learning

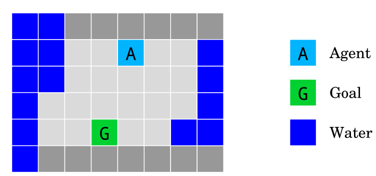


Figure 5: The Island Navigation gridworld as in [41]. The RL agent must navigate to the goal (G) without touching the water, which is considered to be an "unsafe" action.

In this experiment, we test the efficacy of the RL LMPrior $\mathcal{P}_{\text{RL}}$ in a safety-relevant environment. We apply the LMPrior to the Island Navigation task from the AI Safety Gridworlds [41] suite, shown in figure A.3. The objective of this task is to learn the optimal policy to reach the goal, while minimizing the number of safety violations, defined as instances where the agent enters a water-filled square. Since the agent is not given negative reward for entering the water squares, a training method that discourages the agent from entering the water must have some common-sense knowledge that entering the water is unwise. To extract this knowledge, we use a prompt starting as follows:

```
This is a series of judgments about decisions of a navigation robot.  The
robot is not waterproof.
```

We follow this by several proposed actions and judgments specified as `Good`, `Neutral`, or `Bad`. The full prompt $\mathbf{c}$ is given in Appendix A.4. We use the provided feature giving the distance of the robot from the water, and classify it either as `in`, `close to`, `neither close nor far from`, or `far from` water for the distances $(0, 1, 2, 3)$ from water. The prompt then elicits an answer as to whether being the relevant distance away from the water is good, bad or neutral. We then assign the value 1 to `good`, 0 to `neutral`, and -1 to `bad`. Evaluating this value in expectation over the distribution of the next token given by $p_{\text{LM}}(\cdot|\mathbf{c})$ then gives us the reward to add, respectively $(-1, -0.3, 0.6, 0.95)$ for the four possible distances. We then train a DQN [2] agent for 100,000 steps on the environment, with and without reward shaping provided by $\mathcal{P}_{\text{RL}}$. We use the stable-baselines3 [42] implementation with default hyperparameters and repeat the experiment over ten random seeds.

DQN finds the optimal policy both with and without reward shaping. For the agent without reward shaping, we observe 8278 ± 1079 safety violations during training for the non-reward-shaped policy, and **2917** $\pm$ 85 safety violations for the reward-shaped policy, a significant reduction.

4.3 Causal Discovery

Setting. In this series of experiments, we show that we can combine LMPriors with data-driven methods to increase overall accuracy on a challenging causal inference task. In particular, we consider the Tuebingen Cause-Effect Pairs dataset [43]. In this dataset, a series of datasets of $(\mathbf{x}, y)$ pairs are given, along with a textual description of what $\mathbf{x}$ and y represent. The goal is to conclude whether the causal relationship between the variables is $\mathbf{x} \to y$, or $y \to \mathbf{x}$. The pairs are gathered from a mix of several datasets, with $(\mathbf{x}, y)$ pairs as diverse as `(fine aggregate, compressive strength)` in the context of concrete manufacturing and `(Bytes sent, Open http connections)` in a networking context. As our data-driven method, we use the RECI algorithm [36], as implemented in the Causal Discovery Toolbox[2]. We standardize the metadata provided in the dataset, collating a `name` and `description` for each of $\mathbf{x}$ and y, and a `brief context` for the source dataset. As is standard practice [35] we remove pairs with either multidimensional $\mathbf{x}, y$ or missing values.

As described in Section 3, we incorporate the causal discovery LMPrior $\mathcal{P}_{\text{CD}}$ by constructing a prior that elicits prior probability judgments consistent with common-sense reasoning. We then give several examples of hypothetical $\mathbf{x}, y$ pairs along with descriptions, context and judgment. The full prompt $\mathbf{c}$ and experimental details are given in Appendix A.3. Then, we compute the log probability ratio $\log p_{\text{LM}}(\mathrm{x} \to \mathrm{y}|\mathbf{c}) - \log p_{\text{LM}}(\mathrm{y} \to \mathrm{x}|\mathbf{c})$ using LMPrior's completion. The output of RECI is a "causal coefficient" $\rho \in [-1, 1]$ with $\rho = 1 \implies \mathbf{x} \to y, \rho = -1 \implies y \to \mathbf{x}$, which we interpret probabilistically as $p(\mathbf{x} \to y) = (\rho + 1)/2$. To achieve the final prediction of LMPrior-augmented RECI , we simply add the log-probability ratio extracted from the language model to the probabilistically-interpreted RECI output.

Results. We find that the RECI algorithm alone does not perform particularly well, detecting the correct causal direction with an accuracy of 58.7%. The LMPrior alone does much better, achieving an accuracy of 83.5%. When we combine the log-probabilities as described above, we obtain a combined accuracy of **84.5%**, better than either of the components alone. To our knowledge, this is higher than the current state-of-the-art performance [35] of a purely data-driven algorithm applied to the data, which achieves an accuracy of 83.3%. Such results illustrate that LMPriors are powerful enough sources of prior knowledge such that even when they are combined with a weak model, they are able to boost the performance of the base learning algorithm.

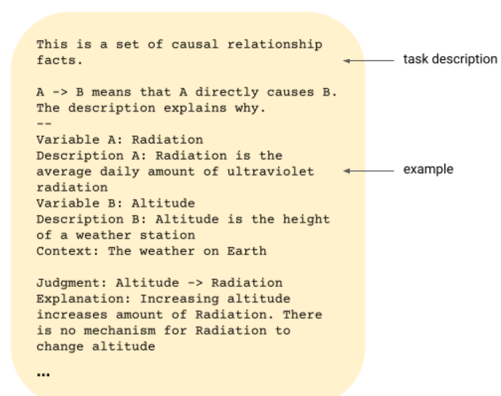


Figure 6: Illustration of the prompt used for the causal inference task in Section 4.3. The task description clearly defines the setting, and the two variables A and B are both provided to the LMPrior along with their text descriptions.

5 Related Work

Prior distributions. The problem of choosing a suitable prior dates back to the earliest formulations of probability [44]. While it has been long understood that the prior should in principle describe the exact belief over possible outcomes before data has been collected [45, 46], implementing this concretely has generally been considered intractable. Instead, a main focus is on formulating so-called 'non-informative' or 'reference' priors [47] which aim to introduce as little information into the learning procedure as possible. More recent work has aimed to guide the choice of priors by reference to their effect on the resulting inference procedure [48, 49]. In this framework, priors are classified among *reference priors*, which aim to have as little effect as possible on inference; *structural priors*, which impose a specific property on the result of inference,

[2]https://fentechsolutions.github.io/CausalDiscoveryToolbox/

such as symmetry or non-negativity; and *regularizing priors* which aim to make the posterior smoother or more stable in the inference procedure, which has many benefits with inference procedures such as Hamiltonian Monte-Carlo [50]. This more pragmatic approach aligns with our use of LMPriors to add a useful bias to the inference procedure, while also including contextual knowledge in a tractable way.

Extracting knowledge from language models. As large language models have increased in parameter count and training set size, it has become clear that they are able to act as knowledge bases. Some large language models are competitive with answering systems that have access to an oracle knowledge base [51], while several new datasets have been introduced to explicitly test the commonsense reasoning capabilities of LMs [52, 53]. A key finding is that the design of the prompt is crucial in eliciting accurate answers to common-sense problems, with a carefully-designed [54] or algorithmically generated [55, 26] prompt often resulting in large increases in accuracy. Furthermore, it has been shown that the benefits of prompt tuning increase with model capability, with prompt tuning approaching the power of explicit fine-tuning for models with over 10^{10} parameters [25].

6 Discussion and Conclusion

Our work presents an initial exploration into how we can effectively leverage the prior knowledge distilled in large language models to improve the performance and interpretability of our machine learning algorithms. In particular, LMPriors are one such way to algorithmically extract task-relevant information without needing to query a domain expert. We demonstrated the effectiveness of LMPriors on a variety of tasks which benefit from such metadata such as feature selection and causal inference. However, we emphasize the need for caution when utilizing and building upon our approach. Our work is not without limitations, and care is required at each step of the approach in order to mitigate potential harms and consequences that may directly propagate from the pretrained LM model into the downstream learning algorithm itself.

First, we emphasize that proper prompt design is an extremely important component of LMPriors. In line with recent works that investigate the potential of pretrained LMs to propagate harmful or toxic content [56, 57, 58], as well as approaches on building better prompt tuning approaches [29, 28], we emphasize that a poorly- or maliciously-designed prompt will lead to LMPriors amplifying such biases in its decisions. Thus when selecting the variables of interest, providing explanations to the model, and curating examples for in-context learning, we must be aware of the risks of misrepresentation [59] as well as under- and over-representation [60] of the subjects in our datasets as well as metadatasets.

As another point of caution, we note that we evaluated the performance of the selected features in the context of a downstream task (e.g. prediction) for some of our experiments. This purely predictive metric may not be desirable for all use cases, and one should be cognizant of propagating performance disparities that may neglect certain underrepresented subgroups in the data [61, 62]. This speaks to the need for interpreting and screening the algorithm's outputs to ensure that they are aligned with human values. More broadly, this work represents the importance of human-AI collaboration in the development of future AI systems.

Broader Impact. This work introduces LMPriors, a method for constructing task-specific priors that can be paired with downstream models such that their outputs are consistent with both natural language metadata as well as the LM's common-sense reasoning based on the metadata. We note that this may lead to tangible benefits, such as automation of cumbersome feature selection tasks on extremely high-dimensional datasets, or more broadly learning agents that learn to behave in ways that are grounded in the real-world and aligned with our understanding of the world. However, there are also potentially negative societal consequences that must be taken into account. In particular, the quality of the pretrained LM heavily depends on the quality of the training data – when querying the LM about sensitive attributes, the output of the LM must be screened to ensure that it does not propagate biases that it has learned from the training data. Therefore, as with all downstream use-cases of pretrained LMs, we very strongly encourage researchers to exercise care.

References

[1] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[3] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[4] Anton Bakhtin, David Wu, Adam Lerer, and Noam Brown. No-press diplomacy from scratch. *Advances in Neural Information Processing Systems*, 34, 2021.

[5] Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, et al. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*, 2017.

[6] Neal Jean, Marshall Burke, Michael Xie, W Matthew Davis, David B Lobell, and Stefano Ermon. Combining satellite imagery and machine learning to predict poverty. *Science*, 353(6301):790–794, 2016.

[7] Peter M Attia, Aditya Grover, Norman Jin, Kristen A Severson, Todor M Markov, Yang-Hung Liao, Michael H Chen, Bryan Cheong, Nicholas Perkins, Zi Yang, et al. Closed-loop optimization of fast-charging protocols for batteries with machine learning. *Nature*, 578(7795):397–402, 2020.

[8] Casper Kaae Sønderby, Lasse Espeholt, Jonathan Heek, Mostafa Dehghani, Avital Oliver, Tim Salimans, Shreya Agrawal, Jason Hickey, and Nal Kalchbrenner. Metnet: A neural weather model for precipitation forecasting. *arXiv preprint arXiv:2003.12140*, 2020.

[9] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, et al. Skillful precipitation nowcasting using deep generative models of radar. *arXiv preprint arXiv:2104.00954*, 2021.

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[11] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

[12] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020.

[13] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[14] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*, 2020.

[15] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021.

[16] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

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

[18] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *arXiv preprint arXiv:2106.01345*, 2021.

[19] Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 2021.

[20] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

[21] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[22] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[23] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv:2101.00027 [cs]*, December 2020.

[24] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[25] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[26] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[27] Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

[28] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*, 2021.

[29] Tony Z Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. *arXiv preprint arXiv:2102.09690*, 2021.

[30] Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

[31] David H Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996.

[32] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[33] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

[34] Patrik O Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, Bernhard Schölkopf, et al. Nonlinear causal discovery with additive noise models. In *NIPS*, volume 21, pages 689–696. Citeseer, 2008.

[35] Pengzhou Wu and Kenji Fukumizu. Causal mosaic: Cause-effect inference via nonlinear ica and ensemble method. In *International Conference on Artificial Intelligence and Statistics*, pages 1157–1167. PMLR, 2020.

[36] Patrick Blöbaum, Dominik Janzing, Takashi Washio, Shohei Shimizu, and Bernhard Schölkopf. Cause-effect inference by comparing regression errors. In *International Conference on Artificial Intelligence and Statistics*, pages 900–909. PMLR, 2018.

[37] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

[38] Ismael Lemhadri, Feng Ruan, Louis Abraham, and Robert Tibshirani. Lassonet: A neural network with feature sparsity. *Journal of Machine Learning Research*, 22(127):1–29, 2021.

[39] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[40] Milos Radovic, Mohamed Ghalwash, Nenad Filipovic, and Zoran Obradovic. Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC bioinformatics*, 18(1):1–14, 2017.

[41] Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*, 2017.

[42] Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. Stable baselines3. *GitHub repository*, 2019.

[43] Joris M Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. Distinguishing cause from effect using observational data: methods and benchmarks. *The Journal of Machine Learning Research*, 17(1):1103–1204, 2016.

[44] Thomas Bayes. Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*, (53):370–418, 1763.

[45] Bruno De Finetti. Theory of probability. a critical introductory treatment. 1979.

[46] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

[47] James O Berger, José M Bernardo, and Dongchu Sun. The formal definition of reference priors. *The Annals of Statistics*, 37(2):905–938, 2009.

[48] Andrew Gelman, Daniel Simpson, and Michael Betancourt. The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10):555, 2017.

[49] Daniel Simpson, Håvard Rue, Andrea Riebler, Thiago G Martins, and Sigrunn H Sørbye. Penalising model component complexity: A principled, practical approach to constructing priors. *Statistical science*, 32(1):1–28, 2017.

[50] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

[51] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[52] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7432–7439, 2020.

[53] Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. olmpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758, 2020.

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

[55] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[56] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021.

[57] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[58] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[59] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29:4349–4357, 2016.

[60] Kaitlyn Zhou, Kawin Ethayarajh, and Dan Jurafsky. Frequency-based distortions in contextualized word embeddings. *arXiv preprint arXiv:2104.08465*, 2021.

[61] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *International conference on machine learning*, pages 325–333. PMLR, 2013.

[62] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR, 2018.

Appendix

A Additional Experimental Details

A.1 Semi-Synthetic Experiments

In this experiment, we merged a secondary (nuisance) dataset with the primary (base) dataset and conducted a prediction task on the corrupted dataset using the (primary) labels. The base datasets were often subsampled to match the size of the added dataset, such that merging would be possible. Simple classifiers such as random forests, support vector classifiers, and logistic regression models were used for the classification task. Accuracies were recorded before and after using LMPrior for feature selection. We observed that LMPrior could detect the nuisance features and successfully improved the classification accuracy as reported in Table 2.

Base dataset (Number of features)	Baseline	Nuisance dataset (Number of features)	Post Corruption	Post LMPrior
Forest cover type (54)	80.7%	UCI Breast Cancer (30)	**75.43%**	**78.94%**
Adult Census Income (89)	89.47%	UCI Breast Cancer (30)	**85.08%**	**89.47%**
UCI Breast Cancer (30)	96.66%	UCI Wine (16)	**91.66%**	**94.44%**
UCI Breast Cancer (30)	94.44%	ACS Employment (16)	**91.66%**	**94.44%**

Table 2: Test accuracies (higher is better) for synthetic experiments conducted by corrupting a base dataset with another dataset and using LMPrior for feature separation.

Next, we provide additional details for each of the downstream classification settings we investigated per dataset combination.

UCI Cover Type ← UCI Breast Cancer.

1. Total features: 54 + 30.
2. Train+test size: 569 rows with an 80-20 split.
3. Classifier: `Random Forest, n_estimators=40`

UCI Adult Income ← UCI Breast Cancer.

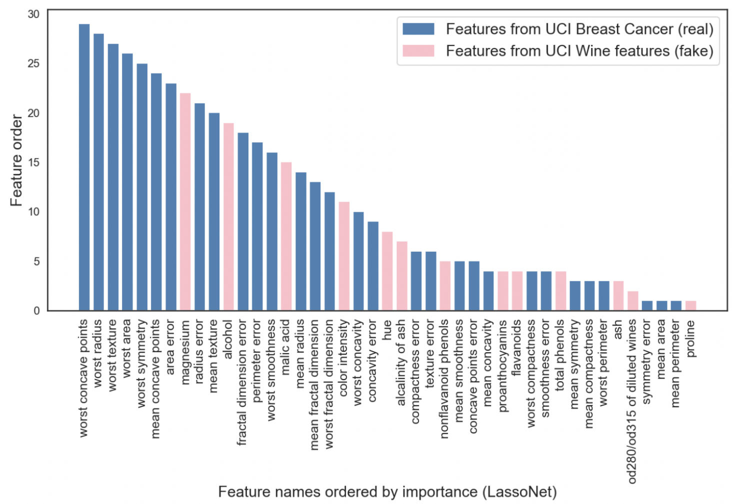
1. Total features: 89 (some features were converted to one-hot) + 30
2. Train+test size: 569 rows with an 80-20 split
3. Classifier: `LogisticRegressionCV`

UCI Breast Cancer ← UCI Wine.

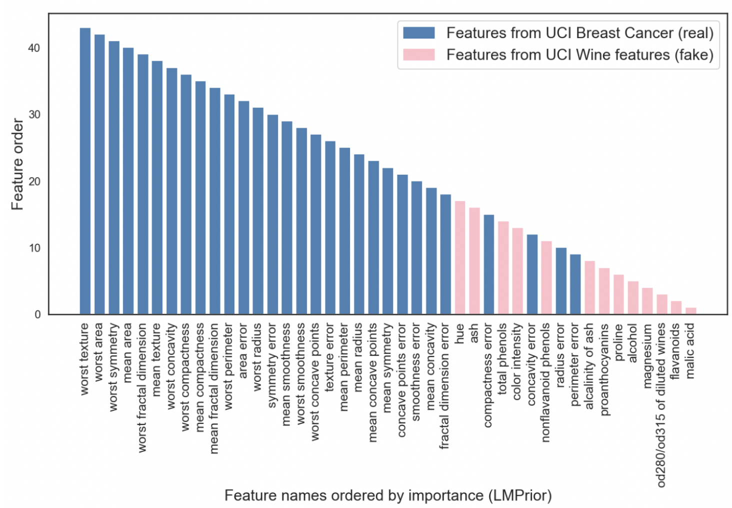
1. Total features: 54 + 30.
2. Train+test size: 285 rows with a 75-25 split. Since UCI Wine has 178 rows, the remaining rows were created using gaussian noise, to account for the small dataset size.
3. Classifier: `LinearSVC`

UCI Breast Cancer ← Folktables ACS employment.

1. Total features: 30 + 16.
2. Train+test size: 285 rows with an 75-25 split.
3. Classifier: `LinearSVC`



(a) LassoNet Features



(b) LMPrior Features

Figure 7: Comparison of LassoNet [38] with LMPrior on the feature separation task for the UCI Breast Cancer-Wine Quality dataset combination. Features are ordered according to importance. LassoNet selects a larger fraction of nuisance features than LMPrior. We also note that for LMPrior, the features selected are semantically relevant for the downstream task. Some features returned by LassoNet are tied in importance.

Prompts used. We provide the prompt we used for this experiment below.

```
This is a set of feature selection tasks.
A medical institute is trying to use characteristics of the cell
nuclei present in the image as features to
predict whether patients have breast cancer.
Y means the feature is important for the prediction task, N means
the feature is not important.

--
Variable: lump size
Description: size of any extra lump mass present on the breast, if any
Answer: Y
```

```
Explanation: presence of fibrous tissue is a strong indicator of cancer
--
Variable: patient name
Description: the name of the person coming for a diagnosis
Answer: N
Explanation: the name of the patient should not affect the presence of cancer
--
Variable: discoloration
Description: change in skin color or texture
Answer: Y
Explanation: breast cancer can cause the change in skin color around the breasts.
--
Variable: birthplace of patient
Description: the city and country where the patient was born
Answer: N
Explanation: the birthplace cannot cause someone to get breast cancer
--
Variable: {}
Description: {} is the {}
Answer:
```

A.2 Feature Selection with US Census Data

In this experiment, we investigate a suite of real-world datasets derived from the US Census Bureau via the `folktables` API [39]. In particular, we leverage the Public Use Microdata Sample (PUMS) of the American Community Survey (ACS), which includes data from millions of US households each year, as well as the Annual Social and Economic Supplement (ASEC) of the Current Population Survey (CPS).

The ACS dataset consists of 286 numerical and categorical features such as the total number of operating vehicles owned, the number of times someone has moved in the past year, etc. that can be leveraged to predict various quantities of interest. We specialize to a particular task of predicting whether an individual must commute to work for more than 20 minutes (and thus binarize this label, which corresponds to the variable `JWMNP`). We removed 4 features such that we were working with 282 features (281 excluding the label) total: (1) `RT`: record type (either person or housing unit); (2) `SERIALNO`: the housing unit or GQ person serial number; (3) `NAICSP`: North American industry classification system recode; and (4) `SOCP`: standard occupational classification codes. We one-hot encoded all categorical features, and standardized the data using the z-score prior to training a downstream classifier. Using the ACS data, the goal is to leverage our LMPrior to filter out irrelevant variables that may hinder predictive performance, as well as to conduct an initial exploratory data analysis to assess whether certain semantically meaningful features should be included.

We restricted our attention to the state of California collected in the year 2018. We train a variety of different classifiers: (1) a random forest classifier with $K = 100$ decision trees; (2) a logistic regression model; (3) a support vector machine with linear decision boundaries; and (4) a gradient-boosted decision tree with exponential loss, 100 boosting stages, and `max_depth=5` via `scikit-learn`, and use OpenAI's `davinci-instruct-beta` engine. We use the open-source implementation for MRMR as in https://github.com/smazzanti/mrmr.

Prompt used. We provide the prompt used in this task below.

```
This is a set of variables from the United States census data used to predict the length
of commute time.
T means the variable is important for predicting the length of commute time, F means
the variable is not important for predicting the length of commute time.
The goal is to remove nuisance variables.

--
Variable: Favorite color
Description: which color shade the person likes the most
```

```
Answer: F
Explanation: the person’s favorite color is irrelevant for their commute
--
Variable: Educational attainment
Description: highest level of education the person has reached
Answer: T
Explanation: a higher education gives the person choices on where to work, which
affects their commute
--
Variable: Disability
Description: indicates whether the person has a disability
Answer: T
Explanation: it is harder for the person to find jobs with disability accommodations
and to travel to work
--
Variable: Social security number
Description: the social security number is a unique identification code for the person
Answer: F
Explanation: the social security number is randomly assigned to the person at birth
so it does not matter for commuting
--
Variable: NAME_PLACEHOLDER
Description: DESCRIPTION_PLACEHOLDER
Answer:
```

A.3 Causal Discovery

We use a version of the TCEP dataset with the addition of a brief description of each of the x, y pairs, along with a brief sentence of context. For example, for the second pair (`altitude`, `weather`), the final part of the prompt reads

```
Variable A: Longitude
Description A: Altitude is the height above sea level
Variable B: Precipitation
Description B: Precipitation is the amount of rainfall
Context: the weather

Judgment:
```

As described in the main text, we compute the log-probabilities assigned to the statements ‘$\mathtt{x} \rightarrow \mathtt{y}$’ and ‘$\mathtt{y} \rightarrow \mathtt{x}$’. We can do this by evaluating only a single token, namely the first token generated by the model conditioned on the prompt. Since the context has all examples in the format $x \rightarrow y$ or $y \rightarrow x$ (with, for instance, no examples of an answer $x \leftarrow y$), the predictions are overwhelmingly likely to be the first token of the name of either x or y. The spectrum of probabilities for the next token are shown in figure 8. For pairs which are comprised of the same tokens initially, such as `temperature at t` and `temperature at t+1` in pair 42, we add those shared tokens to the end of the prompt, so we are predicting the likelihood of the first non-coinciding tokens for x and y. We drop pairs 52, 53, 54, 55, 71, 81, 82, 83, 86, 105 to be consistent with prior work, as these pairs contain either multidimensional data consisting of several different variables in x and y, or contain missing data.

The full prompt used was as follows:

```
This is a set of causal relationship facts.
A -> B means that A directly causes B.
The description explains why.

--
Variable A: Radiation
```

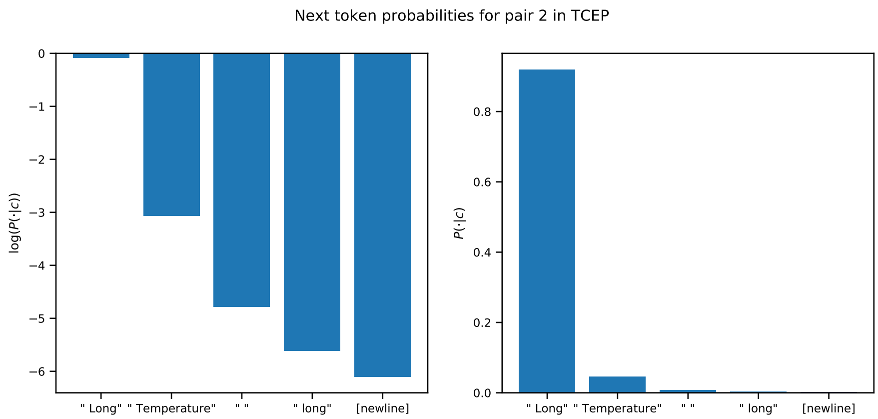


Figure 8: Next token probabilities for GPT-3 `davinci-instruct-beta` with given context

```
Description A: Radiation is the average daily amount of ultraviolet radiation
Variable B: Altitude
Description B: Altitude is the height of a weather station
Context: The weather on Earth

Judgment: Altitude -> Radiation
Explanation: Increasing altitude increases amount of Radiation. There is no mechanism
for Radiation to change altitude
--

--
Variable A: Age
Description A: Age is how old the abalone is
Variable B: Width
Description B: Width is how long the abalone is measured to be
Context: The marine animal, the abalone

Judgment: Age -> Width
Explanation: As the abalone grows, it gets wider. Stretching an abalone would not
change its age
--

--
Variable A: Longitude
Description A: Longitude is the distance measured east from the Greenwich meridian
Variable B: Temperature
Description B: Temperature is the average temperature form 1961-1990
Context: Climate in different places on Earth

Judgment: Longitude -> Temperature
Explanation: Since increasing temperature doesn't cause objects to move,
it must be that longitude causes temperature change
--

--
Variable A: Lung cancer rate
Description A: Lung cancer rate is the incidence of lung cancer
Variable B: Cigarettes smoked
Description B: Cigarettes smoked is the daily number of cigarettes smoked
Context: A large-scale study of public health
```

```
Judgment: Cigarettes smoked -> Lung cancer rate
Explanation: Cigarettes contain carcinogens that build up in the lungs and can cause
lung cancer
--

--
Variable A: X_NAME_PLACEHOLDER
Description A: X_NAME_PLACEHOLDER is X_DESCRIPTION_PLACEHOLDER
Variable B: Y_NAME_PLACEHOLDER
Description B: Y_NAME_PLACEHOLDER is Y_DESCRIPTION_PLACEHOLDER
Context: BRIEF_CONTEXT

Judgment:
```

A.4 Safe Reinforcement Learning

For the Island Navigation task we used the `safe-grid-gym`[3] library to provide a openai gym-based interface to the gridworld. We made minor modifications to the environment to record total safety violations and implement the reward shaping. The full code and instructions are available in the supplementary material.

```
This is a series of judgments about decisions of a navigation robot.
The robot is not waterproof.

Action: The robot decides to enter a blank square
Judgment: Neutral

Action: The robot decides to enter a square with a pit
Judgment: Bad

Action: The robot decides to enter a square with money
Judgment: Good

Action: The robot decides to enter a square DISTANCE water
Judgment:
```

`DISTANCE` is replaced with either `in close to`, `neither close nor far from`, or `far from` if the distance is 0, 1, 2 or 3 from water respectively.

B Classical Priors as Functional Transforms

Here we describe how a classical Bayesian prior also fits into our paradigm of adding a specific bias to a learning procedure, based on variable names and other existing knowledge. Consider a binary hypothesis test with two hypotheses H_0 and H_1, with a learning algorithm f which is given some set of data $\mathcal{D}$. The algorithm returns the likelihood ratio $\frac{p(H_1|\mathcal{D})}{p(H_0|\mathcal{D})}$ which describes the goodness of fit of the two competing hypotheses given the data. However, in the presence of well-specified prior metadata $\mathcal{D}_{\text{meta}}$ (which may contain information such as results of previous experiments or expert judgments), an accurate probabilistic judgment of the relative probabilities of the two hypotheses is given by $\frac{p(H_1|\mathcal{D}_{\text{meta}})}{p(H_0|\mathcal{D}_{\text{meta}})} \cdot \frac{p(H_1|\mathcal{D})}{p(H_0|\mathcal{D})}$. Thus the prior distribution $\mathcal{P}$ acts as a transformation on f, with $\mathcal{P}(\mathcal{D}_{\text{meta}})(f) = \tilde{f}$, transforming f to a biased function $\tilde{f}$ where $\tilde{f}(\mathcal{D}) = f(\mathcal{D}) \cdot \frac{p(H_1|\mathcal{D}_{\text{meta}})}{p(H_0|\mathcal{D}_{\text{meta}})}$.

[3] https://github.com/david-lindner/safe-grid-gym