Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-26)
Special Track on AI and Robotics

# TeNet: Text-to-Network for Compact Policy Synthesis

**Ariyan Bighashdel**[1,2] and **Kevin Sebastian Luck**[2]
[1]Utrecht University
[2]Vrije Universiteit Amsterdam
a.bighashdel@uu.nl, k.s.luck@vu.nl

## Abstract

Robots that follow natural-language instructions often either plan at a high level using hand-designed interfaces or rely on large end-to-end models that are difficult to deploy for real-time control. We propose TeNet (Text-to-Network), a framework for instantiating compact, task-specific robot policies directly from natural language descriptions. TeNet conditions a hypernetwork on text embeddings produced by a pretrained large language model (LLM) to generate a fully executable policy, which then operates solely on low-dimensional state inputs at high control frequencies. By using the language only once at the policy instantiation time, TeNet inherits the general knowledge and paraphrasing robustness of pretrained LLMs while remaining lightweight and efficient at execution time. To improve generalization, we optionally ground language in behavior during training by aligning text embeddings with demonstrated actions, while requiring no demonstrations at inference time. Experiments on MuJoCo and Meta-World benchmarks show that TeNet produces policies that are orders of magnitude smaller than sequence-based baselines, while achieving strong performance in both multi-task and meta-learning settings and supporting high-frequency control. These results show that text-conditioned hypernetworks offer a practical way to build compact, language-driven controllers for resource-constrained robot control tasks with real-time requirements.

## 1 Introduction

Recent advances in large language models (LLMs), such as GPT [Brown *et al.*, 2020] and LLaMA [Touvron *et al.*, 2023], have demonstrated that natural language can act as a powerful and flexible interface across a wide range of domains. In robotics, this has led to growing interest in language-conditioned control, where robots are guided by natural-language instructions, often alongside perceptual inputs. Prominent examples include vision–language–action (VLA) systems such as PaLM-E [Driess *et al.*, 2023], SayCan [Brohan *et al.*, 2023], RT-2 [Zitkovich *et al.*, 2023], OpenVLA [Kim *et al.*, 2025], and OCTO [Team *et al.*, 2024]. These systems highlight how expressive language can be for specifying complex robotic behaviors. However, this expressiveness often comes at the cost of scale. **Many recent approaches to language-conditioned control rely on large end-to-end architectures, which can be computationally expensive and difficult to deploy in high-frequency control loops or on robots with limited onboard compute.**

At the other end of the spectrum are compact sequence models such as Decision Transformers (DT) [Chen *et al.*, 2021] and Prompt-DT [Xu *et al.*, 2022], which prioritize efficiency and ease of deployment in offline reinforcement learning. However, these models do not naturally incorporate language: instead, they rely on trajectory prompts or task-specific demonstrations to distinguish tasks, often requiring demonstrations even at test time and degrading as task diversity increases. This leaves a clear gap between expressive language-conditioned systems and compact policies that are efficient but are not language-enabled.

Several works attempt to bridge this gap by using language indirectly. Code-as-Policies [Liang *et al.*, 2023] translates instructions into robot API calls, while Code-as-Rewards [Venuto *et al.*, 2024] maps task descriptions into reward functions for reinforcement learning. Although effective in specific settings, these methods depend on predefined interfaces or accurate simulators, limiting their applicability to real-world robotics.

In this work, we ask a simpler question: *can language itself be used as a direct conditioning signal for policy instantiation?* Rather than executing a large language model inside the control loop, we use it once—at policy instantiation—through a hypernetwork [Ha *et al.*, 2016]. We introduce **TeNet (Text-to-Network)**, a framework that conditions a hypernetwork on LLM-derived text embeddings to generate compact, task-specific policies.[1] The resulting controller operates solely on low-dimensional state inputs, requires no demonstrations at inference time, and can run at high control frequencies on resource-constrained robots.

While direct text-to-policy instantiation is effective, we

[1]Project page, code, and supplementary materials are available at: https://sites.google.com/view/tenet-policy-synthesis.

find that performance improves when language is grounded in behavior. By aligning language representations with expert trajectories during training, task descriptions capture not only linguistic intent but also behavioral semantics, leading to stronger generalization in multi-task and meta-learning settings. Importantly, grounding is used only during training: at inference, policies are instantiated from text alone.

This paper takes a first step toward utilizing large robotic foundation models for resource-constrained robots through *language-enabled hypernetworks for compact policy synthesis*. While this could be applied to VLAs in the future, our first steps focus on low-dimensional, trajectory-based domains (Mujoco and Meta-World), isolating the role of language in policy instantiation without addressing perception.

In summary, our contributions are:

- **Text-to-Network Policy Generation.** We introduce TeNet, a framework that conditions a hypernetwork on LLM text embeddings to synthesize compact, task-specific robot policies suitable for real-time deployment.
- **Grounding Language in Behavior.** We show that aligning language with expert trajectories during training enriches linguistic representations with behavioral semantics and improves generalization in multi-task and meta-learning settings.
- **Empirical Insights into a New Paradigm.** We provide an extensive study on Mujoco and Meta-World benchmarks, highlighting both the promise and limitations of language-enabled hypernetworks and outlining paths toward future vision-grounded extensions.

## 2 Related Work

**Language-Conditioned Control in Robotics.** Large language models (LLMs) have recently been integrated into robotic systems to enable natural language instruction following and high-level planning. Early approaches such as SayCan [Brohan *et al.*, 2023] and PaLM-E [Driess *et al.*, 2023] leverage pretrained LLMs to map language into symbolic plans or action primitives executed by low-level controllers. These methods exploit LLMs' world knowledge but typically operate at a goal or planning level rather than synthesizing executable control policies.

Other works connect language and control indirectly. Code-as-Policies [Liang *et al.*, 2023] translates instructions into robot API calls, while Code-as-Rewards [Venuto *et al.*, 2024] maps task descriptions into reward functions for reinforcement learning. SayTap [Tang *et al.*, 2023] similarly maps language into structured locomotion patterns. While effective in constrained settings, these approaches rely on predefined interfaces or accurate simulators, limiting their generality. More recent vision-language-action systems, such as RT-2 [Zitkovich *et al.*, 2023], OpenVLA [Kim *et al.*, 2025], and OCTO [Team *et al.*, 2024], integrate language and perception in large end-to-end models, but their computational demands hinder deployment in resource-constrained or high-frequency control settings.

Several works also explore grounding language in behavior through representation learning. For example, CLASP [Rana *et al.*, 2023] learns joint language–state–action embeddings via contrastive objectives, focusing on representation pretraining rather than policy synthesis. In contrast, our use of alignment is auxiliary: language grounding serves to stabilize text-conditioned policy generation rather than constituting the primary modeling objective.

**Compact Sequence Models for Policy Learning.** A separate line of research explores compact sequence models as policies for reinforcement learning. The Decision Transformer (DT) [Chen *et al.*, 2021] formulates offline RL as conditional sequence modeling, generating actions autoregressively given states and return-to-go. While effective in single-task settings, DT lacks an explicit mechanism for task identification and therefore struggles in multi-task or meta-learning regimes.

Extensions such as Prompt-DT [Xu *et al.*, 2022] and Meta-DT [Wang *et al.*, 2024] introduce task-conditioning via trajectory prompts, improving generalization at the cost of requiring demonstrations at inference time. Diffusion-based approaches, including MTDiff [He *et al.*, 2023] and MetaDiffuser [Ni *et al.*, 2023], similarly condition on trajectories or task contexts to generalize across tasks. Although these methods demonstrate strong performance, their reliance on prompt trajectories limits scalability when demonstrations are unavailable or expensive.

In parallel, visuomotor diffusion policies such as Diffusion Policy [Chi *et al.*, 2025] generate actions directly from images and have shown impressive real-world results. These approaches differ fundamentally from the low-dimensional, state-based settings we consider. We therefore focus on DT-based baselines to maintain architectural comparability and isolate the role of language-conditioned policy instantiation.

Overall, compact sequence models demonstrate that lightweight architectures can scale to multi-task RL, but their dependence on trajectory prompts and lack of direct language grounding constrain their applicability as instruction-following agents.

**Hypernetworks and Policy Generation.** Hypernetworks [Ha *et al.*, 2016] generate the parameters of another network and have been widely explored for rapid specialization and meta-learning in reinforcement learning [Beck *et al.*, 2023]. Prior work conditions hypernetworks on a variety of signals, including structured task embeddings [Rezaei-Shoshtari *et al.*, 2023; Ren *et al.*, 2025], demonstration trajectories [Hegde *et al.*, 2024; Liang *et al.*, 2024], behavior descriptors or archives [Hegde *et al.*, 2023], robot morphology [Xiong *et al.*, 2024], or visual observations [Gklezakos *et al.*, 2022].

In parallel, language-conditioned hypernetworks have been studied in NLP to generate adapter or LoRA weights from task descriptions [Ye and Ren, 2021; Mahabadi *et al.*, 2021; Lv *et al.*, 2024; Charakorn *et al.*, 2025]. These methods focus on adapting large language models rather than synthesizing control policies.

Across these domains, existing approaches either rely on structured task descriptors, demonstrations, morphology signals, or use language only to adapt large models. None directly combine LLM-based text encoders with hypernetworks to synthesize compact, task-specific robot control policies.

**Summary.** Prior work has explored language-conditioned planning, compact sequence models, and hypernetwork-based policy generation. However, no existing approach directly instantiates executable robot policies from natural language via a shared hypernetwork. Our work fills this gap by using language as a conditioning signal for compact policy synthesis, grounded in behavior during training and executable without demonstrations at inference time.

## 3 Problem Statement

**Language-Augmented MDP (LA-MDP).** We model a single task as a Language-Augmented MDP

$$\tilde{\mathcal{M}} = (\mathcal{S}, \mathcal{A}, P, R, \mu, H, \mathbb{L}), \tag{1}$$

which extends a standard MDP by including a language descriptor. The first six elements $(\mathcal{S}, \mathcal{A}, P, R, \mu, H)$ are the standard MDP components: $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $P(s' \mid s, a)$ the transition dynamics, $R(s, a)$ the reward function, $\mu$ the initial state distribution, and $H$ the horizon. The additional component $\mathbb{L} \in \Delta(\mathcal{L})$ is a *language descriptor*, i.e., a probability distribution over natural-language strings in the space $\mathcal{L}$. Each task is associated with its own descriptor distribution $\mathbb{L}$, which generates natural-language paraphrases (e.g., "move forward" vs. "go straight") of the same underlying dynamics $P$ and reward function $R$. Thus, the LA-MDP can be viewed as a standard MDP augmented with a generative source of equivalent task descriptions. A policy $\pi(a \mid s)$ induces a trajectory distribution in $\tilde{\mathcal{M}}$, and its performance is

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{H-1} R(s_t, a_t)\right], \tag{2}$$

with the task-optimal policy $\pi^* = \arg\max_{\pi \in \Pi} J(\pi)$.

**Multi-task LA-MDP.** We consider a distribution over tasks, where each task $\tau \in \mathcal{T}$ is an LA-MDP

$$\tilde{\mathcal{M}}_\tau = (\mathcal{S}_\tau, \mathcal{A}, P_\tau, R_\tau, \mu_\tau, H, \mathbb{L}_\tau). \tag{3}$$

Tasks may differ in $\mathcal{S}_\tau, P_\tau, R_\tau, \mu_\tau$ and $\mathbb{L}_\tau$, while sharing the action space $\mathcal{A}$. The multi-task objective is to learn a single policy that maximizes expected return across tasks: $\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}_{\tau \sim p(\mathcal{T})}\big[J_\tau(\pi)\big]$.

**Offline setting.** No online interaction is permitted. The learner receives a static dataset collected from training tasks $\mathcal{T}_{\text{train}}$, each modeled as an LA-MDP

$$\mathcal{D}_{\text{train}} = \big\{ (\mathcal{X}_\tau, \mathcal{D}_\tau) \;\big|\; \tau \in \mathcal{T}_{\text{train}} \big\}, \tag{4}$$

where $\mathcal{X}_\tau = \{\xi_\tau^{(k)}\}_{k=1}^K$ is a set of expert trajectories $\xi_\tau^{(k)} = (s_0, a_0, r_0, \ldots, s_H)$, and $\mathcal{D}_\tau = \{d_\tau^{(m)}\}_{m=1}^M$ are i.i.d. descriptions sampled from the language descriptor, $d_\tau^{(m)} \sim \mathbb{L}_\tau$.

**Multi-task learning.** The learner is trained on demonstrations from a set of tasks $\mathcal{T}_{\text{train}}$. The objective is to learn a single model that approximates $\pi_\tau^*$ for all $\tau \in \mathcal{T}_{\text{train}}$, exploiting shared structure across tasks instead of training disjoint policies.

**Meta-learning.** The learner is trained on a collection of tasks $\mathcal{T}_{\text{train}}$ with the objective of generalizing to previously unseen tasks $\tau \in \mathcal{T}_{\text{test}}$. The challenge is to acquire transferable structure from $\mathcal{T}_{\text{train}}$ that enables rapid policy instantiation for new tasks without further environment interaction.

**Few-shot adaptation (baselines).** A common meta-RL strategy is to provide a small number of expert trajectories from the unseen task as adaptation data (few-shot setting). Prompt Decision Transformers (Prompt-DT) implement this by using short expert rollouts (*prompt trajectories*) as test-time task identifiers.

**Language-based instantiation (ours).** In contrast Prompt-DT, we do not rely on prompt trajectories; instead we leverage natural-language descriptions sampled from $\mathbb{L}_\tau$ to instantiate policies for $\tau \in \mathcal{T}_{\text{test}}$, requiring the learner to ground language into behavior.

## 4 Method

Our framework, **TeNet (Text-to-Network)**, synthesizes compact, task-specific robot policies directly from natural language descriptions by conditioning a hypernetwork on language embeddings. At training time (Figure 1, top), the model receives task descriptions and expert demonstrations. Task descriptions are first encoded into text embeddings. Expert demonstrations supervise the policy through an imitation loss. In the grounded variant, we additionally introduce a trajectory encoder, and align its embeddings with the text embeddings (i.e., language grounding), thereby enriching the language representation with behavioral semantics. At inference time (Figure 1, bottom), a new task description is passed through the text encoder, projected to the appropriate embedding space, and fed into the hypernetwork to generate a policy that can be executed without further demonstrations.

We present two variants of our approach: **Direct TeNet**, which conditions the hypernetwork solely on text embeddings, and **Grounded TeNet**, which aligns text embeddings with trajectory embeddings during training to capture behavioral semantics and improve generalization.

### 4.1 Direct TeNet

In the Direct TeNet variant, policies are instantiated directly from task descriptions without trajectory grounding. Given a description $d \in \mathcal{L}$, the text encoder $f_{\text{text}}$ produces an embedding $z_d = f_{\text{text}}(d) \in \mathbb{R}^{d_z}$. A projection network $g$ maps $z_d$ into the conditioning space of the hypernetwork: $\tilde{z}_d = g(z_d)$. The hypernetwork $h$ then generates the parameters $\theta_\pi$ of a task-specific policy network $\pi_{\theta_\pi}$

$$\theta_\pi = h(\tilde{z}_d), \qquad \pi_{\theta_\pi}(a \mid s). \tag{5}$$

Training relies on expert demonstrations $\xi_\tau = \{(s_t, a_t)\}_{t=0}^H$ from task $\tau$. The policy is supervised by behavior cloning (imitation learning)

$$\mathcal{L}_{\text{BC}} = -\mathbb{E}_{(s,a) \sim \xi_\tau}\big[\log \pi_{\theta_\pi}(a \mid s)\big]. \tag{6}$$

Thus, Direct TeNet provides a simple mechanism for mapping language directly into executable policies through the hypernetwork.

### 4.2 Grounded TeNet

Direct TeNet instantiates policies solely from projected text embeddings (Section 4.1). To better capture behavioral semantics, Grounded TeNet augments training with additional

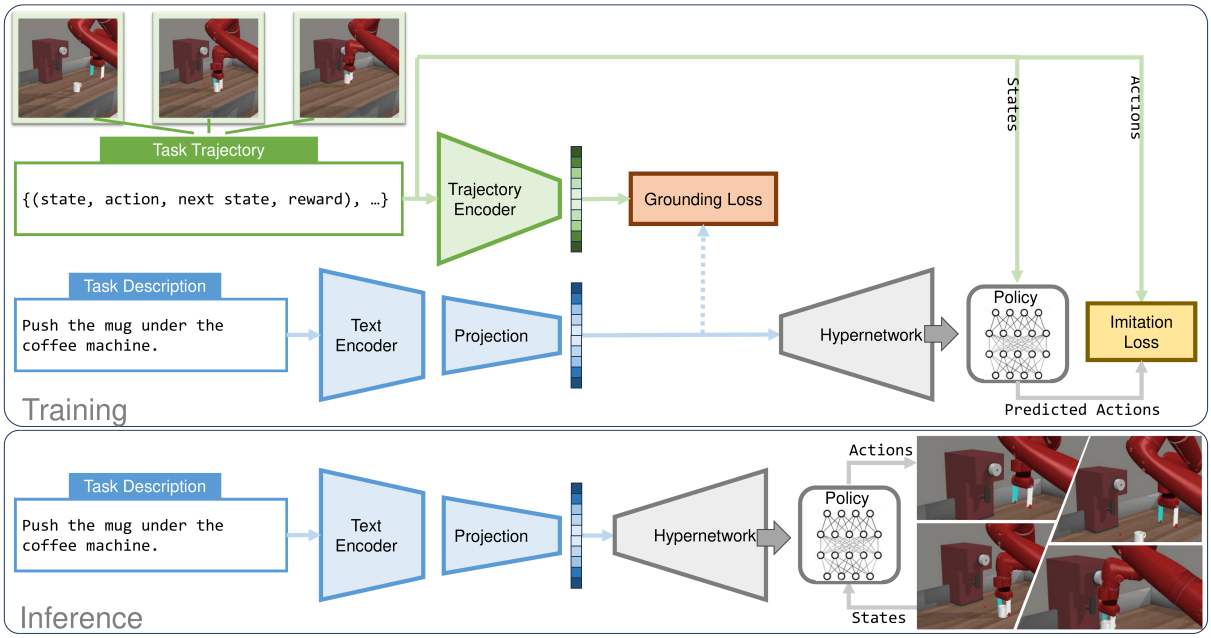


Figure 1: Training (top) and inference (bottom) of the proposed framework. During training, trajectories and task descriptions are encoded, projected, and aligned through a language grounding module, with a hypernetwork generating task-specific policies optimized by imitation and grounding losses. At inference, only the task description conditions the hypernetwork to instantiate a policy that maps states to actions.

grounding objectives that align text and trajectory embeddings. We emphasize that grounding is not the primary conceptual contribution of TeNet: it is an auxiliary mechanism that stabilizes and enriches the text embeddings, while the core novelty lies in generating executable policy parameters directly from natural language.

Given an expert trajectory $\xi = \{(s_t, a_t, r_t, s_{t+1})\}_{t=0}^{H}$, the trajectory encoder $f_{\text{traj}}$ produces an embedding $z_\xi = f_{\text{traj}}(\xi)$. Both $z_\xi$ and the projected text embedding $\tilde{z}_d$ are mapped into a shared space, and a grounding loss $\mathcal{L}_{\text{ground}}$ is applied. We explore two variants:

**Direct alignment (MSE).** A simple strategy is to directly minimize the squared distance between projected text and trajectory embeddings $\mathcal{L}_{\text{align}} = \mathbb{E}_{(d,\xi)}\left[\|\tilde{z}_d - z_\xi\|_2^2\right]$. This objective enforces absolute closeness of paired embeddings in the shared space.

**Contrastive alignment.** Let $\text{sim}(\cdot,\cdot)$ denote cosine similarity and $\beta > 0$ a temperature parameter. For each update, we consider a finite candidate set of trajectory embeddings $\mathcal{C}_\xi$ and a finite candidate set of text embeddings $\mathcal{C}_d$ that provide negatives for the contrastive normalization.

*(i) Text–trajectory contrastive (symmetric).* For paired $(\tilde{z}_d, z_\xi)$, we align text to trajectory and trajectory to text with a symmetric InfoNCE

$$\mathcal{L}_{\text{text-traj}} = \tfrac{1}{2}\,\mathbb{E}_{(d,\xi)}\left[ -\log \frac{\exp\left(\text{sim}(\tilde{z}_d, z_\xi)/\beta\right)}{\sum_{\xi' \in \mathcal{C}_\xi} \exp\left(\text{sim}(\tilde{z}_d, z_{\xi'})/\beta\right)} - \log \frac{\exp\left(\text{sim}(\tilde{z}_d, z_\xi)/\beta\right)}{\sum_{d' \in \mathcal{C}_d} \exp\left(\text{sim}(\tilde{z}_{d'}, z_\xi)/\beta\right)} \right]. \quad (7)$$

*(ii) Text–text contrastive.* Task descriptions can be structurally similar (e.g., differing only in goal parameters), which may collapse text embeddings. To encourage description-level discrimination, we add

$$\mathcal{L}_{\text{text-text}} = \mathbb{E}_d\left[ -\log \frac{\exp\left(\text{sim}(\tilde{z}_d, \tilde{z}_d)/\beta\right)}{\sum_{d' \in \mathcal{C}_d} \exp\left(\text{sim}(\tilde{z}_d, \tilde{z}_{d'})/\beta\right)} \right]. \quad (8)$$

The final contrastive objective is $\mathcal{L}_{\text{contrastive}} = \mathcal{L}_{\text{text-traj}} + \mathcal{L}_{\text{text-text}}$.

**Summary.** The total training loss combines imitation learning with grounding: $\mathcal{L} = \mathcal{L}_{\text{BC}} + \lambda_{\text{g}}\,\mathcal{L}_{\text{ground}}$, where $\mathcal{L}_{\text{ground}}$ may include $\mathcal{L}_{\text{align}}$ or $\mathcal{L}_{\text{contrastive}}$, and $\lambda_{\text{g}}$ balances their contribution. At inference time, no trajectories are required – the policy is instantiated from text alone. Grounding is used only during training to shape the representation.

## 5 Experiments

We conduct an extensive empirical study to evaluate TeNet and to provide insights into the design and behavior of language-enabled hypernetworks. Our experiments are performed on Mujoco control benchmarks (HalfCheetah-Vel, HalfCheetah-Dir, Ant-Dir) [Todorov *et al.*, 2012] and Meta-World manipulation benchmarks (ML1 Pick-Place, MT10, MT50) [Yu *et al.*, 2020], covering both multi-task and meta-learning settings.

Beyond reporting standard performance, our goal is to systematically answer a series of questions about when and why TeNet is effective, how grounding influences policy quality, and how design choices affect performance. This section is therefore organized around these questions, with results interleaved with analysis.

## 5.1 Experimental Setup

**Benchmarks.** We evaluate on *Mujoco* locomotion (HalfCheetah-Dir, HalfCheetah-Vel, Ant-Dir) and *Meta-World* manipulation (ML1 Pick-Place, MT10, MT50), spanning multi-task and meta-learning regimes. Full task definitions, state/action spaces, and splits are provided in the supplementary materials.

**Models.** We compare **DT** [Chen *et al.*, 2021], **Prompt-DT** [Xu *et al.*, 2022], and three TeNet variants: **TeNet** (direct, no grounding), **TeNet-MSE** (MSE grounding), and **TeNet-Contrast** (contrastive grounding). Implementation details, Prompt-DT size variants, and the Prompt-DT+Hypernetwork modification are provided in the supplementary materials.

**Metrics & protocol.** We report *episodic return* on Mujoco and *success rate* on Meta-World, plus *controller size* and *control frequency* for deployability. Results are averaged over 3 seeds; each task is evaluated with 50 rollouts.

**Defaults.** Unless stated otherwise: the text encoder is *Llama-3 8B* (frozen), the trajectory encoder is *Prompt-DT* (used only for grounded variants), and TeNet uses a small MLP hypernetwork to instantiate a $\sim$40K-parameter policy. Training is strictly offline.

## 5.2 Results

Figure 2 summarizes performance across all six benchmarks, with a shared legend shown on top.

Several general trends are clear. First, **DT** is consistently the weakest model across all domains, confirming that a compact sequence model without explicit task signals is not suitable for multi-task or meta-learning. Both **Prompt-DT** and **TeNet** address this limitation by providing task signals, but they do so in fundamentally different ways: Prompt-DT relies on short expert rollouts (prompt trajectories) as identifiers, while TeNet derives task signals directly from natural language descriptions. This text-based conditioning avoids the need for demonstrations at test time, making TeNet more scalable and practical, as it removes the requirement for task-specific trajectory prompts.

Second, when comparing **TeNet** variants (more specifically **TeNet-Contrast**) against **Prompt-DT**, we observe consistent advantages. TeNet-Contrast outperforms Prompt-DT in HalfCheetah-Dir and Ant-Dir, matches it in HalfCheetah-Vel, and is slightly worse in ML1 Pick-Place (which we analyze further in Section 5.2). Most strikingly, in MT10 and MT50 TeNet-Contrast *hugely outperforms* Prompt-DT. This large gap prompted us to investigate why Prompt-DT struggles in multi-task benchmarks and to identify which design choices in TeNet are responsible for its robust performance. We return to this question later in this section.

### Can We Directly Build Policies from Language, or Do We Need Grounding?

The results in Figure 2 reveal a mixed picture. Direct TeNet already provides a substantial improvement over DT across all benchmarks, confirming that natural language is an effective source of task signals. However, its relative performance compared to Prompt-DT depends critically on the setting. On **meta-learning benchmarks** (HalfCheetah-Vel, Ant-Dir, ML1 Pick-Place), Direct TeNet falls behind Prompt-DT, suggesting that text encodings, while informative, do not generalize to unseen tasks as effectively as trajectory prompts. In contrast, on **multi-task benchmarks** (MT10, MT50), Direct TeNet consistently outperforms Prompt-DT. These results indicate that *direct language-to-policy instantiation is viable* and scales well in diverse multi-task regimes, but that *grounding is required for robust generalization* in meta-learning settings where the agent must extrapolate to unseen tasks.

### How Should We Ground Language in Behavior?

The results in Figure 2 show that grounded TeNet, regardless of the chosen strategy, consistently outperforms Direct TeNet on the meta-learning benchmarks (HalfCheetah-Vel, Ant-Dir, ML1 Pick-Place). This confirms that additional grounding is necessary for robust generalization to unseen tasks.

Among the grounding methods, **contrastive alignment** generally performs better than direct alignment (MSE). The reason is that MSE enforces absolute closeness between paired text and trajectory embeddings, but provides no mechanism to separate embeddings from different tasks. As a result, embeddings from similar descriptions may collapse, limiting discriminability. In contrast, contrastive objectives simultaneously *pull together* matching text–trajectory pairs and *push apart* non-matching pairs, yielding a representation space that is both semantically aligned and better separated across tasks. This improved structure in the shared embedding space translates into stronger policy generalization.

### Why Does Prompt-DT Struggle in MT10 and MT50?

The Meta-World multi-task benchmarks (MT10 and MT50) contain tasks that are far more distinct than those in Mujoco (e.g., pick-place versus drawer-open, compared to velocity or direction variations). This task diversity poses a major challenge for Prompt-DT. Furthermore, as the number of tasks increases, the success rate of Prompt-DT drops (from 0.73 on MT10 to 0.61 on MT50; see Figure 2). To better understand this gap, we conduct two follow-up experiments.

First, we ask whether the failure is simply due to *insufficient model capacity*. If trajectory prompts are expressive enough, then increasing the size of Prompt-DT (from small to medium to large) should yield meaningful improvements. Table 1 shows that this is not the case: larger Prompt-DT models achieve only marginal gains, indicating that the issue lies deeper than model capacity.

Second, we test whether the limitation arises from the lack of *task-specific parameterization*. In this variant, Prompt-DT-HN serves as a trajectory-conditioned hypernetwork baseline, where the prompt trajectory is encoded and used to generate policy weights via a shared hypernetwork. To this end, we add a hypernetwork on top of Prompt-DT to generate policy parameters conditioned on task signals. Table 1 indicates that this modification yields a substantial boost in success rates on both MT10 and MT50. The comparison demonstrates that generating task-specific parameters is crucial when dealing with distinct multi-task benchmarks. TeNet naturally benefits from this principle while also being language-enabled, removing the reliance on demonstration prompts.

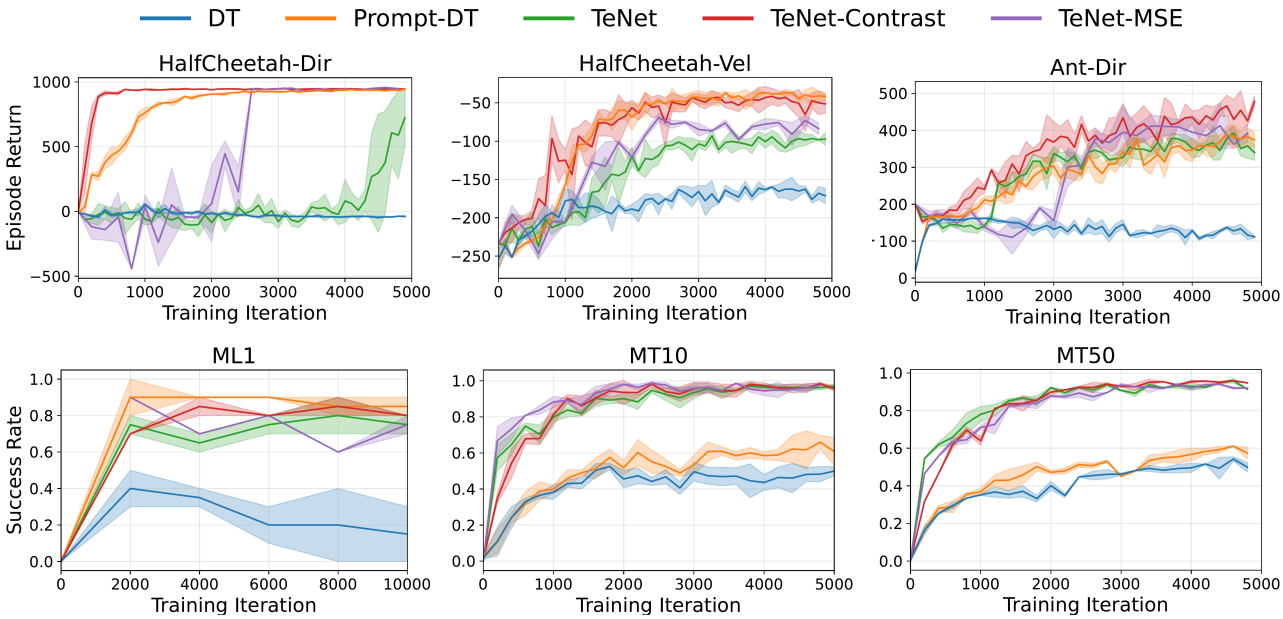


Figure 2: Performance across Mujoco (HalfCheetah-Dir, HalfCheetah-Vel, Ant-Dir) and Meta-World (ML1 Pick-Place, MT10, MT50). Each subplot reports mean and standard deviation over three seeds. A shared legend is shown at the top.

| Model | Success Rate | | Ctrl Size | Ctrl Freq. |
|---|---|---|---|---|
| | MT10 | MT50 | | |
| Prompt-DT-S | 0.73 | 0.61 | 1M | 557 Hz |
| Prompt-DT-M | 0.79 | 0.65 | 6M | 331 Hz |
| Prompt-DT-L | 0.74 | 0.58 | 39M | 190 Hz |
| Prompt-DT-HN | 0.99 | 0.97 | 5M | 462 Hz |
| TeNet | **0.99** | **0.98** | **40K** | **9300 Hz** |

Table 1: Success rate on MT10 and MT50, along with controller size and control frequency. Prompt-DT-S is the default configuration.

### How Fast Are TeNet Policies?

Beyond task success, deployability depends critically on the efficiency of the policy: controllers must be compact enough to fit on resource-constrained robots, and fast enough to support high-frequency control loops. Table 1 reports both the number of parameters (controller size) and the control frequency that the method can sustain.

The results highlight a stark contrast. Prompt-DT variants range from 1M to 39M parameters, with control frequencies between 190 Hz and 600 Hz. Adding a hypernetwork further increases model size to 5M parameters, while improving task success, but the resulting policies remain limited to the sub-kHz regime. In contrast, TeNet policies contain only **40K parameters** and sustain control rates of over **9 kHz**, more than an order of magnitude faster than all Prompt-DT baselines.

### Does Scaling the Number of Training Tasks Improve TeNet's Generalization?

In Section 5.2 we noted that TeNet-Contrast slightly underperforms Prompt-DT on ML1 Pick-Place. To investigate further, we study how scaling the number of training tasks affects generalization. Specifically, we vary the number of ML1 tasks available during training (50, 100, 200, 400, 800, 1600), while always holding out 10% of tasks for testing. The results are shown in Figure 3.

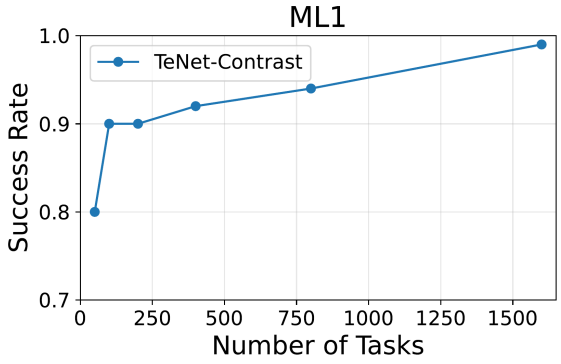


Figure 3: ML1 Pick-Place with varying numbers of tasks.

Performance improves steadily from a success rate of 0.80 with 50 tasks to 0.99 with 1600 tasks. This indicates that scaling the diversity of training tasks substantially enhances TeNet's ability to generalize. One possible factor is that as the number of training tasks grows, the domain gap between train and test tasks decreases, making generalization easier. In any case, reaching a success rate of **99%** with 1600 training tasks shows that TeNet can fully solve ML1 Pick-Place when provided with sufficient data. These results highlight both the promise and the data demands of language-enabled hypernetworks: like foundation models in other domains, TeNet benefits strongly from scale, even if it is data hungry.

### How Robust Is TeNet to Paraphrased Task Descriptions?

Since language is used only once to instantiate a policy, a key question is how robust TeNet is to variations in how a

| Level | Evaluation descriptions | LLaMA | BERT |
|---|---|---|---|
| L0 | Easy canonical (10/task) | 0.99 | 0.99 |
| L1 | Medium clauses (10/task) | 0.95 | 0.89 |
| L2 | Hard distractors (10/task) | 0.89 | 0.82 |

Table 2: MT10 success rates under paraphrasing.

task is described. In practice, semantically identical instructions may differ substantially in wording, syntax, or length. We therefore evaluate TeNet's sensitivity to paraphrasing by training on canonical task descriptions and testing on increasingly complex paraphrases. We conduct this study on MT10, comparing two text encoders—LLaMA [Touvron *et al.*, 2023] and BERT [Devlin *et al.*, 2019]. Models are trained using 10 canonical (Level 0) descriptions per task and evaluated on unseen paraphrases of growing linguistic complexity (Level 1 and Level 2), which differ syntactically but describe the same underlying task.

Table 2 shows that both encoders achieve similar performance on canonical descriptions, indicating that TeNet can reliably instantiate policies from simple instructions regardless of the encoder choice. However, as paraphrasing complexity increases, performance degrades for both models, with a larger drop observed for BERT. This gap suggests that richer language models produce more stable and semantically consistent embeddings under linguistic variation, leading to more reliable policy instantiation from natural language.

**Qualitative Evaluation: Velocity Following in HalfCheetah-Vel**

HalfCheetah-Vel evaluates velocity tracking by rewarding policies for matching a target forward speed, $r = -|v_{\text{current}} - v_{\text{target}}|$, so episodic return directly reflects tracking accuracy. Target velocities are defined on a grid from $0.075$ to $3.0\,\text{m/s}$. Models are trained on a subset of this grid and evaluated on unseen target velocities $\{0.225,\ 0.6,\ 1.2,\ 1.8,\ 2.025\}\,\text{m/s}$, as well as an out-of-distribution instruction at $3.5\,\text{m/s}$. Policies are instantiated from commands such as *"Move forward with target velocity X m/s."* and evaluated over 50 rollouts. Achieved velocity is computed as the average speed over the final 20 steps.

Figure 4 compares achieved versus instructed velocity for TeNet-Contrast and Prompt-DT. TeNet-Contrast closely tracks commanded speeds across all unseen test velocities, indicating smooth generalization over the continuous velocity range. For the extrapolated $3.5\,\text{m/s}$ instruction, both methods saturate near $\sim 3\,\text{m/s}$, reflecting the limits of the HalfCheetah dynamics rather than a failure of instruction following.

### 5.3 Summary of Empirical Insights

Our results show that direct text-to-policy instantiation is viable, but that grounding language in behavior is essential for robust generalization. Across meta-learning benchmarks, grounded TeNet variants consistently outperform Direct TeNet, with contrastive alignment providing stronger task discrimination than direct (MSE) alignment.

We further find that task-specific parameterization is critical in diverse multi-task settings. Prompt-DT degrades

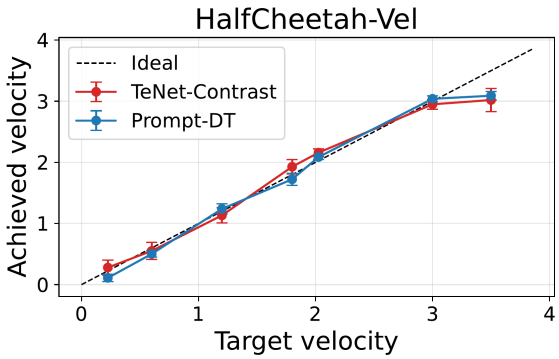


Figure 4: Velocity following in HalfCheetah-Vel.

sharply on MT10 and MT50, and increasing model capacity alone does not resolve this issue. In contrast, explicitly generating task-conditioned parameters—most effectively via language-conditioned hypernetworks—enables TeNet to scale to large and heterogeneous task sets.

Despite this added flexibility, TeNet remains highly efficient. Instantiated policies contain only ∼40K parameters and sustain control rates above 9 kHz, exceeding Prompt-DT baselines by more than an order of magnitude. TeNet also exhibits robustness to linguistic variation: performance degrades gracefully under increasingly complex paraphrases, with larger language models such as LLaMA producing more stable policy instantiations than BERT. Qualitative evaluation on HalfCheetah-Vel further confirms that text-instantiated policies accurately follow commanded velocities and generalize smoothly across unseen targets.

Additional ablation studies—analyzing the text–text contrastive term, grounded-flow, fine-tuning, and multiple task descriptions—are provided in the supplementary materials.

Overall, these findings indicate that compact, language-enabled hypernetworks can close much of the gap between lightweight sequence models and large language-conditioned systems within state-based, offline imitation settings. Extending TeNet to real-world robotics will require addressing noisy demonstrations, multimodal (vision–language) grounding, and reinforcement fine-tuning, which we leave as directions for future work.

## 6 Conclusion

We introduced TeNet, a text-to-network framework for instantiating compact, task-specific policies directly from natural language. By combining LLM-based text embeddings, trajectory grounding, and hypernetwork-based parameter generation, TeNet produces lightweight controllers that generalize across tasks without requiring demonstrations at inference time. Experiments on Mujoco and Meta-World benchmarks show that TeNet outperforms Prompt-DT in multi-task settings, achieves competitive performance in meta-learning, and supports control frequencies above 9 kHz. Together, these results highlight language-enabled hypernetworks as a promising direction for scalable, efficient, and deployable robot learning.

## Acknowledgements

This work was supported by the project *TeNet: Text-to-Network for Fast and Energy-Efficient Robot Control*, file number NGF.1609.241.015, under the research programme *National Growth Fund AiNed XS Europe 24-2*, financed by the Dutch Research Council (NWO).

## References

[Beck *et al.*, 2023] Jacob Beck, Matthew Thomas Jackson, Risto Vuorio, and Shimon Whiteson. Hypernetworks in meta-reinforcement learning. In *Conference on Robot Learning*, pages 1478–1487. PMLR, 2023.

[Brohan *et al.*, 2023] Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, and Others. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pages 287–318. PMLR, 2023.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and Others. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[Charakorn *et al.*, 2025] Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. Text-to-LoRA: Instant Transformer Adaption. *arXiv preprint arXiv:2506.06105*, 2025.

[Chen *et al.*, 2021] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[Chi *et al.*, 2025] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[Driess *et al.*, 2023] Danny Driess, Fei Xia, Mehdi S M Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, and Others. PaLM-E: an embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, pages 8469–8488, 2023.

[Gklezakos *et al.*, 2022] Dimitrios C Gklezakos, Rishi Jha, and Rajesh P N Rao. Hyper-universal policy approximation: Learning to generate actions from a single image using hypernets. *arXiv preprint arXiv:2207.03593*, 2022.

[Ha *et al.*, 2016] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

[He *et al.*, 2023] Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. *Advances in neural information processing systems*, 36:64896–64917, 2023.

[Hegde *et al.*, 2023] Shashank Hegde, Sumeet Batra, K R Zentner, and Gaurav Sukhatme. Generating behaviorally diverse policies with latent diffusion models. *Advances in Neural Information Processing Systems*, 36:7541–7554, 2023.

[Hegde *et al.*, 2024] Shashank Hegde, Satyajeet Das, Gautam Salhotra, and Gaurav S Sukhatme. Latent Weight Diffusion: Generating reactive policies instead of trajectories. *arXiv preprint arXiv:2410.14040*, 2024.

[Kim *et al.*, 2025] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, and Others. OpenVLA: An Open-Source Vision-Language-Action Model. In *Conference on Robot Learning*, pages 2679–2713. PMLR, 2025.

[Liang *et al.*, 2023] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as Policies: Language Model Programs for Embodied Control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[Liang *et al.*, 2024] Yongyuan Liang, Tingqiang Xu, Kaizhe Hu, Guangqi Jiang, Furong Huang, and Huazhe Xu. Make-an-agent: A generalizable policy network generator with behavior-prompted diffusion. *Advances in Neural Information Processing Systems*, 37:19288–19306, 2024.

[Lv *et al.*, 2024] Chuancheng Lv, Lei Li, Shitou Zhang, Gang Chen, Fanchao Qi, Ningyu Zhang, and Hai-Tao Zheng. HyperLoRA: Efficient Cross-task Generalization via Constrained Low-Rank Adapters Generation. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16376–16393, Miami, Florida, USA, nov 2024. Association for Computational Linguistics.

[Mahabadi *et al.*, 2021] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient Multi-task Fine-tuning for Transformers via Shared Hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2021.

[Ni *et al.*, 2023] Fei Ni, Jianye Hao, Yao Mu, Yifu Yuan, Yan Zheng, Bin Wang, and Zhixuan Liang. Metadiffuser: Diffusion model as conditional planner for offline meta-rl.

In *International Conference on Machine Learning*, pages 26087–26105. PMLR, 2023.

[Rana *et al.*, 2023] Krishan Rana, Andrew Melnik, and Niko Sünderhauf. Contrastive language, action, and state pre-training for robot learning. *arXiv preprint arXiv:2304.10782*, 2023.

[Ren *et al.*, 2025] Hanxiang Ren, Li Sun, Xulong Wang, Pei Zhou, Zewen Wu, Siyan Dong, Difan Zou, Youyi Zheng, and Yanchao Yang. Hypogen: Optimization-biased hypernetworks for generalizable policy generation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[Rezaei-Shoshtari *et al.*, 2023] Sahand Rezaei-Shoshtari, Charlotte Morissette, Francois R Hogan, Gregory Dudek, and David Meger. Hypernetworks for zero-shot transfer in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9579–9587, 2023.

[Tang *et al.*, 2023] Yujin Tang, Wenhao Yu, Jie Tan, Heiga Zen, Aleksandra Faust, and Tatsuya Harada. SayTap: Language to Quadrupedal Locomotion. In *Conference on Robot Learning*, pages 3556–3570. PMLR, 2023.

[Team *et al.*, 2024] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, and Others. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[Todorov *et al.*, 2012] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and Others. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, 2023.

[Venuto *et al.*, 2024] David Venuto, Mohammad Sami Nur Islam, Martin Klissarov, Doina Precup, Sherry Yang, and Ankit Anand. Code as Reward: Empowering Reinforcement Learning with VLMs. In *International Conference on Machine Learning*, pages 49368–49387. PMLR, 2024.

[Wang *et al.*, 2024] Zhi Wang, Li Zhang, Wenhao Wu, Yuanheng Zhu, Dongbin Zhao, and Chunlin Chen. Meta-DT: Offline meta-RL as conditional sequence modeling with world model disentanglement. *Advances in Neural Information Processing Systems*, 37:44845–44870, 2024.

[Xiong *et al.*, 2024] Zheng Xiong, Risto Vuorio, Jacob Beck, Matthieu Zimmer, Kun Shao, and Shimon Whiteson. Distilling morphology-conditioned hypernetworks for efficient universal morphology control. *arXiv preprint arXiv:2402.06570*, 2024.

[Xu *et al.*, 2022] Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pages 24631–24645. PMLR, 2022.

[Ye and Ren, 2021] Qinyuan Ye and Xiang Ren. Learning to Generate Task-Specific Adapters from Task Description. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 646–653, 2021.

[Yu *et al.*, 2020] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1094–1100. PMLR, 2020.

[Zitkovich *et al.*, 2023] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, and Others. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.