Language Model Distillation: A Temporal Difference Imitation Learning Perspective

Zishun Yu[1*], **Shangzhe Li**[2*], **Xinhua Zhang**[1]

[1]Department of Computer Science, University of Illinois Chicago
[2]Department of Computer Science, University of North Carolina at Chapel Hill
zyu32@uic.edu, shangzhe@unc.edu, zhangx@uic.edu

Abstract

Large language models have led to significant progress across many NLP tasks, although their massive sizes often incur substantial computational costs. Distillation has become a common practice to compress these large and highly capable models into smaller, more efficient ones. Many existing language model distillation methods can be viewed as behavior cloning from the perspective of imitation learning or inverse reinforcement learning. This viewpoint has inspired subsequent studies that leverage (inverse) reinforcement learning techniques, including variations of behavior cloning and temporal difference learning methods. Rather than proposing yet another specific temporal difference method, we introduce a general framework for temporal difference-based distillation by exploiting the distributional sparsity of the teacher model. Specifically, it is often observed that language models assign most probability mass to a small subset of tokens. Motivated by this observation, we design a temporal difference learning framework that operates on a reduced action space (a subset of vocabulary), and demonstrate how practical algorithms can be derived and the resulting performance improvements.

Introduction

The successes of large language models (LLMs) have brought transformative advancements across many NLP tasks, albeit often at the cost of substantial computational resources due to their massive size. Model distillation is a widely adopted approach to compressing these large, high-capacity models (teachers) into smaller, efficient models (students). This practice has become standard, ranging from proprietary ones (Achiam et al. 2023) to open-source model families (Touvron et al. 2023; Grattafiori et al. 2024; Yang et al. 2024).

The seminal work of knowledge distillation (KD) proposed by Hinton, Vinyals, and Dean (2015) established the foundation for distribution-matching distillation methods, which were later extended to sequence/auto-regressive models (Kim and Rush 2016). In distribution-matching methods, at each time step during text generation, a student model $\pi : \mathcal{X} \to \Delta(\mathcal{V})$ seeks to approximate the probability distribution, over an vocabulary $\mathcal{V}$, produced by the teacher $\pi^\star$, for example by minimizing a (forward) KL-divergence (Sanh et al. 2019; Jiao et al. 2019; Wang et al. 2020) $\min_\pi \mathbb{KL}[\pi^\star(x_{\le t})\|\pi(x_{\le t})]$, where $x_{\le t} \in \mathcal{X}$ is a sequence of length t, and $\Delta(\Omega)$ denotes a probability simplex over space Ω.

Notably, this sequential distribution matching can be interpreted as behavior cloning (BC) (Pomerleau 1991), within the imitation learning (IL) and inverse reinforcement learning (IRL) literature (Abbeel and Ng 2004; Ng, Russell et al. 2000; Ziebart et al. 2008; Ho and Ermon 2016). While there are nuanced differences between IL and IRL, we use them interchangeably.

An important distinction between IL and distillation lies in the accessibility of the expert/teacher model $\pi^\star$. In IL settings, $\pi^\star$ is typically not available as a white-box model; instead, it is inferred empirically from expert demonstrations. A common estimate is given by $\hat{\pi}^\star(y \mid x) \propto \mathbb{E}_{(s,a)\sim\rho_{\mathbb{D}}}[\mathbb{1}_{\{(x,y)=(s,a)\}}]$, where $\rho_{\mathbb{D}}$ is the empirical distribution induced from a dataset $\mathbb{D}$. In contrast, it is reasonable to assume direct access to $\pi^\star$ as a white-box for distillation. This suggests that distillation is an even easier problem to IL, enabling the application of tools and insights from the broader IL literature.

Casting distillation as an IL problem is not new per se; indeed, many existing distillation works (Kim and Rush 2016; Agarwal et al. 2024; Gu et al. 2023; Wen et al. 2023) perform BC. Before reviewing these works through the lens of IL, we highlight two key distinctions that help characterize the distillation setup: (i) offline vs. online: This refers to whether distillation is performed on a fixed, pre-collected dataset (offline) or involves continuously generating new data during training (online). (ii) off-policy vs. mixed-policy: A method is considered purely on-policy if it uses only student-generated data for training; otherwise, it is categorized as off-policy. However, purely on-policy distillation is rare in practice—most methods use a mixture of teacher and student data. We refer to this setup as mixed-policy, though it is technically off-policy in the RL literature.

The pioneer work of sequence-level KD (SeqKD) (Kim and Rush 2016) is an offline off-policy BC. Follow up BC style works fall into two main categories: (i) online mixed-policy (Agarwal et al. 2024; Gu et al. 2023; Ko et al. 2024); and (ii) offline mixed-policy (Wen et al. 2023); potentially with different choice of divergence measure - for-

*These authors contributed equally.

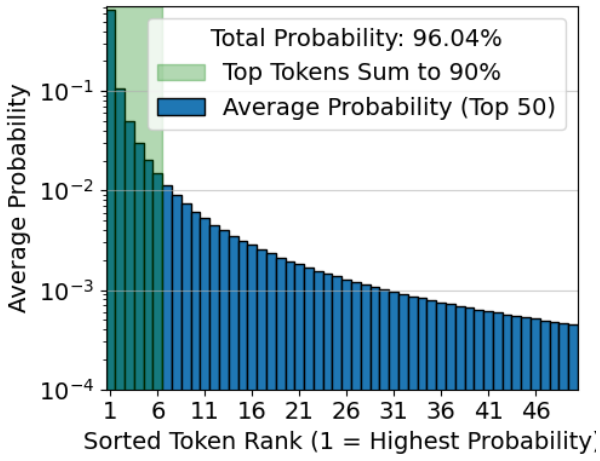

Figure 1: We average the sorted token probabilities across 20 sequences generated by Qwen-2.5 3B. Top 50 tokens account for 96% of the total mass, and the top 7 tokens contribute $\geq 90\%$.

ward/reverse KL (Kim and Rush 2016; Gu et al. 2023), Jensen-Shannon divergence (Agarwal et al. 2024), and general f-divergence (Wen et al. 2023).

However, BC is known to suffer from compounding errors (Ross, Gordon, and Bagnell 2011; Tu et al. 2022), also referred as exposure bias (Ranzato et al. 2015) in the context of auto-regressive models. To address these issues, the IL literature offers a rich toolbox beyond BC, for example generative adversarial training (Ho and Ermon 2016), occupancy matching (Syed, Bowling, and Schapire 2008), and more. We refer readers to Osa et al. (2018) for a comprehensive review. While not exhaustive, many modern IL methods (Ho and Ermon 2016; Ziebart et al. 2008) involve modeling an inverse reward function and then optimizing a policy with respect to it using RL. These RL algorithms are often based on temporal-difference (TD) learning (Sutton, Barto et al. 1998), which estimates the long-term impact of an action (or token) given a state (or sub-sequence) to mitigate the compounding errors.

Distillation may potentially benefit from the broader IL toolbox—particularly from TD learning methods. Earlier work has already demonstrated the potential of such approaches in smaller auto-regressive models. For instance, Yu et al. (2017) and Wu, Li, and Yu (2021) can be viewed as early applications of Ho and Ermon (2016). For modern LLMs, Jia (2024) explored TD-style IL methods for distillation, through value moment matching (Abbeel and Ng 2004; Ziebart et al. 2008) in particular. However, applying TD learning to LLMs is generally challenging due to their vast state-action space $\mathcal{V}^T$ (Snell et al. 2022; Yu et al. 2023; Havrilla et al. 2023).

Instead of directly applying a specific TD-learning-based IL method to distillation, we ask a more fundamental question: is there an exploitable structure unique to the distillation setting? A natural observation is that, for LLMs, the output distribution over the vocabulary is often sparse—only a small subset of tokens typically receive significant probability mass. An illustrative example is shown in Figure 1: For responses we generated, top 50 tokens account for 96% total mass, and top 7 tokens contribute 90%. Motivated by this observation, a natural idea is: during TD learning, it may be sufficient to consider only a small subset of candidate actions, reducing the effective action space from $\mathcal{V}$ to a much smaller set.

Preliminaries

MDP notations. We define the conventional MDP tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$ as follows. Let $w_{\leq t} = (w_1, w_2, \ldots, w_t) \in \mathcal{W}_t := \mathcal{V}^t$ be a sequence of tokens. We define a state and an action as $s_t := s_0 \circ w_{<t} \in \mathcal{S}$ and $a_t := w_t \in \mathcal{A}$, where $\circ$ denotes concatenation and s_0 is a prompt (or a query). We will sometimes use x and y as substitute of s and a whenever required by context, and use the conventional prime notation s', a' to denote s_{t+1}, a_{t+1} when only relative temporal index matters. The transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is defined as simple concatenation $s' = \mathcal{P}(s, a) := s \circ a$; $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is an unknown reward function; and $\gamma \in [0, 1]$ is a discount factor.

Policy. We denote the student policy as $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ (or parameterized π_θ) and the teacher model as $\pi^\star : \mathcal{S} \to \Delta(\mathcal{A})$, which may sometimes be referred to as the expert policy using IL terminology.

Soft value functions. The goal of soft RL is to maximize the expected return $J(\pi) = \mathbb{E}[\sum_t \gamma^t \tilde{r}_t \mid \pi, \mathcal{M}]$. The state-action value function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and the state value function $V^\pi : \mathcal{S} \to \mathbb{R}$ are essentially recursive definition of the return $J(\pi)$. In particular, we consider soft value functions (entropy-regularized) (Littman 1996; Ziebart et al. 2008) in this work, which are defined as follows:

$$\begin{aligned} Q^\pi(s, a) &:= \mathbb{E}[\textstyle\sum_{t=0}^{\infty} \gamma^t \tilde{r}_t \mid \pi, \mathcal{M}, S_0 = s, A_0 = a] \\ &= r(s, a) + \gamma V^\pi(s'), \qquad (1) \\ V^\pi(s) &:= \mathbb{E}_{a \sim \pi(s)}[Q(s, a) - \log \pi(a \mid s)] \qquad (2) \end{aligned}$$

where $s' = s \circ a$, and $\tilde{r}(s, a) := r(s, a) - \log \pi(a \mid s)$ is the entropy-regularized reward.

In an operator notation, the soft Bellman operator $\mathcal{B}_r^\pi : \mathbb{R}^{\mathcal{S} \times \mathcal{A}} \to \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$ is defined as $(\mathcal{B}_r^\pi Q)(s, a) = r(s, a) + \gamma V^\pi(s')$. Note this soft Bellman (evaluation) operator is a contractor (Haarnoja et al. 2018), in contrast to the non-contractive soft Bellman improvement operator (Littman 1996). We will throughout this paper use soft Bellman operator to denote $\mathcal{B}_r^\pi$, the policy evaluation operator.

Remark on $\mathcal{B}_r^\pi$. The contractive property of $\mathcal{B}_r^\pi$ defines a unique soft Q-function, for a reward function r, as the fixed point of $Q = \mathcal{B}_r^\pi Q$, given the Banach fixed-point theorem (Banach 1922). The contraction property of $\mathcal{B}_r^\pi$ later helps us to obtain our results in Section .

Occupancy measures. Let x be a state, the occupancy measure induced from a policy π is $\rho_x^\pi(s, a) := \mathbb{E}_\pi[\sum_{t=\tau}^{\infty} \gamma^{t-\tau} \mathbb{1}_{\{S_t = s, A_t = a\}} \mid S_\tau = x]$. Occupancy measures are convenient notation for formulating RL problems, though not a must, we introduce them to make the formulations more succinct. It is convenient as the value function can be written as $V^\pi(s) = \mathbb{E}_{(x,y) \sim \rho_s^\pi}[\tilde{r}(x, y)]$.

Max-entropy IRL. With these definitions, we are now ready to introduce our choice of concrete IRL algorithm,

IQL (Garg et al. 2021), which is an extension of maximum-entropy IRL (Ziebart et al. 2008). We start with the Generative Adversarial IL (GAIL) (Ho and Ermon 2016) objective function:

$$\min_{\pi} \max_{r} L(\pi, r) := \mathbb{E}_{\rho^\star}[r(s,a)] - \mathbb{E}_{\rho^\pi}[r(s,a)] - \mathcal{H}(\pi) - \psi(r), \tag{3}$$

where $\mathcal{H}(\pi) \coloneqq \mathbb{E}_{\rho^\pi}[-\log \pi(a \mid s)]$ and $\psi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}$ is a convex reward regularizer.

Inverse soft Q-learning. IQL (Garg et al. 2021) is a recent popular extension of GAIL, which is also the choice of algorithm of our paper. It's objective function is defined as:

$$\begin{aligned} \max_Q \min_\pi \mathcal{J}(\pi, Q) \coloneqq \mathbb{E}_{\rho^\star}[Q(s,a) - \gamma V^\pi(s')] \\ - (1-\gamma)\mathbb{E}_{s_0}[V^\pi(s_0)] - \psi(\mathcal{T}^\pi Q), \end{aligned} \tag{4}$$

$$\begin{aligned} \Leftrightarrow \quad \max_Q \mathcal{J}^\star(Q) \coloneqq \mathbb{E}_{\rho^\star}[\phi(Q(s,a) - \gamma V^Q(s'))] \\ - (1-\gamma)\mathbb{E}_{s_0}[V^Q(s_0)] \end{aligned} \tag{5}$$

where ϕ is a choice of concave regularizer; $\mathcal{T}^\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ is an inverse soft Bellman operator, defined as $(\mathcal{T}^\pi Q)(s,a) = Q(s,a) - \gamma V^\pi(s')$; and $V^Q \coloneqq \log \sum_a \exp Q(s,a)$.

The core idea is can be summarized as: (i) The original GAIL objective $\min_\pi \max_r L(\pi, r)$ can be equivalently swapped as $\max_r \min_\pi L(\pi, r)$ by strong duality, given proper regularization and feasible domains; (ii) For a fixed policy π, the saddle-point problem $\max_r \min_\pi L(\pi, r)$ for a fixed π is provably equivalent to $\max_Q \min_\pi \mathcal{J}(\pi, Q)$. This equivalence allows direct optimization in the value-function space, eliminating the need to explicitly model the inverse reward r; (iii) For a fixed Q, it is well-known that the entropy-regularized policy has a closed-form solution given by $\pi_Q = \exp Q(s,a) / \sum_a \exp Q(s,a)$. And Danskin's theorem (Danskin 2012) implies the saddle-point optimization $\max_Q \min_\pi \mathcal{J}(\pi, Q)$ can be casted as a simple maximization $\max_Q \mathcal{J}^\star(Q)$.

Remark on IQL. IQL reparameterizes the saddle-point formulation of IRL, enabling simpler and more stable optimization, and has thus recently gained popularity. We refer readers to (Ho and Ermon 2016; Garg et al. 2021; Al-Hafez et al. 2023), for a detailed derivation, background, and follow-up works. Throughout this paper, we use the deterministic form of IQL since the transition function $\mathcal{P}$ for LLMs is deterministic; the original stochastic formulation can be found in Garg et al. (2021).

Method

Notation Setup

Defining different sort of backup operators with different desired properties has been a significant string of works in the RL literature. We refer to Asadi and Littman (2017) and Miahi et al. (2024) for detailed discussions. We start with the example of soft Bellman operator used in IQL

$$(\mathcal{B}_r^\pi Q)(s,a) = r(s,a) + \gamma \mathbb{E}_{a' \sim \pi(s')}[Q(s',a') - \log \pi(a'|s')] \tag{6}$$

The future term $\mathbb{E}_{a' \sim \pi(s')}[Q(s',a') - \log \pi(a' \mid s')]$ is often called target. While evaluating the target, the candidate set of actions is the entire action space $\mathcal{A} = \mathcal{V}$, which is often huge.

Given the sparsity observation we made in Figure 1, a natural idea is to only use a subset of candidate actions that have a significant total density mass, say top-p tokens, we define the top-p candidate set:

Definition 1 (top-p candidate set)**.** *Given $\pi^\star$ and a state s, the top-p candidate set is defined as*

$$\begin{aligned} &\mathcal{A}_p^\star(s) \coloneqq \\ &\{a_i : \textstyle\sum_i \pi^\star(a_i \mid s) = p;\ \pi^\star(a_i|s) \geq \pi^\star(a_j \mid s)\ \forall\, i,j\} \subseteq \mathcal{A}. \end{aligned} \tag{7}$$

where actions are sorted in descending order of probability density (i.e., $\pi^\star(a_i|s) \geq \pi^\star(a_j|s)$ for all $i < j$). Thus, for state s, the set $\mathcal{A}_p^\star(s)$ consists of actions with the largest probabilities whose cumulative probability equals p, a.k.a. top-p tokens (Holtzman et al. 2019). Although it's uncommon to have a set whose cumulative probability equals exactly p, we use this notation for brevity, as the intended meaning is clear from context.

This top-p set filtered out the actions that have low densities w.r.t. the teacher $\pi^\star$. It is intuitive as the teacher model is a high-capable model and hence the reserved candidate tokens are high-quality.

Remark on $\mathcal{A}_p^\star(s)$. Note that $\mathcal{A}_p^\star(s)$ is a state-dependent set, which creates notational inconvenience for subsequent analysis. Without loss of generality, we simplify that $\mathcal{A}_p^\star$ is state-independent, it is easy to extend the following results to state-dependent $\mathcal{A}_p^\star(s)$. A simple construction is $\mathcal{A}_p^\star \coloneqq \cup_s \mathcal{A}_p^\star(s)$.

Instead proceeding to direct implementation with this top-p set, lets first define a top-p MDP $\mathcal{M}_p$ as a filtered counterpart of the original MDP $\mathcal{M}$.

Definition 2 (top-p MDP and $\bar{Q}$)**.** *Given a MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathbb{P}, r, \gamma)$ and a teacher policy $\pi^\star$, its top-p counterpart is $\mathcal{M}_p = (\mathcal{S}, \mathcal{A}_p^\star, \mathbb{P}, r, \gamma)$. To make the notation clearer, we use $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $\bar{Q} : \mathcal{S} \times \mathcal{A}_p^\star \to \mathbb{R}$ to denote Q-functions live in $\mathcal{M}$ and $\mathcal{M}_p$, respectively.*

With this definition, we aim to address an important question: **What's the performance of the optimal policy, if exist any, in $\mathcal{M}_p$ compared to the optimal policy in $\mathcal{M}$? Do we preserve proximal optimality when we use a subset of actions $\mathcal{A}_p^\star$?**

Let $\pi^\star$ and $Q^\star$ be the optimal policy and its corresponding soft Q-function in $\mathcal{M}$, and let $\pi_p^\star$ and $\bar{Q}_p^\star$ denote their counterpart in $\mathcal{M}_p$. As $\mathcal{M}_p$ is also a well-defined MDP, the existence of $\pi^\star, \pi_p^\star, Q^\star, \bar{Q}_p^\star$ follows directly from the fixed point theorem (Banach 1922). Note that we use $\pi^\star$ to refer to both optimal policy and the teacher model, as in IRL it is often assumed that the demonstrator (the teacher) is optimal.

Definition 3 (top-p projection)**.** *For any policy π, its top-p counterpart in the top-p MDP $\mathcal{M}_p$ is a state-wise projection* PROJ_p *onto the top-p candidate set $\mathcal{A}_p^\star$, for all state $s \in \mathcal{S}$.*

$$\text{PROJ}_p(\pi)(a \mid s) \coloneqq \begin{cases} \pi(a \mid s) / \sum_{a \in \mathcal{A}_p^\star} \pi(a \mid s), & \text{if } a \in \mathcal{A}_p^\star \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Remark on PROJ_p. $\text{PROJ}_p(\pi_p^\star) = \pi_p^\star$ by definition, and $\text{PROJ}_p(\pi^\star)$ is not necessarily $\pi_p^\star$. However, the projected policy $\text{PROJ}_p(\pi^\star)$ is handy for our analysis, see later Proposition 3 and 4.

With this projection definition, we are now ready to define a top-p version of soft Bellman operator.

Definition 4 (top-p soft Bellman operator)**.** *For any policy $\pi : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{A})$ and reward function r, We define a top-p counterpart $\mathcal{B}_p^\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}_p^\star} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}_p^\star}$ to the soft Bellman operator $\mathcal{B}_r^\pi$ as*

$$(\mathcal{B}_p^\pi \bar{Q})(s,a) = r(s,a) + \gamma \mathbb{E}_{a' \sim \text{PROJ}_p(\pi)}[\bar{Q}(s',a') \quad (9)$$

$$- \log \text{PROJ}_p(\pi)(a' \mid s')]. \quad (10)$$

The top-p operator $\mathcal{B}_p^\pi$ is defined by projecting π onto $\mathcal{A}_p^\star$. To characterize the optimality of $\mathcal{B}_p^\pi$, we define the supported q-norm to measure the magnitude of vectors w.r.t. the action set of interest $\mathcal{A}_p^\star$:

Definition 5 (supported q-norm)**.** *For $\Phi : \mathcal{S} \times \mathcal{Y} \to \mathbb{R}$, we define the supported q-norm by,*

$$\|\Phi\|_{q,\mathcal{Y}} := \max_s (\textstyle\sum_{a\in\mathcal{Y}} |\Phi(s,a)|^q)^{1/q}. \quad (11)$$

And slightly abusing notations, for $\Phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $\bar{\Phi} : \mathcal{S} \times \mathcal{A}_p^\star \to \mathbb{R}$, we define

$$\|\Phi - \bar{\Phi}\|_{q,\mathcal{A}_p^\star} := \max_s (\textstyle\sum_{a\in\mathcal{A}_p^\star} |\Phi(s,a) - \bar{\Phi}(s,a)|^q)^{1/q}. \quad (12)$$

Note that the supported q-norm is not exactly a norm because the definiteness does not hold. However, non-negativity, homogeneity, and especially the triangle inequality do hold.

Optimality in $\mathcal{M}_p$

Definition 6 (fixed point of $\mathcal{B}_p^\pi$)**.** *$\bar{Q} : \mathcal{S} \times \mathcal{A}_p^\star \to \mathbb{R}$ is a fixed point of $\mathcal{B}_p^\pi$ iff*

$$\bar{Q}(s,a) = (\mathcal{B}_p^\pi \bar{Q})(s,a) \quad \textit{for all } s \in \mathcal{S} \textit{ and } a \in \mathcal{A}_p^\star. \quad (13)$$

Proposition 1 (contraction)**.** *$\mathcal{B}_p^\pi$ is a contraction in the supported ∞-norm.*

We start with characterizing the gap between $\bar{Q}^{\text{PROJ}_p\pi^\star}$ and $Q^\star$, and then the characterization of $\bar{Q}_p^\star$, the optimal value function in $\mathcal{M}_p$, will follow directly from the sandwich condition 3.

Proposition 2. *Suppose $\kappa(p) := -\frac{\gamma}{1-\gamma}\log p$ and $\bar{Q}^{\text{PROJ}_p\pi^\star}$ is the fixed point of $\mathcal{B}_p^{\text{PROJ}_p\pi^\star}$, the sub-optimality $\|Q^\star - \bar{Q}^{\text{PROJ}_p\pi^\star}\|_{\infty,\mathcal{A}_p^\star} \le \kappa(p)$.*

Proposition 2 shows that the sub-optimality between $\bar{Q}^{\text{PROJ}_p\pi^\star}$ and $Q_p^\star$ is bounded. This suggests that the optimal policy $\pi^\star$ in $\mathcal{M}$ projected onto $\mathcal{A}_p^\star$, the action set of $\mathcal{M}_p$, is still a good policy. While projection is trivial in the tabular case, it is in general difficult to project a parameterized $\pi^\star$ onto the action set $\mathcal{A}_p^\star$. Instead, it is natural to implement an (inverse) RL algorithm in $\mathcal{M}_p$.

However, when implementing conventional IRL algorithms in $\mathcal{M}_p$, these algorithms typically seek to find the optimal policy $\pi_p^\star$ in $\mathcal{M}_p$ rather than the projected $\text{PROJ}_p\pi^\star$. The quantity of interest is the gap $\|Q^\star - \bar{Q}_p^\star\|_{\infty,\mathcal{A}_p^\star}$, which measures how well $\mathcal{M}_p$ approximates the original task $\mathcal{M}$ using the action subset $\mathcal{A}_p^\star$. This is intuitive: since $\pi_p^\star$ is optimal in $\mathcal{M}_p$, it should perform at least as well as any projected policy, including $\text{PROJ}_p\pi^\star$.

Formally, we first provide a sandwich condition in Proposition 3, and the desired gap, as shown in Proposition 4, trivially follows.

Proposition 3 (sandwich condition)**.** $\bar{Q}^{\text{PROJ}_p\pi^\star}(s,a) \le \bar{Q}_p^\star(s,a) \le Q^\star(s,a)$*, for all $(s,a) \in \mathcal{S} \times \mathcal{A}_p^\star$.*

Proposition 4 (bounded sub-optimality)**.** *Given a MDP $\mathcal{M}$ and its top-p counterpart $\mathcal{M}_p$, let $Q^\star$ and $\bar{Q}_p^\star$ be their optimal soft Q-functions, respectively, we have: $\|Q^\star - \bar{Q}_p^\star\|_{\infty,\mathcal{A}_p^\star} \le \kappa(p)$.*

The takeaway is: **The optimal policy $\pi_p^\star$ learned in $\mathcal{M}_p$ is provably near-optimal relative to $\pi^\star$ (the teacher), as established by Proposition 4**. This guarantee means one can deploy *any* IRL algorithm within $\mathcal{M}_p$ and (theoretically) find a policy whose performance closely matches that of $\pi^\star$, even though the action subset $\mathcal{A}_p^\star$ (for reasonable choice of p) is much smaller than the full action space $\mathcal{A}$, i.e. the raw vocabulary $\mathcal{V}$. This makes TD learning more efficient. In the next section, we show how to tailor a concrete IRL algorithm to our top-p MDP $\mathcal{M}$.

Implementation

We've demonstrated that: Optimal policy $\pi_p^\star$ in $\mathcal{M}_p$ yields a bounded sub-optimality 4 compared the optimal policy $\pi^\star$ in the original problem $\mathcal{M}$. Technically, any soft IRL algorithm could be applied to $\mathcal{M}_p$, by tailoring its backup operator through top-p projection (definition 3), as the example of top-p soft Bellman operator (definition 4).

We choose IQL (Garg et al. 2021) as our base IRL algorithm, and show that how to practically implement its top-p counterpart. Recall that the IQL objective is given by:

$$\max_Q \mathcal{J}^\star(Q) := \mathbb{E}_{\rho^\star}[\phi(Q(s,a) - \gamma V^{\pi_Q}(s'))]$$
$$- (1-\gamma)\mathbb{E}_{s_0}[V^{\pi_Q}(s_0)], V^{\pi_Q}(s) := \log \textstyle\sum_a \exp Q(s,a). \quad (14)$$

Often in (inverse) RL objectives, there are two functions Q and π depends on action a. Hence to constraint objective (14) (or any other objective) to the top-p candidate action set $\mathcal{A}_p^\star$, one should constraint these two functions to the candidate set.

Q-function masking. We only need to update Q-values of interests, i.e. $Q(s,a)$ for $a \in \mathcal{A}_p^\star$.

Definition 7 (top-p mask)**.** *For any $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we define a top-$p$ mask $\mathcal{F}_p^\star : \mathcal{S} \times \mathcal{A} \to \mathcal{S} \times \mathcal{A}$:*

$$(\mathcal{F}_p^\star Q)(s,a) := \begin{cases} Q(s,a), & \textit{if } a \in \mathcal{A}_p^\star \\ -\infty, & \textit{otherwise} \end{cases} \quad (15)$$

Applying $\mathcal{F}_p^\star$ leads to: $\max_Q \mathcal{J}^\star(Q) := \mathbb{E}_{\rho^\star}[\phi((\mathcal{F}_p^\star Q)(s,a) - \gamma V^{\pi_Q}(s'))] - (1-\gamma)\mathbb{E}_{s_0}[V^{\pi_Q}(s_0)]$.

Policy projection. We are also ready to handle the policy π_Q through projection. By definition $\pi_Q(a \mid s) =$

Algorithm 1: Bellman Distill

Input: Teacher model $\pi^\star$; Instruction set $\mathcal{D} = \{X\}$;Student model Q_θ; Pre-train dataset $\mathcal{D}^{PT}$.
Data Generation: Generate the teacher dataset $\mathcal{D}^\star = \{X, Y^\star\}$ using $\pi^\star$.
for each epoch **do**
  Sample mini-batches $\mathcal{D}^\star_{\text{mini}}, \mathcal{D}^{PT}_{\text{mini}}$ from datasets.
  Compute $\mathcal{F}^\star_p$ according to Definition 7 and $Q(s,a)$.
  Compute $\pi_Q(a \mid s)$ and $\text{PROJ}_p\pi_Q$ using Definition 3.
  Compute projected value function using $V^{\text{PROJ}_p\pi_Q}(s)$.
  Compute $\mathcal{J}^\star$ in Eq. 17 and $\mathcal{J}_{PT}$ for student update.
end for

$\exp Q(s,a)/\sum_a \exp Q(s,a)$, as a result its projected counterpart is $(\text{PROJ}_p\pi_Q) = \exp Q(s,a)/\sum_{\mathcal{A}^\star_p} \exp Q(s,a)$ for $a \in \mathcal{A}^\star_p$ otherwise 0.

Applying both Q-function masking and policy projection leads to:

$$\max_Q \mathcal{J}^\star(Q) \coloneqq \mathbb{E}_{\rho^\star}[\phi((\mathcal{F}^\star_p Q)(s,a) - \gamma V^{\text{PROJ}_p\pi_Q}(s'))] - (1-\gamma)\mathbb{E}_{s_0}[V^{\text{PROJ}_p\pi_Q}(s_0)] \quad (16)$$

Further IQL details. $\phi(\cdot)$ is a reward regularizer, we follow Garg et al. (2021) to use χ^2, corresponding to $\phi(x) = x - x^2/4\alpha$ for some coefficient α. We use $\alpha = 0.1$ for our experiments. In the IQL implementation, the second term in Eq. (14) is rewritten equivalently to a TD update through the telescopic identity (Garg et al. 2021). This identity states: for any π, valid occupancy measure μ, and value function V^π, we have $\mathbb{E}_{(s,a)\sim\mu}[V^\pi(s) - \gamma\mathbb{E}_{s'}V^\pi(s')] = (1-\gamma)\mathbb{E}_{s_0}[V^\pi(s_0)]$.

Let $\mu = \rho^\star$ and $\phi(x) = x - x^2/4\alpha$, the projected IQL objective is thereby given by:

$$\max_Q \mathcal{J}^\star(Q) \coloneqq \mathbb{E}_{\rho^\star}[\phi((\mathcal{F}^\star_p Q)(s,a) - \gamma V^{\text{PROJ}_p\pi_Q}(s'))] - \mathbb{E}_{\rho^\star}\left[V^{\text{PROJ}_p\pi_Q}(s) - \gamma\mathbb{E}_{s'}V^{\text{PROJ}_p\pi_Q}(s')\right] \quad (17)$$

where $V^{\text{PROJ}_p\pi_Q}(s) = \mathbb{E}_{\text{PROJ}_p\pi_Q}[Q(s,a) - \log \text{PROJ}_p\pi_Q]$. Note this is consistent with Eq. (14), since without projection we have $V^{\pi_Q}(s) = \mathbb{E}_{\pi_Q}[Q(s,a) - \log \pi_Q(a \mid s)] = \log \sum_a \exp Q(s,a)$.

Here the expectation w.r.t. $\rho^\star$ implies that data should be sampled from the teacher model $\pi^\star$.

In practice, we parameterize Q_θ with θ, the parameters of the student model, the student policy $\pi_\theta = \exp Q_\theta(s,a)/\sum_{a\in\mathcal{A}} \exp Q(s,a)$. Therefore, the Q_θ values are effectively the (constant-shifted) logits of student network, hence it is sufficient to have one model to serve as both Q_θ-function and student policy π_θ.

Further implementation details. Following Gu et al. (2023), we maximize a language modeling objective $\mathcal{J}_{PT} = \mathbb{E}_{(s,a)\sim\mathcal{D}^{PT}} \log \pi_Q(a \mid s)$ to retain performance on established NLP benchmarks. We refer to Algorithm 1 and Section 2.3 of Gu et al. (2023) for further details. We clip the Q values using a minimum value $Q_{\text{min}} = -10$ for numerical stability.

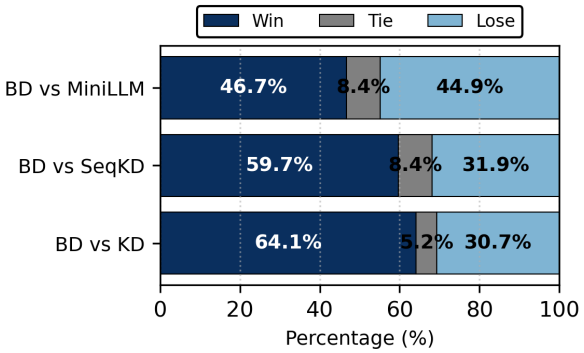

Figure 2: Comparison of win rates against KD, SeqKD, and MiniLLM baselines. We evaluate using GPT-4o-mini (OpenAI 2024) as the judging oracle, with Qwen-2.5 (3B) as the teacher model and a smaller (0.5B) model as the student. Results are based on 500 responses per distilled model, generated under the Dolly evaluation setting.

Experiments

In this section, we call our method as Bellman Distill (BD) for brevity.

Experiment Setup

We take instruction-following (Ouyang et al. 2022) as the conditional text generation task, where models are trained to generate responses according to the instructions. We finetune a large model on a dataset consisting of instruction-response pairs as the teacher model. Then, we compare different KD methods by evaluating the student model's instruction-following performance.

Model selections. We conduct experiments using three model families: the GPT-2 family (Radford et al. 2019), the OPT family (Zhang et al. 2022), and the Qwen-2.5 family (Yang et al. 2024). For the GPT-2 family, we use the 1.5B model as the teacher and the 120M, 340M, and 760M models as students. In the OPT family, the 6.7B model serves as the teacher, with the 125M, 350M, and 1.3B models as students. For the Qwen-2.5 family, we use the 3B model as the teacher and the 0.5B model as the student.

Training. We build our training dataset using `databricks-dolly-15K` (https://github.com/databrickslabs/dolly/tree/master), which contains 15,000 human-authored instruction-response pairs. To accommodate model constraints, we remove any samples that exceed the maximum context length. From the remaining data, we randomly select 500 examples for validation and 1,000 for testing, leaving around 12,500 samples for training. For the pretraining dataset $\mathcal{D}^{PT}$, we adopt OpenWebText (Gokaslan and Cohen 2019) for models in the GPT-2 and Qwen-2.5 families, and use the RoBERTa training corpus (Liu et al. 2019) for OPT models. Following the MiniLLM setup (Gu et al. 2023), we select hyperparameters based on Rouge-L (Lin 2004) scores evaluated on the validation set.

As our approach operates in an offline setting, we begin by generating responses from a teacher model using queries from dataset $\mathcal{D}$. Each query may yield multiple responses.

Method	GPT-2*			OPT			Qwen-2.5		
	Dolly	SelfInst	Vicuna	Dolly	SelfInst	Vicuna	Dolly	SelfInst	Vicuna
Teacher	1.5B			6.7B			3B		
	27.6	14.3	16.3	27.6	16.4	17.8	28.8	24.1	21.0
Student	120M			125M			0.5B		
SFT	23.2	9.9	14.3	23.2	9.9	14.3	24.5	16.5	17.6
KD	22.8	10.8	13.4	21.9	9.7	14.0	24.4	14.7	17.8
SeqKD	22.7	10.1	14.3	22.0	10.1	13.7	24.7	15.3	17.4
MiniLLM	24.6	13.2	**16.9**	23.8	10.2	**15.3**	26.7	19.1	20.5
BD (Ours)	**24.7**	**13.3**	16.2	**25.1**	**11.4**	14.8	**27.8**	**19.5**	**20.7**
Student	340M			350M			–		
SFT	25.5	13.0	16.0	23.6	10.6	15.5	–	–	–
KD	25.0	12.0	15.4	22.5	11.1	14.9	–	–	–
SeqKD	25.3	12.6	16.9	23.1	11.4	14.7	–	–	–
MiniLLM	25.4	**15.6**	**17.7**	24.3	11.5	17.9	–	–	–
BD (Ours)	**25.8**	15.4	16.5	**26.0**	**12.1**	**18.1**	–	–	–
Student	760M			1.3B			–		
SFT	25.4	12.4	16.1	26.0	11.4	15.6	–	–	–
KD	25.9	13.4	16.9	25.5	12.0	15.4	–	–	–
SeqKD	25.6	14.0	15.9	26.0	12.5	16.4	–	–	–
MiniLLM	**26.4**	15.9	**18.3**	26.2	13.7	**16.7**	–	–	–
BD (Ours)	26.2	**16.1**	17.3	**27.1**	**15.1**	16.3	–	–	–

Table 1: Main results: The Rouge-L scores, averaged over five random seeds, of our method and baselines across model families. Parameter sizes are listed at the top of each block. Best results per block are in bold. $*$: For GPT-2 class, the baseline results are directly duplicated from Gu et al. (2023).

These query-response pairs are then used to fine-tune the student models. Additional training details are provided in Appendix B.

Baselines choices. Following (Gu et al. 2023), we choose four baselines: (i) **SFT** directly fine-tunes the student model on $\mathcal{D}$ with golden responses; (ii) **KD** (Sanh et al. 2019; Song et al. 2020) fine-tunes the student model on $\mathcal{D}$ using the teacher distribution as supervision at each token step, also known as word-level KD; (iii) **SeqKD** (Kim and Rush 2016; Chiang et al. 2023; Taori et al. 2023; Peng et al. 2023; Zhou et al. 2023) fine-tunes the student model on the response sequences generated by the teacher model; (iv) **MiniLLM** (Gu et al. 2023) fine-tunes the student model using a policy gradient approach, where the reward is defined by the reverse KL divergence between the output distributions of the teacher and student.

Evaluation. Following (Gu et al. 2023), we evaluate the trained models on three instruction-following datasets: (i) **DollyEval**: The 500-sample test set we split from the `databricks-dolly-15K` dataset; (ii) **SelfInst**: (Wang et al. 2022a) A user-oriented instruction-following set with 252 samples; (iii) **Vicuna**: (Chiang et al. 2023) The 80 challenging questions used in the Vicuna evaluation.

Metrics. Following (Gu et al. 2023), we employ two complementary evaluation metrics: (i) **Rouge-L** (Lin 2004) to assess the precision of generated responses, following prior work demonstrating its effectiveness for large-scale instruction-following evaluation (Wang et al. 2022b); (ii) **Win rate** of our approach evaluated by a GPT-4o-mini (OpenAI 2024) oracle against baseline methods to measure generation quality.

A more detailed description on data generation and evaluation can be found in Appendix C.

Experimental Results

We present our experimental results in terms of Rouge-L scores for the GPT-2, OPT, and Qwen-2.5 model families in Table 1. We also demonstrate the win rate results in Figure 2. Our method consistently outperforms the KD and SeqKD baselines across all settings. While our approach performs comparably to MiniLLM on the GPT-2 family, it generally surpasses MiniLLM on the OPT and Qwen-2.5 families. Furthermore, the results highlight the scalability of our method across different student model sizes within all three families: as the student model size increases, the Rouge-L score also improves. This demonstrates the strong scalability and generalization ability of our approach across both model sizes and architectures.

Models	GPT-2 1.5B → 340M			OPT 6.7B → 125M			Qwen-2.5 3B → 0.5B		
	Dolly	SelfInst	Vicuna	Dolly	SelfInst	Vicuna	Dolly	SelfInst	Vicuna
1.0	24.9	15.2	15.7	23.6	10.5	14.2	26.6	18.1	18.6
0.8	25.8(+0.9)	15.4(+0.2)	16.5(+0.8)	25.1(+1.5)	11.4(+0.9)	14.8(+0.6)	27.8(+1.2)	19.5(+1.4)	20.7(+2.1)
0.5	25.6(+0.7)	15.5(+0.3)	16.1(+0.4)	24.4(+0.8)	11.0(+0.5)	14.7(+0.5)	27.6(+1.0)	19.2(+1.1)	20.2(+1.6)

Table 2: Impact of p: We demonstrate the impact of different choices of p across three models from distinct model families. Results are reported using the Rouge-L score and averaged over five random seeds. The performance gains relative to the baseline ($p = 1.0$, without top-p mask) are shown in green. It is evident that our top-p framework achieves consistent improvements.

Models	GPT-2 1.5B → 340M	OPT 6.7B → 350M	Qwen-2.5 3B → 0.5B
MiniLLM	11.2h	12.8h	10.7h
BD (Ours)	1.3h	3.5h	0.8h

Table 3: Training time comparison: Our approach achieves the optimal validation performance in significantly less training time compared to Gu et al. (2023), thanks to its offline training paradigm.

Models	GPT-2 1.5B → 760M			Qwen-2.5 3B → 0.5B		
	Dolly	SelfInst	Vicuna	Dolly	SelfInst	Vicuna
w χ^2 regularization	26.2	16.1	17.3	27.8	19.5	20.7
w/o χ^2 regularization	25.9	15.7	17.1	27.4	19.2	20.3

Table 4: χ^2 regularization: We demonstrate the effectiveness of incorporating χ^2 regularization into the distillation objective. Results reported are the Rouge-L score, averaged over five random seeds.

Impact of p

In this section, we analyze the impact of the top-p value in the top-p distillation setting. We conduct the analysis using GPT-2 340M, OPT 125M, and Qwen-2.5 0.5B models, with results shown in Table 2 for $p = 0.5$, 0.8, and 1.0. The table shows that omitting the top-p mask (i.e., using $p = 1.0$) leads to a notable drop in distillation performance. When $p = 0.5$, performance on OPT 125M is slightly lower than with $p = 0.8$, while the results are comparable for the other two models. Based on this observation, we adopt $p = 0.8$ in most experiments reported in Table 1, except when using OPT 1.3B as the student model, where $p = 0.5$ yields slightly better results.

Training Time

Online vs offline training. In general, the choice between online and offline training is a trade-off between computational cost and performance. While auto-regressive online generation is computationally expensive, offline training allows for the reuse of pre-collected datasets, making it more efficient. However, online training typically yields better final performance (Klein, Geist, and Pietquin 2011; Lee, Srinivasan, and Doshi-Velez 2019; Jarrett, Bica, and van der Schaar 2020; Garg et al. 2021). In particular, Wen et al. (2023); Ko et al. (2024) discussed how the offline setting can accelerate distillation, especially given the high cost of online generation for LLMs. We choose offline training due to computational constraints, although the proposed approach is inherently agnostic to the choice of training setting.

Wall-time comparison. To demonstrate the computational efficiency of our offline approach, we compare its training time to reach the optimal validation performance with the online method Gu et al. (2023). All experiments are conducted on 4×A40 GPUs. Results are shown in Table 3.

Ablation on the χ^2 Regularization

We perform an ablation study on the use of χ^2 regularization, as introduced in the implementation section, to assess its impact on distillation performance. Our empirical results show that incorporating χ^2 regularization improves the performance of the distilled student model. The study is conducted on two distillation setups: GPT-2 1.5B to 760M and Qwen-2.5 3B to 0.5B. The results are presented in Table 4.

Conclusion

Our work connects language model distillation to imitation learning in large discrete action spaces, where prior methods struggle without action space priors, while distillation benefits from teacher guidance—motivating our top-p TD learning approach. A key limitation is the shared vocabulary requirement between teacher and student for distribution matching. We propose a plug-and-play top-p TD framework that focuses on high-probability tokens, demonstrating empirical gains when integrated with IQL.

Acknowledgments

We sincerely thank the reviewers and the meta-reviewer for their time and constructive feedback. This work was supported by NSF Grant RI:2312955. We also gratefully acknowledge the NSF ACCESS program for providing computational resources under Award No. CIS250042.

References

Abbeel, P.; and Ng, A. Y. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, 1.

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Agarwal, R.; Vieillard, N.; Zhou, Y.; Stanczyk, P.; Garea, S. R.; Geist, M.; and Bachem, O. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

Al-Hafez, F.; Tateo, D.; Arenz, O.; Zhao, G.; and Peters, J. 2023. Ls-iq: Implicit reward regularization for inverse reinforcement learning. *arXiv preprint arXiv:2303.00599*.

Asadi, K.; and Littman, M. L. 2017. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, 243–252. PMLR.

Banach, S. 1922. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta mathematicae*, 3(1): 133–181.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; Stoica, I.; and Xing, E. P. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.

Danskin, J. M. 2012. *The theory of max-min and its application to weapons allocation problems*, volume 5. Springer Science & Business Media.

Garg, D.; Chakraborty, S.; Cundy, C.; Song, J.; and Ermon, S. 2021. Iq-learn: Inverse soft-q learning for imitation. *Advances in Neural Information Processing Systems*, 34: 4028–4039.

Gokaslan, A.; and Cohen, V. 2019. OpenWebText Corpus. http://Skylion007.github.io/OpenWebTextCorpus.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gu, Y.; Dong, L.; Wei, F.; and Huang, M. 2023. MiniLLM: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870. Pmlr.

Havrilla, A.; Zhuravinskyi, M.; Phung, D.; Tiwari, A.; Tow, J.; Biderman, S.; Anthony, Q.; and Castricato, L. 2023. trlX: A framework for large scale reinforcement learning from human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 8578–8595.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Ho, J.; and Ermon, S. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29.

Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; and Choi, Y. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Jarrett, D.; Bica, I.; and van der Schaar, M. 2020. Strictly batch imitation learning by energy-based distribution matching. *Advances in Neural Information Processing Systems*, 33: 7354–7365.

Jia, C. 2024. Adversarial Moment-Matching Distillation of Large Language Models. *arXiv preprint arXiv:2406.02959*.

Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; and Liu, Q. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Kim, Y.; and Rush, A. M. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, 1317–1327.

Klein, E.; Geist, M.; and Pietquin, O. 2011. Batch, off-policy and model-free apprenticeship learning. In *European Workshop on Reinforcement Learning*, 285–296. Springer.

Ko, J.; Kim, S.; Chen, T.; and Yun, S.-Y. 2024. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*.

Lee, D.; Srinivasan, S.; and Doshi-Velez, F. 2019. Truly batch apprenticeship learning with deep successor features. *arXiv preprint arXiv:1903.10077*.

Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, 74–81. Barcelona, Spain: Association for Computational Linguistics.

Littman, M. L. 1996. *Algorithms for sequential decision-making*. Brown University.

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

Miahi, E.; MacQueen, R.; Ayoub, A.; Masoumzadeh, A.; and White, M. 2024. Resmax: An Alternative Soft-Greedy Operator for Reinforcement Learning. *Transactions on Machine Learning Research*.

Ng, A. Y.; Russell, S.; et al. 2000. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, 2.

OpenAI. 2024. GPT-4o mini: advancing cost-efficient intelligence.

Osa, T.; Pajarinen, J.; Neumann, G.; Bagnell, J. A.; Abbeel, P.; Peters, J.; et al. 2018. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 7(1-2): 1–179.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Peng, B.; Li, C.; He, P.; Galley, M.; and Gao, J. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Pomerleau, D. A. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1): 88–97.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners.

Ranzato, M.; Chopra, S.; Auli, M.; and Zaremba, W. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Ross, S.; Gordon, G.; and Bagnell, D. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 627–635. JMLR Workshop and Conference Proceedings.

Sanh, V.; Debut, L.; Chaumond, J.; and Wolf, T. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Snell, C.; Kostrikov, I.; Su, Y.; Yang, M.; and Levine, S. 2022. Offline rl for natural language generation with implicit language q learning. *arXiv preprint arXiv:2206.11871*.

Song, K.; Sun, H.; Tan, X.; Qin, T.; Lu, J.; Liu, H.; and Liu, T.-Y. 2020. LightPAFF: A two-stage distillation framework for pre-training and fine-tuning. *arXiv preprint arXiv:2004.12817*.

Sutton, R. S.; Barto, A. G.; et al. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Syed, U.; Bowling, M.; and Schapire, R. E. 2008. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*, 1032–1039.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford Alpaca: An Instruction-following LLaMA model.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Tu, S.; Robey, A.; Zhang, T.; and Matni, N. 2022. On the sample complexity of stability constrained imitation learning. In *Learning for Dynamics and Control Conference*, 180–191. PMLR.

Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; and Zhou, M. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33: 5776–5788.

Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N. A.; Khashabi, D.; and Hajishirzi, H. 2022a. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Wang, Y.; Mishra, S.; Alipoormolabashi, P.; Kordi, Y.; Mirzaei, A.; Arunkumar, A.; Ashok, A.; Dhanasekaran, A. S.; Naik, A.; Stap, D.; et al. 2022b. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*.

Wen, Y.; Li, Z.; Du, W.; and Mou, L. 2023. F-divergence minimization for sequence-level knowledge distillation. *arXiv preprint arXiv:2307.15190*.

Wu, Q.; Li, L.; and Yu, Z. 2021. Textgail: Generative adversarial imitation learning for text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 14067–14075.

Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Yu, L.; Zhang, W.; Wang, J.; and Yu, Y. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Yu, Z.; Tao, Y.; Chen, L.; Sun, T.; and Yang, H. 2023. $\mathcal{B}$-Coder: Value-Based Deep Reinforcement Learning for Program Synthesis. *arXiv preprint arXiv:2310.03173*.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhou, C.; Liu, P.; Xu, P.; Iyer, S.; Sun, J.; Mao, Y.; Ma, X.; Efrat, A.; Yu, P.; Yu, L.; et al. 2023. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36: 55006–55021.

Ziebart, B. D.; Maas, A. L.; Bagnell, J. A.; Dey, A. K.; et al. 2008. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, 1433–1438. Chicago, IL, USA.