Video PreTraining (VPT): Learning t
Watching Unlabeled Online Vid

Bowen Baker[*†] bowen@openai.com
Ilge Akkaya[*†] ilge@openai.com
Peter Zhokhov[*†] peterz@openai.com
Jie Tang[*†] jietang@openai.com
Adrien Ecoffet[*†] adrien@openai.com
Brandon Houghton[*†] brandon@openai.com
Jeff Clune[*†‡] jclune@gmail.com

Abstract

Pretraining on noisy, internet-scale datasets has been heavily studied for training models with broad, general capabilities for text, ima modalities.[1–6] However, for many sequential decision domains su video games, and computer use, publicly available data does not co required to train behavioral priors in the same way. We extend the pretraining paradigm to sequential decision domains through se imitation learning wherein agents learn to act by watching online un Specifically, we show that with a small amount of labeled data w inverse dynamics model accurate enough to label a huge unlabeled s data — here, online videos of people playing Minecraft — from whi train a general behavioral prior. Despite using the native human in and keyboard at 20Hz), we show that this behavioral prior has r shot capabilities and that it can be fine-tuned, with both imitatio reinforcement learning, to hard-exploration tasks that are impossibl scratch via reinforcement learning. For many tasks our models e level performance, and we are the first to report computer agents diamond tools, which can take proficient humans upwards of 20 m environment actions) of gameplay to accomplish.

1 Introduction

Work in recent years has demonstrated the efficacy of pretraining large models[7] on noisy internet-scale datasets for use in downstream tasks in n computer vision.[5,6,8] For sequential decision domains (e.g. robotics, game usage) where agents must repeatedly act within an environment, a wealth o web; however, most of this data is in the form of *unlabeled* video (i.e. w at each frame), making it much less straightforward to train a behavioral than it is in e.g. natural language. In a few rare settings, such as Chess, G

[*]This was a large effort by a dedicated team. Each author made huge contribution time periods. All members were full time on the project for over six months. BB, I original VPT project team and were thus involved for even longer (over a year). Asid members, author order is random. It was also randomized between IA and PZ.

[†]OpenAI
[‡]University of British Columbia

already exist large datasets with action labels from various online platform
used for imitation learning.[9,10] When large labeled datasets do not exist
for training capable agents is reinforcement learning (RL),[11] which can be
expensive for hard-exploration problems.[12–18] Many virtual tasks, e.g. nav
Photoshop, booking flights, etc., can be very hard to learn with RL and do no
available sources of labeled data.[19,20] In this paper, we seek to extend th
large, general-purpose foundation models to sequential decision domains by
internet-scale unlabeled video datasets with a simple semi-supervised imitati
call this method Video PreTraining (VPT) and demonstrate its efficacy in th

Existing semi-supervised imitation learning methods aim to learn with few or
however, they generally rely on the policy's ability to explore the environm
making them susceptible to exploration bottlenecks.[21–25] Furthermore, mos
imitation learning work was tested in the relatively low data regime; because
more data (∼70k hours of unlabeled video), we hypothesize that we can ac
with a much simpler method, a trend that has proven true for pretraining i
as text.[1] In particular, given a large but unlabeled dataset, we propose gene
gathering a small amount of labeled data to train an inverse dynamics mo
the action taken at each timestep in a video. Behavioral cloning (BC) can
of data because the model must learn to infer intent and the distribution ove
only past observations. In contrast, the inverse dynamics modeling task
non-causal, meaning it can look at both past and future frames to infer ac
environment mechanics are far simpler than the breadth of human behavior th
the environment, suggesting that non-causal IDMs could require far less data
models. Using pseudo-labels generated from the IDM, we then train a model
of behavior in the previously unlabeled dataset with standard behavioral clon
not require any model rollouts and thus does not suffer from any potential
in the environment. Finally, we show we can fine-tune this model to down
behavioral cloning or reinforcement learning.

Figure 1
crafting
mouse a
menus a

We chose to test our method in Minecraft because (a) it is one
of the most actively played games in the world[26] and thus has
a wealth of commonly available video data online, (b) it is a
fairly open-ended sandbox game with an extremely wide variety
of potential things to do, build, and collect, making our results
more applicable to real-world applications such as computer us-
age, which also tends to be varied and open-ended, and (c) it
has already garnered interest by the RL community as a research
domain due to its complexity and correspondingly difficult ex-
ploration challenges.[27–31] In this work we use the native human
interface for Minecraft so that we can (1) most accurately model
the human behavior distribution and reduce domain shift between
video data and the environment, (2) make data collection easier by allowing c
play the game without modification, and (3) eliminate the need to hand-eng
for models to interact with the environment. This choice means that our m
per second and must use a mouse and keyboard interface to interact with h
smelting, trading, etc., including dragging items to specific slots or navigati
the mouse cursor (Fig. 1). Compared to prior work in Minecraft that uses
constructs crafting and attacking macros,[30,32–34] using the native human inter
the environment's exploration difficulty, making most simple tasks near in
scratch. Even the simple task of gathering a single wooden log while alread
consecutive attack actions with the human interface, meaning the chance for
succeed is $\frac{1}{2}^{60}$. While this paper shows results in Minecraft only, the VPT
could be applied to any domain.

In Section 4 we show that the VPT foundation model has nontrivial zero-sh
plishing tasks impossible to learn with RL alone, such as crafting planks a
requiring a human proficient in Minecraft a median of 50 seconds or ∼9
Through fine-tuning with behavioral cloning to smaller datasets that target
distributions, our agent is able to push even further into the technology t

(taking a human a median of 2.3 minutes or ∼2790 actions). Finally, fine-
the most dramatic improvements: our agent is able to craft diamond tools,
in Minecraft made even more challenging by using the native human inter
a proficient human a median upwards of 20 minutes or ∼24000 actions.
of this work are (1) we are the first to show promising results applying se
learning to extremely large, noisy, and freely available video datasets for sequ
(2) we show that such pretraining plus fine-tuning enables agents to solve ta
impossible to learn, (3) we show that labeled contractor data is far more
the VPT method than it would be by directly training a foundation model
source our contractor data, trained model weights, and Minecraft environm
into learning to act via semi-supervised imitation learning at scale.

2 Preliminaries and Related Work

Imitation learning methods[35–38] seek to construct a policy that accurately m
behavior in some dataset $\mathcal{D} = \{(o_i, a_i)\},\ i \in \{1 \dots N\}$ of action-observatio
out these policies in an environment, they must be *causal*, meaning they co
from the current timestep t and past timesteps only, i.e. $\pi \sim p(a_t|o_1 \dots o_t$
simplest when demonstrations are labeled with corresponding actions. Imita
has seen success in aerial vehicles,[39,40] self-driving cars,[41,42] board games,[9,

When labeled demonstrations are not available, standard behavioral cloning
there is a large body of work in imitating behavior from unlabeled demons
GAIL[23] constructs an adversarial objective incentivizing the trained poli
indistinguishable from those in the target dataset. Edwards et al.[45] propo
policy using unlabeled demonstrations and then map the learned latent act
a small amount of environment interaction. Peng et al.[46] first use motion-c
agent positions in videos and then train RL agents to match these waypoint
et al.[47] and Aytar et al.[48] task a RL agent to match waypoints; however, they
are embeddings from unsupervised feature learning models. Pathak et al.[4
goal conditioned policies to take actions that advance the current state towa
states expressed as high dimensional visual waypoints. Most similar to our o
simultaneously train (1) an inverse dynamics model (IDM),[51] which aims to
action between timesteps given observations of past and future timesteps, e.g
(2) a behavioral cloning (BC) model on trajectories of observations labeled
train the IDM is collected by rolling out the BC model in the target envi
models improve in tandem. However, at any point in training if there are sequ
the IDM performs poorly on, it requires that the BC model perform those
order for the IDM to improve and correctly label them. Therefore, if the BC
efficiently, it could severely slow down learning. In order to avoid this poten
simpler two-stage approach: we first train an IDM on a small number of labe
from human contractors (they play the game as would normally as we reco
mouse movements). Because human contractors reach most relevant parts o
hold the IDM fixed throughout BC training.

Compared to most previous work in semi-supervised imitation learning, we
more complex and open-ended environment of Minecraft. Minecraft is a
game that, due its popularity and wide variety of mechanics, has attracte
research.[27,28,30–34,52–60] A large body of work focuses on small, custom-
with tasks such as navigation,[53,60] block placing,[54,55] instruction follow
others.[28,31,57] Work operating in the massive, randomly generated environm
has included hill climbing,[52] automated curriculum learning[30] and, most c
experiments presented in Sec. 4.4, diamond mining.[27,32–34] However, to the
there is no published work that operates in the full, unmodified human actio
drag-and-drop inventory management and item crafting.

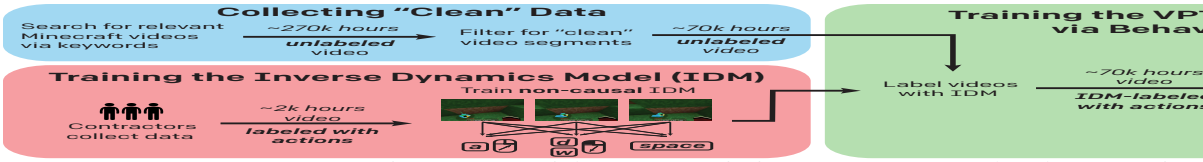

Figure 2: Video Pretraining (VPT) Method Overview

3 Methods

Inverse Dynamics Models (IDM) VPT, illustrated in Figure 2, requires amount of labeled contractor data with which to train an inverse dynamics which seeks to minimize the negative log-likelihood of an action at timestep observations $o_t \;:\; t \in [1 \dots T]$. In contrast to an imitation learning policy, the meaning its prediction for a_t can be a function of both past and *future events* the behavioral cloning objective of modeling the distribution of human inten we hypothesize that inverting environment dynamics is easier and more data Sec. 4.1 will show that the IDM objective is much easier to learn, and further that with very little labeled data (as few as 100 hours) we can train a fairly a can be used to label online videos, providing the large amount of data requir behavioral cloning. See appendices D and B for IDM training and data col

Data Filtering We gather a large dataset of Minecraft videos by search keywords (Appendix A). Online videos often (1) include overlaid artifac of the player's face, channel logos, watermarks, etc., (2) are collected fr a computer with different gameplay, or (3) are from different game mod only want "survival mode" where players start from scratch and must gath We call data "clean" if it does not contain visual artifacts and is from sur other data "unclean." With enough data, a large enough model, and enough model trained on both unclean and clean videos would likely still perform v environment. However, for simplicity and training compute efficiency, we ch segments of video (note that a video may contain both clean and unclean s training a model to filter out unclean segments using a small dataset (8800) online videos labeled by contractors as clean or unclean (Appendix A.2).

VPT Foundation Model We train a foundation model with standard beha mizing the negative log-likelihood of actions predicted by the IDM on cle trajectory of length T we minimize

$$\min_{\theta} \sum_{t \in [1 \dots T]} -\log \pi_\theta(a_t | o_1, \dots, o_t), \text{ where } a_t \sim p_{\text{IDM}}(a_t | o_1, \dots$$

As we will see in the following sections, this model exhibits nontrivial zero-s fine-tuned with both imitation learning and RL to perform even more comp

4 Results

4.1 Performance of the Inverse Dynamics Model

The IDM architecture is comprised primarily of a temporal convolution l processing stack, and residual unmasked attention layers, from which t predicts keypresses and mouse movements (see Appendix D for IDM a details). A key hypothesis behind our work is that IDMs can be trained with a of labeled data. While more data improves both mouse movement and keypr

Figure 3: (Left) IDM keypress accuracy and mouse movement R^2 (exp function of dataset size. **(Right)** IDM vs. behavioral cloning data efficiency

IDM trains on only 1962 hours of data (compared to the ∼70k hours of clear the internet) and achieves 90.6% keypress accuracy and a 0.97 R^2 for mous on a held-out validation set of contractor-labeled data (Figure 3 left).

Figure 3 (right) validates our hypothesis that IDMs are far more data efficien because inverting environment mechanics is far easier than modeling the enti behavior. The IDM is two orders of magnitude more data efficient than a B same data and improves more quickly with more data. This evidence support more effective to use contractor data within the VPT pipeline by training an foundation model from contractor data directly (Sections 4.5 and 4.6 provid

4.2 VPT Foundation Model Training and Zero-Shot Performance

Figure 4: (Left) Training and validation loss on the `web_clean` internet da labels, and loss on the main IDM contractor dataset, which has ground-tr distribution (see text). **(Right)** Amount a given item was collected per epis 60-minute survival episodes as a function of training epoch, shaded with t mean. Basic mining refers to collection of dirt, gravel, or sand (all materi without tools). Logs are obtained by repeatedly hitting trees for three secon RL agent to achieve as we show in Sec. 4.4. Planks can be crafted from l crafted from planks. Crafting requires using in-game crafting GUIs, and p median of 50 seconds (970 consecutive actions) to make a crafting table.

We now explore the emergent behavior learned by a behavioral cloning policy large, but noisy, internet dataset labeled with our IDM. To collect the un we searched for publicly available videos of Minecraft play with search te survival for beginners." These searches resulted in ∼270k hours of video, w "clean" video segments yielding an *unlabeled* dataset of ∼70k hours, which w (Appendix A has further details on data scraping and filtering). We then g for `web_clean` with our best IDM (Section 3) and then trained the VPT behavioral cloning. Preliminary experiments suggested that our model could of training and that a 0.5 billion parameter model was required to stay i regime[63] for that training duration (Appendix H), which took ∼9 days on 7

We evaluate our models by measuring validation loss (Fig. 4, left) and Minecraft environment. Unless otherwise noted, in all environment evaluatio standard survival mode game where they play for 60 minutes, i.e. 72000 con plot the mean and shade the standard error of the mean for various game st and collection rates (Fig. 4, right). The VPT foundation model quickly le to collect logs, a task we found near impossible for an RL agent to achieve interface (Sec. 4.4). It also learns to craft those logs into wooden planks an

to craft a crafting table, which are required to unlock most other technology human proficient in Minecraft approximately 50 seconds (970 consecutive a these behaviors are fairly complex in the native human action space, the VPT these items at a rate far below that of our proficient contractors, e.g. on aver 5.44 crafting tables in 60 minutes of play versus 0.19 for the foundation mod a non-negligible amount of wooden sticks, which are required to make wood flowers and crafts dyes from them; kills zombies that appear during the ni collects various berries and mushrooms and eats them; and finds game-gener to collect various rare items from chests. The model also learned to naviga and pillar jump, which involves the agent repeatedly jumping and quickly pla such that it climbs upward by making a pillar.[iv]

While training and validation loss decrease healthily over training (Fig. 4, lef dataset (which the VPT model does not train on) begins increasing after 7 could be out-of-distribution because our contractors may have a differen because there is some impactful visual domain shift compared to videos fr could have expected this would be predictive of declining evaluation perf notable game statistics from the VPT foundation model rollouts (Figure training, and in the next section we show that BC fine-tuning performance co VPT foundation model trains. We provide more insight into this curious phen

4.3 Fine-Tuning with Behavioral Cloning

Foundation models are designed to have a broad behavior profile and be ge wide variety of tasks. To incorporate new knowledge or allow them to speci distribution, it is common practice to fine-tune these models to smaller, mor VPT foundation model trained on the broad `web_c1ean` dataset had nontrivia it was able to craft a crafting table yet unable to go past this in the tech study into BC fine-tuning, we attempt to improve the VPT foundation m and craft these "early game" items by fine-tuning to two narrower datase behavior within the first few minutes of players starting in a fresh wor `contractor_house`, contractors have 10 minutes to build a basic house fro wood, sand, and dirt. Collecting contractor data can be difficult and expensiv dataset `earlygame_keyword` by searching for videos online with descriptio such as "new world", "let's play episode 1", etc.; this is a subset of `web_c1` the IDM. See Appendix B.4 and A.3 for full descriptions of both datasets.

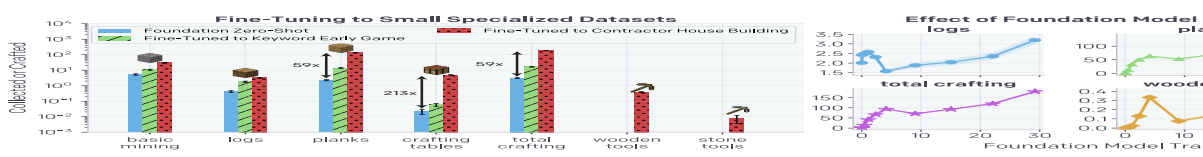

Figure 5: **(Left)** Collection and crafting rates for three policies: the dation model, and the VPT foundation model BC fine-tuned to the e `contractor_house` datasets. BC fine-tuning to either dataset improves per the `contractor_house` dataset) yielding wooden and stone tools. Proficien a median of 1.2 minutes (1390 actions) to construct wooden tools and 2.3 to construct stone tools. **(Right)** Collection and crafting rates for VPT foun throughout training *after* they are BC fine-tuned to the `contractor_hou` crafting-related behaviors increase throughout foundation model training. task terms (logs, planks, crafting tables, and total crafting).

[iv] Sample videos: https://www.youtube.com/playlist?list=PLNAOIb_agjf3U3rSv

Fine-tuning to `early_game_keyword` results in a large boost compared to t
model: 2.5x more crafting tables, 6.1x more planks, 4.3x more logs, and 5.
(Fig. 5). However, when fine-tuning to this dataset we did not see any
only a refinement of existing skills. We saw an even bigger improvement
`contractor_house` dataset: 213x more crafting tables, 59x more wooden
and 59x more crafting over all. In addition, we saw the emergence of crafti
requires placing a crafting table on the ground, opening it to reveal a new cra
using it to craft wooden tools. This entire sequence takes a proficient human
minutes (1390 consecutive actions) to accomplish. The model goes further a
which requires a wooden pickaxe to mine, and crafts stone tools, requiring
table; this takes a proficient human player a median of 2.3 minutes (2790 c
also saw this model more frequently raiding villages that randomly spaw
animals for food, in addition to many behaviors we saw performed by the fo

Despite the foundation model's zero-shot rollout performance plateauing
right), fine-tuning performance *does* continue to increase throughout fou
(Fig. 5, right). Additionally, there is a stark difference in performance when
fine-tuning from the VPT foundation model (Fig. 5 right, comparing the lef

4.4 Fine-Tuning with Reinforcement Learning

Figure 6: Typical sequence of items for obtaining a diamond pickaxe. Below
time and number of actions contractors required to obtain that item and the p
that got the item within 10 minutes. The median time to obtain a diamond pic
that it is $>$ 20m) because contractors obtained this item in less than 50% of

To demonstrate the efficacy of RL fine-tuning, we chose the challenging goa
pickaxe within 10 minutes starting from a fresh Minecraft survival world. Do
a sequence of difficult-to-obtain items that require complex skills like mining
crafting with and without a crafting table, tool use, operating a furnace, a
depths, where many hazards like enemies and lava exist (Fig. 6). Adding to
can be easily lost by dropping items, destroying items, or dying. Obtaining a
often than not takes a proficient human over 20 minutes (24,000 actions).

Agents are rewarded for each item obtained in the sequence, with lower rewa
be collected in bulk and higher rewards for items near the end of the sequenc
with the phasic policy gradient[64] RL algorithm for $\sim$1.3 million episodes (rou
Episodes last for 10 minutes. See Appendix G.1 for reward function and RL
computational constraints, RL experiments use a $\sim$ 248 million parameter V

A major problem when fine-tuning with RL is catastrophic forgetting[65,66] be
skills can be lost before their value is realized. For instance, while our VPT
exhibits the entire sequence of behaviors required to smelt iron zero-shot, it
players smelting with furnaces. It therefore may have some latent ability to s
prerequisites to do so have been performed. To combat the catastrophic fo
such that they can continually improve exploration throughout RL fine-tun
Kullback-Leibler (KL) divergence loss between the RL model and the froze

Training from a randomly initialized policy fails to achieve almost *any* rew
hard an exploration challenge the diamond pickaxe task is for RL in the na
(Fig. 7a). The model never learns to reliably collect logs, typically the first of
a diamond pickaxe (Fig. 7b). RL fine-tuning from the VPT foundation m
better (Fig. 7a), learning everything up to mining iron ore and crafting furna
this agent fails at smelting an iron ingot, the next item required to get further

[v]Sample Videos: https://www.youtube.com/playlist?list=PLNAOIb_agjf2yDSs4

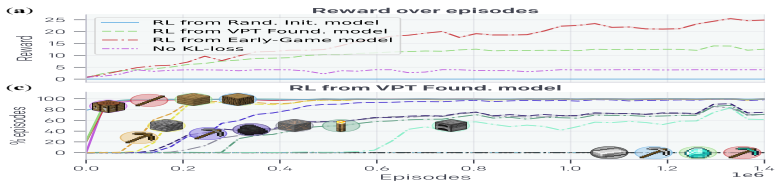

Figure 7: RL Fine-tuning results. **(a)** RL from a randomly initialized m
any reward, RL fine-tuning from the VPT foundation model performs sub
reward near 13, and RL fine-tuning from the early-game model performs b
When training the early-game model without a KL loss to the original polic
stalls after 100,000 episodes, suggesting that the skills necessary to make fu
catastrophically forgotten. **(b)** RL from a randomly initialized model occasi
breaking leaves (an easy but inefficient method of getting sticks that does no
and never learns to reliably collect logs. **(c)** RL fine-tuning from the VPT F
everything in the curriculum up to iron ore and making furnaces, but fails to l
smelt iron ingots. **(d)** RL fine-tuning from the early-game model learns to ob
items in the sequence towards a diamond pickaxe and crafts a diamond picka

because the zero-shot probability that the VPT foundation model smelts an ir
when given the prerequisite materials.

Results further improve by first BC fine-tuning the VPT Foun
`earlygame_keyword` dataset (the *early-game model*, Sec. 4.3) and the
(Fig. 7a), which in preliminary experiments we found to perform better
`contractor_house` followed by fine-tuning with RL (Appendix G.2). T
(pretraining, BC fine-tuning, and then RL fine-tuning) succeeds in learning
it achieves over 80% reliability on iron pickaxes, almost 20% reliability on
2.5% reliability on obtaining a diamond pickaxe (Fig. 7d). For compariso
the objective of obtaining a diamond pickaxe collect these items in 57%, 15
respectively, meaning our model is human-level for crafting iron pickaxe
Others have managed to obtain diamonds with $\sim 0.1\%$ reliability in 15 minut
simplified action space designed to ease exploration. To the best of our knowl
report non-zero success rates on crafting a diamond pickaxe. Qualitative
useful skills for diamond mining, such as efficient mining patterns, cave e
previously placed objects like crafting tables, and advanced techniques like
as fuel when moving on to iron tools.[vi]

Finally, we validated the importance of the KL loss to the pretrained mode
The treatment without a KL loss obtains only items early in the sequence (
crafting tables) limiting its reward (Fig. 7a). This failure to progress furth
likely because, while the initial skills of chopping logs and crafting planks ar
subsequent skills like crafting a wooden pickaxe are lost due to catastrophic

4.5 Data Scaling Properties of the Foundation Model

In this section we validate a core hypothesis behind this work: that it is f
labeled contractor data to train an IDM within the VPT method than it i
foundation model from that same small contractor dataset. If we could cl
contractor dataset of a similar order of magnitude as `web_clean`, then this
however, collecting that scale of data would have cost millions of dolla
foundation models trained on increasing orders of magnitude of data from 1
`web_clean` dataset. Foundation models trained up to and including 1k hours

[vi] Videos found at https://www.youtube.com/playlist?list=PLNAOIb_agjf3e_UKv

Figure 8: **(Left)** Zero-shot rollout performance of foundation models train of data. Models to the left of the dashed black line (points $\leq$ 1k hours) we data (ground-truth labels), and models to the right were trained on IDM of `web_clean`. Due to compute limitations, this analysis was performed v parameter) models except for the final point, which is the 0.5 billion para model. **(Right)** The corresponding performance of each model *after* BC fi the `contractor_house` dataset.

contractor data, and those trained on 5k hours and above are trained on subse does not contain any IDM contractor data. Scaling training data increases log crafting capabilities. The zero-shot model only begins to start crafting craf hours of training data. When fine-tuning each foundation model to `contrac` crafting rates for crafting tables and wooden tools increase by orders of ma entire $\sim$70k hour `web_clean` dataset. We furthermore only see the emergenc at the largest data scale.

4.6 Effect of Inverse Dynamics Model Quality on Behavioral Cloning

This section investigates how downstream BC performance is affected by IDM quality. We train IDMs on increasingly larger datasets and use each to independently label the `early_game_keyword` dataset (this smaller dataset was chosen due to a limited compute budget). We then train a BC model from scratch on each dataset and report game statistics for each model as a function of IDM contractor dataset size (Fig. 9).

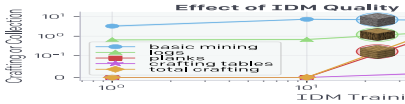

Figure 9: Zero-shot perf trained from scratch on th dataset labeled with IDM increasing amounts of co

IDMs trained on at least 10 hours of data are required for any crafting, and the crafting rate increases quickly up until 100 hours of data, after which there are few to no g likely due to noise. Similarly, crafting tables are only crafted after 50 or more again gains plateau after 100 hours. While in all previous experiments we u on 1962 hours of data, these results suggest we could reduce that number to

5 Discussion and Conclusion

The results presented in this paper help pave the path to utilizing the wealth web for sequential decision domains. Compared to generative video modelin that would only yield representational priors, VPT offers the exciting possib *to act* during pretraining and using these learned behavioral priors as extrem priors for RL. VPT could even be a better general representation learning downstream task is not learning to act in that domain—for example, fine-t happening in a video—because arguably the most important information in a present in features trained to correctly predict the distribution over future h this intriguing direction to future work.

Future work could improve results with more data (we estimate we could larger, better-tuned models. Furthermore, all the models in this work condit only; we cannot ask the model to perform specific tasks. Appendix I presents on conditioning our models on closed captions (text transcripts of speech i

become weakly steerable; we believe this a rich direction for future resea
consistently correlated with downstream evaluation metrics (Sec. 4.2), wh
slow and hard-won. Another fruitful future direction would be to investigate
various training metrics and downstream evaluations. Finally, while we do
negative societal impacts from the models trained in this work, as VPT impro
domains it will be important to assess and mitigate harms that emerge with o
on internet datasets, such as emulating inappropriate behavior.[67]

In conclusion, VPT extends the paradigm of training large and general purpos
freely available internet-scale data to sequential decision domains. Our mod
zero-shot behavior and, when fine-tuned with RL, achieved an unprecede
diamond pickaxe in Minecraft (all the more difficult given the human interf
that contractor data is far better used within the VPT pipeline than to train a fo
and that only a small amount of contractor data (about $2000 USD) was req
amounts of unlabeled online data for use in BC. Finally, learning with the hu
interface is highly general and allows losslessly modeling the entire distribu
While we only experiment in Minecraft, we believe that VPT provides a ge
behavioral priors in hard, yet generic, action spaces in any domain that has
available unlabeled data, such as computer usage.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D K
Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et a
few-shot learners. *Advances in neural information processing systems*

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanov
deep bidirectional transformers for language understanding. *arXiv pre*
2018.

[3] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi
Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly op
approach. *arXiv preprint arXiv:1907.11692*, 2019.

[4] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan N
Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer
text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[5] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Ma
Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limi
pretraining. In *Proceedings of the European conference on compute*
181–196, 2018.

[6] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel
Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Lea
models from natural language supervision. In *International Conferenc*
pages 8748–8763. PMLR, 2021.

[7] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Sim
Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma
opportunities and risks of foundation models. *arXiv preprint arXiv:21*

[8] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. S
ers. *CoRR*, abs/2106.04560, 2021. URL https://arxiv.org/abs/

[9] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre,
che, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, M
tering the game of go with deep neural networks and tree search. *Natu*
2016.

[10] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathi
young Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Geo
level in starcraft ii using multi-agent reinforcement learning. *Nature*, 57

[11] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An i* 2018.

[12] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Prze Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*,

[13] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn and Igor Mordatch. Emergent tool use from multi-agent autocur *arXiv:1909.07528*, 2019.

[14] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Gu Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avi Human-level performance in 3d multiplayer games with population-ba ing. *Science*, 364(6443):859–865, 2019.

[15] Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo S skyi, Zhaohan Daniel Guo, and Charles Blundell. Agent57: Outperf benchmark. In *International Conference on Machine Learning*, pages

[16] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, D Munos. Unifying count-based exploration and intrinsic motivatio *information processing systems*, 29, 2016.

[17] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

[18] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and then explore. *Nature*, 590(7847):580–586, 2021.

[19] Peter C Humphreys, David Raposo, Toby Pohlen, Gregory Thornton, Ra Muldal, Josh Abramson, Petko Georgiev, Alex Goldin, Adam Santo approach for learning to control computers. *arXiv preprint arXiv:2201*

[20] Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and bits: An open-domain platform for web-based agents. In Doina Pre editors, *Proceedings of the 34th International Conference on Machine Proceedings of Machine Learning Research*, pages 3135–3144. PMLR https://proceedings.mlr.press/v70/shi17a.html.

[21] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforc volume 1, page 2, 2000.

[22] Faraz Torabi, Garrett Warnell, and Peter Stone. Recent advances in observation. *arXiv preprint arXiv:1905.13566*, 2019.

[23] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learn *information processing systems*, 29, 2016.

[24] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning f *preprint arXiv:1805.01954*, 2018.

[25] YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imi Learning to imitate behaviors from raw video via context translation. In *Conference on Robotics and Automation (ICRA)*, pages 1118–1125. IE

[26] Twinfinite Staff. Most played games in 2021, ranke rent players. *Twinfinite*. URL https://twinf most-played-games-in-2020-ranked-by-peak-concurrent-p

[27] William H Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, C Veloso, and Ruslan Salakhutdinov. Minerl: A large-scale dataset of m *arXiv preprint arXiv:1907.13440*, 2019.

[28] Chen Tessler, Shahar Givony, Tom Zahavy, Daniel Mankowitz, and
hierarchical approach to lifelong learning in minecraft. In *Proceedings*
on Artificial Intelligence, volume 31, 2017.

[29] Christian Scheller, Yanick Schraner, and Manfred Vogel. Sample
learning through learning from demonstrations in minecraft. In *NeurIP*
Demonstration Track, pages 67–76. PMLR, 2020.

[30] Ingmar Kanitscheider, Joost Huizinga, David Farhi, William Hebgen G
Raul Sampedro, Peter Zhokhov, Bowen Baker, Adrien Ecoffet, Jie
curriculum learning in a complex, visual, hard-exploration domain: M
arXiv:2106.14876, 2021.

[31] Junhyuk Oh, Valliappa Chockalingam, Honglak Lee, et al. Control of m
and action in minecraft. In *International Conference on Machine Lea*
PMLR, 2016.

[32] Vihang P Patil, Markus Hofmarcher, Marius-Constantin Dinu, Matthias
Johannes Brandstetter, Jose A Arjona-Medina, and Sepp Hochreiter.
from few demonstrations by reward redistribution. *arXiv preprint arX*

[33] Alexey Skrynnik, Aleksey Staroverov, Ermek Aitygulov, Kirill Aksen
Aleksandr I Panov. Forgetful experience replay in hierarchical reinf
demonstrations. *arXiv preprint arXiv:2006.09939*, 2020.

[34] Zichuan Lin, Junyou Li, Jianing Shi, Deheng Ye, Qiang Fu, and
Playing minecraft with sample-efficient hierarchical reinforcement l
arXiv:2112.04907, 2021.

[35] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neura
neural information processing systems, 1, 1988.

[36] Stefan Schaal. Is imitation learning the route to humanoid robots? *Tren*
3(6):233–242, 1999.

[37] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Brown
learning from demonstration. *Robotics and autonomous systems*, 57(5

[38] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina J
A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(

[39] Claude Sammut, Scott Hurst, Dana Kedzier, and Donald Michie. Lea
Learning Proceedings 1992, pages 385–393. Elsevier, 1992.

[40] Alessandro Giusti, Jérôme Guzzi, Dan C Cireşan, Fang-Lin He, Ju
Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, G
machine learning approach to visual perception of forest trails for mobi
and Automation Letters, 1(2):661–667, 2015.

[41] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Fir
Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang,
for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[42] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, a
End-to-end driving via conditional imitation learning. In *2018 IEEE I*
on robotics and automation (ICRA), pages 4693–4700. IEEE, 2018.

[43] Rémi Coulom. Computing "elo ratings" of move patterns in the game
(4):198–208, 2007.

[44] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul
John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning f
Proceedings of the AAAI Conference on Artificial Intelligence, volume

[45] Ashley Edwards, Himanshu Sahni, Yannick Schroecker, and Charles
policies from observation. In *International conference on machine lea*
PMLR, 2019.

[46] Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, a
Reinforcement learning of physical skills from videos. *ACM Transacti*
37(6):1–14, 2018.

[47] Feryal Behbahani, Kyriacos Shiarlis, Xi Chen, Vitaly Kurin, Sudhanshu
Joao Gomes, Supratik Paul, Frans A Oliehoek, Joao Messias, et al. Lear
in the wild. In *2019 International Conference on Robotics and Au*
775–781. IEEE, 2019.

[48] Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wa
itas. Playing hard exploration games by watching youtube. *Advance*
processing systems, 31, 2018.

[49] Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal,
Evan Shelhamer, Jitendra Malik, Alexei A. Efros, and Trevor Darrell. Z
In *ICLR*, 2018.

[50] Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Ji
Levine. Combining self-supervised learning and imitation for vision-b
pages 2146–2153, 05 2017. doi: 10.1109/ICRA.2017.7989247.

[51] Duy Nguyen-Tuong, Jan Peters, Matthias Seeger, and Bernhard Schö
dynamics: a comparison. In *European symposium on artificial neural n*
2008.

[52] David Abel, Alekh Agarwal, Fernando Diaz, Akshay Krishnamurthy,
Exploratory gradient boosting for reinforcement learning in complex
arXiv:1603.04119, 2016.

[53] Dilip Arumugam, Jun Ki Lee, Sophie Saskin, and Michael L Littma
learning from policy-dependent human feedback. *arXiv preprint arXi*

[54] Alexander Trott, Stephan Zheng, Caiming Xiong, and Richard Soc
tance: Solving sparse reward tasks using self-balancing shaped rewar
Information Processing Systems, 32, 2019.

[55] Stephan Alaniz. Deep reinforcement learning with model learning and
in minecraft. *arXiv preprint arXiv:1803.08456*, 2018.

[56] Hiroto Udagawa, Tarun Narasimhan, and Shim-Young Lee. Fighting zo
deep reinforcement learning. Technical report, Technical report, Te
University, 2016.

[57] Tianmin Shu, Caiming Xiong, and Richard Socher. Hierarchical and in
tion in multi-task reinforcement learning. *arXiv preprint arXiv:1712.0*

[58] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. Zero
with multi-task deep reinforcement learning. In *International Conferen*
pages 2661–2670. PMLR, 2017.

[59] Zhengxiang Shi, Yue Feng, and Aldo Lipani. Learning to execute or as
arXiv preprint arXiv:2204.08373, 2022.

[60] Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Tea
learning. *IEEE transactions on neural networks and learning systems*,

[61] Robert George Douglas Steel, James Hiram Torrie, et al. Principles and
Principles and procedures of statistics., 1960.

[62] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep resi
recognition. In *Proceedings of the IEEE conference on computer vision*
pages 770–778, 2016.

[63] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjam
Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling la
models. *arXiv preprint arXiv:2001.08361*, 2020.

[64] Karl W Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Ph
Marina Meila and Tong Zhang, editors, *Proceedings of the 38th Inter*
Machine Learning, volume 139 of *Proceedings of Machine Learning*
2027. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.p
html.

[65] Dhireesha Kudithipudi, Mario Aguilar-Simon, Jonathan Babb, Max
Blackiston, Josh Bongard, Andrew P Brna, Suraj Chakravarthi Raja, N
et al. Biological underpinnings for lifelong learning machines. *Nature*
(3):196–210, 2022.

[66] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness,
Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka G
Overcoming catastrophic forgetting in neural networks. *Proceedings of*
sciences, 114(13):3521–3526, 2017.

[67] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shm
the dangers of stochastic parrots: Can language models be too big??
2021 ACM Conference on Fairness, Accountability, and Transparency

[68] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent M
Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincen
learn: Machine learning in python. *the Journal of machine Learning r*
2011.

[69] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion
Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advanc*
processing systems, 30, 2017.

[70] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer norm
arXiv:1607.06450, 2016.

[71] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Mu
In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[72] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic opti
arXiv:1412.6980, 2014.

[73] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bra
Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alba
Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejan
Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: A
performance deep learning library. In H. Wallach, H. Larochelle, A. B
Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information*
pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers
9015-pytorch-an-imperative-style-high-performance-dee
pdf.

[74] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun C
Yoshua Bengio. Identifying and attacking the saddle point problem in
convex optimization. *Advances in neural information processing syste*

[75] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How tra
deep neural networks? *Advances in neural information processing sys*

[76] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, an
Transformer-xl: Attentive language models beyond a fixed-length c
arXiv:1901.02860, 2019.

[77] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and
policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 201

[78] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, an
dimensional continuous control using generalized advantage estim
arXiv:1506.02438, 2015.

[79] Ronald J Williams. Simple statistical gradient-following algorithms for
ment learning. *Machine learning*, 8(3):229–256, 1992.

[80] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Gr
Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous me
ment learning. In *International conference on machine learning*, pa
2016.

[81] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rach
Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremb
replay. *Advances in neural information processing systems*, 30, 2017.

[82] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Univ
proximators. In *International conference on machine learning*, pag
2015.

[83] Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarin
Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Micha
ended learning leads to generally capable agents. *arXiv preprint arXi*

[84] Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob Foerster,
Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey
ing informed by natural language. In *Proceedings of the Twenty-Eig
Conference on Artificial Intelligence, IJCAI-19*, pages 6309–6317. Int
ences on Artificial Intelligence Organization, 7 2019. doi: 10.2496
https://doi.org/10.24963/ijcai.2019/880.

[85] DeepMind Interactive Agents Team, Josh Abramson, Arun Ahuja, A
Carnevale, Mary Cassin, Felix Fischer, Petko Georgiev, Alex Goldin, Ti
multimodal interactive agents with imitation and self-supervised le
arXiv:2112.03763, 2021.

[86] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and M
text-conditional image generation with clip latents. *arXiv preprint arX*

[87] Dong Yu and Li Deng. *Automatic speech recognition*, volume 1. Spri

[88] Daulet Nurmanbetov. rpunct, May 25 2021. URL https://github.c
accessed 2022-04-22.

[89] Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Mich
Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. Te
by contrastive pre-training. *arXiv preprint arXiv:2201.10005*, 2022.

Acknowledgements

We thank the following people for helpful discussions and support: Bob
Joel Lehman, Ilya Sutskever, Wojciech Zaremba, Ingmar Kanitscheider, Da
Jonathan Gordon, and the OpenAI supercomputing team, especially Chris
and Christopher Berner.

Supplementary Information

A Collecting Internet Data

A.1 Initial Unclean Dataset Curation

Our goal was to curate a video dataset of Minecraft gameplay from the surviv
ally, we prefer the data come from game modes as close as possible to our
meaning preferably coming from Minecraft version 1.16, being on a compu
and keyboard vs. video game controllers with keypads and other buttons), b
player, and having the default look of the game (vs. modifications that al
make it look realistic). To try to accomplish these goals, we collect a dataset
searches of publicly available videos on the internet. A list of search queri
Table 1.

minecraft survival longplay
minecraft gameplay no webcam
minecraft gameplay survival mode
minecraft survival tutorial
minecraft survival guide
minecraft survival let's play
minecraft survival for beginners
minecraft beginners guide
ultimate minecraft starter guide
minecraft survival guide 1.16
minecraft how to start a new survival world
minecraft survival fresh start
minecraft survival let's play episode 1
let's play minecraft episode 1
minecraft survival 101
minecraft survival learning to play
how to play minecraft survival
how to play minecraft
minecraft survival basic
minecraft survival for noobs
minecraft survival for dummies
how to play minecraft for beginners
minecraft survival tutorial series
minecraft survival new world
minecraft survival a new beginning
minecraft survival episodio 1
minecraft survival епизод 1
minecraft survival 1. bölüm
i made a new minecraft survival world

Table 1: Search terms used for generating the initial web d

For videos that have metadata available, we perform an additional step of
to eliminate videos that do not fit our target distribution. In this step, we lo
keywords in the video title and description and reject videos that contain th
keywords we use are: {ps3, ps4, ps5, xbox 360, playstation, timelapse, m
pocket edition, skyblock, realistic minecraft, how to install, how to download
This process yielded us $\sim$270k hours of unlabeled data, which we filter down
as described in the next section.

A.2 Training a Model to Filter out Unclean Video Segments

We restrict the scope of this work to the Minecraft Survival game mode
training dataset to clips that are obtained from this mode that are relatively f

To do so, we asked contractors to label a set of random video frames (images
(N=8800). These images were from a random subset of the videos we collect
of the project (Section A.1).

A.2.1 Label Collection

We asked 5 workers on Amazon Mechanical Turk (mTurk) that we selected wi
task to label random screen capture images to be used in training the clas
interface that the workers saw on mTurk is given in Figure 10.

We asked workers to label videos as being in one of the following three cate
visual examples of each class):

1. `Minecraft Survival Mode – No Artifacts:` Video frames (
to the Minecraft Survival game mode that do not contain any non-g
subscribe buttons, channel logos, advertisements, picture-in-pictur
2. `Minecraft Survival Mode – with Artifacts:` Video fr
Minecraft Survival game mode that include such visual artifacts.
3. `None of the Above:` Video frames (images) that are not from
game mode, including those from other Minecraft game modes s
even other games/topics entirely.

The full set of instructions workers received are as follows (note that we also
examples from each category in the worker instructions, similar to the sa
Figure 11):

> Please help us identify screenshots that belong only to the survival mode i
> else (Minecraft creative mode, other games, music videos, etc.) should be
> above. Survival mode is identified by the info at the bottom of the screen:
> - a health bar (row of hearts)
> - a hunger bar (row of chicken drumsticks)
> - a bar showing items held
>
> **Survival Mode**
> Valid survival mode videos have health/hunger bars and an item hotbar at t
> **Creative Mode**
> Creative mode only has an item hotbar and should be classified as `None o`
>
> **Label Descriptions**
> - `Minecraft Survival Mode – No Artifacts:` These images
> from the Minecraft survival mode gameplay without any noticeab
> - `Minecraft Survival Mode – with Artifacts:` These imag
> mode screenshots, but with some added artifacts. Typical artif
> overlays (a logo/brand), text annotations, a picture-in-picture of t
> - `None of the Above:` Use this category when the image is not a
> screenshot. It may be a non-Minecraft frame or from a different ga
> game modes such as the creative mode, the health/hunger bars
> image, the item hotbar may or may not be still present.

In total, we spent $319.96 on human labeling experiments on mTurk, of whi
paid to workers. The remaining amount was spent towards Amazon platf
received $0.01 per labeled image, at an hourly compensation of $7.20 (based
time of 5 seconds/image — in our internal sample run of the same task, we fo
time to be $\leq$ 3 seconds).

Since we perform rigorous keyword and metadata based filtering of videos (a
which we served sample images to be labeled, serving offensive content to

low risk and no such images were detected during our manual checks. V
during our experiment, and the workers were fully anonymized via the mTur
personally identifiable information (PII) was collected.

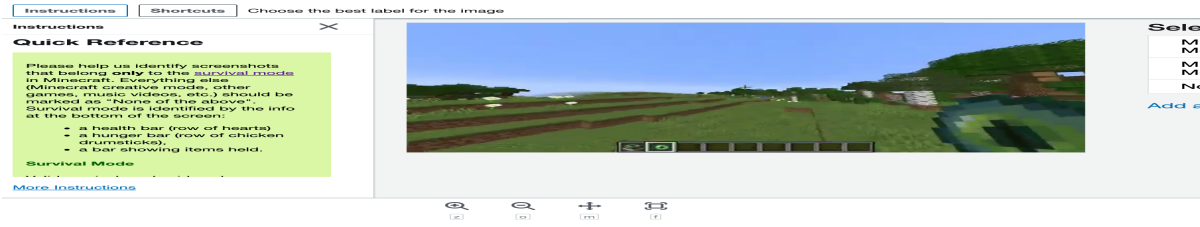

Figure 10: Amazon Mechanical Turk worker interface showing an exa

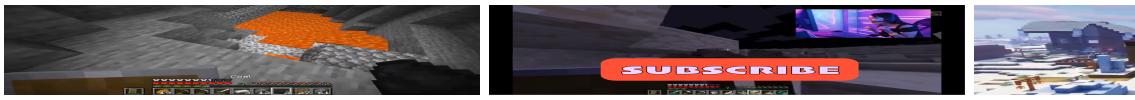

Figure 11: **(Left)** Sample image for Class 1: `Minecraft Survival M` **(Middle)** Sample image for Class 2: `Minecraft Survival Mode – wit` contains annotations and picture-in-picture of the narrator. **(Right)** Sample `of the Above` – Image is missing the hotbar as well as health and armor ba not captured during survival mode gameplay

A.2.2 SVM Training

With the image labels collected as described in the previous section, we train video segments that consist of frames from the `Minecraft Survival M` category. Given a set of labeled images, we obtain embeddings for each im ResNet CLIP Model.[6] This is a ResNet-based CLIP model that is scaled up 64x the compute of a ResNet-50. We then train a Support Vector Machine kernel to obtain a frame classifier. We use the Scikit-learn[68] SVM implement configuration given in Table 2.

Finally, we apply the classifier to frames of raw video sequences at a rate of 3 for videos that consist of at least 80% "clean" frames at this stage (Classes `Mode – with Artifacts` and `None of the Above` are both considered we apply a median filter (with a kernel size of 7) to the labels and segmen "clean" segments that are at least 5s in duration. The result of this is our fin

A.3 `early_game` Dataset

The `early_game` dataset is a ∼3000 hour subset of `web_clean` targeted at behavior, i.e. instances where players start in a fresh world with no items. text that accompanies the videos in `web_clean` and determine whether any expressions match:

CLIP Model Specification	RN50x64 (see tex	
CLIP Input Image Resolution	448x448x3	
CLIP Embedding Feature Length	1024	
SVM Parameters	Kernel	rbf
	C	20
	Gamma	scale
Sample Size	Class 1	2200
	Class 2	2200
	Class 3	4400

Table 2: Feature Extraction Details and SVM Configuration. The param
implementation in Scikit-learn[68].

- (ep|episode|eps|day|session|sesh|chapter|chap
  .|series|part|parte|pt|round|day|tập|bölüm|episodio|епизод|эпизод
)*(\.1|#1|1|\.01|#01|01|one[^0-9]|$)
- start
- beginning
- (new|fresh|clean).*(world|game|play)
- from scratch

From this set of videos, we take only the first 5 minutes of each video.

B Contractor Data

B.1 Recording Contractor Play

Our contractors use a custom Minecraft recorder that we built that records thei
feeds as they play. The recorder is implemented using the MCP-Reborn (gith
Reborn) modding package. To ensure that the recorder environment is
the Minecraft environment used for RL rollouts and evaluations (Append
underlying game engine for both. The recorder is a Java app that runs in
constant resolution of 1280x760. Brightness is set to 0 (the "gloomy" setti
is the default setting. Other graphics settings (field of view, GUI scale) are
in the Minecraft environment (C.1); we explicitly prevented users from cha
Unlike the environment, the recorder allows all keyboard key presses and co
binned) mouse actions. On every game step (or "tick") the frame buffer u
window is downsized to 640x360 and written into a video file. In-game a
separate JSONL file (a text file where each line is a JSON-formatted stri
chunked into 5 minute clips: after each 5 minute segment of contractor
automatically uploads the video file, the JSONL file with actions, as well
To ensure that contractors cannot corrupt each other's data, we provided e
individual cloud bucket, as well as with credentials giving write access only to
also included adjective-adjective-noun names (e.g. grumpy-amethyst-chipm
namegenerator python package to ensure contractor anonymity when we

B.2 Contractor Contract

We recruited contractors by posting the following offer on the UpWork free

> "We are collecting data for training AI models in Minecraft. You'l
> java, download the modified version of Minecraft (that collects an
> play data), and play Minecraft survival mode! Paid per hour of g
> experience in Minecraft not necessary. We do not collect any data t
> to Minecraft from your computer."

We had the applications open for a day, and then randomly selected 10 applic
contractors. Later in the project, as we needed more data and as some contra
their contracts, we added more applicants from the original pool as well as re
working contractors. The contractors were paid $20 per hour (minus Up
applicable taxes). All of the results presented in this paper are based on ab
(including data recorded to gather statistics of human play that was not used
us around $90,000. Over the course of the project, we collected some dat
bugs in the recorder and for some ideas we ultimately did not pursue. In tota
for contractor compensation over the course of the project. However, as we
could likely obtain most of our results with an IDM trained using only $200
foundation VPT model, BC fine-tuning to the `early_game_keyword` dataset
results. Collecting the `contractor_house` dataset cost about $8000. Be
trained on about 2000 hours of contractor data, the actual cost of contractor d
around $40,000.

In early stages of the project, we were planning to use contractor data solely fo
the IDM. As such, no specific tasks were given, other than "play the survival
you normally would." Later in the project, we requested that contractors p
Minecraft, such as:

- Collect as many units of wood as possible, using only wooden or s
- Start a new world every 30 minutes of game play
- Build a basic house in 10 minutes using only dirt, wood, sand, and
  tools (`contractor_house`, more details below in Appendix B.4).
- Starting from a new world and an empty inventory, find resource
  pickaxe in 20 minutes (`obtain_diamond_pickaxe`). This data
  statistics for how long it takes humans on average to complete thi
  required to complete it) when obtaining a diamond pickaxe is their

Since we only recorded in-game events and videos, the data does not includ
information. That being said, the contractors could theoretically use Minecra
to generate personally identifiable information and/or offensive content (
blocks to write their name or offensive messages, then finding a spot from w
be visible). In practice, we have not seen any attempts to do so in the co
watched. Of course, we train our BC models on videos from the internet of p
and if such behavior is in those videos our model could also potentially lear
such behavior is rare enough that our model would not be likely to reproduc

B.3 Data for the Inverse Dynamics Model.

Since the IDM's task is to infer actions given the video, any labelled data
training. In practice, we included general gameplay as well as the `treec`
in the previous section, which amounted to a total of 1962 hours. Due to
`contractor_house` only at late stages of the project, they were not includ

B.4 contractor_house.

The `contractor_house` contains about 420 hours of data. We asked con
house in 10 minutes, using only basic dirt, wood, and sand, blocks. Each tra
generated world and a timer forcibly ends a trajectory after a 20 minute time l
contractors chose to begin their trajectories by crafting basic tools and build
it was common for the first 2 minutes to be spent crafting a wooden pickax
for an assortment of stone tools before gathering more building blocks and
structure.

C Minecraft environment details

Our Minecraft training environment is a hybrid between MineRL[27]
(github.com/Hexeption/MCP-Reborn) Minecraft modding package. Unlik

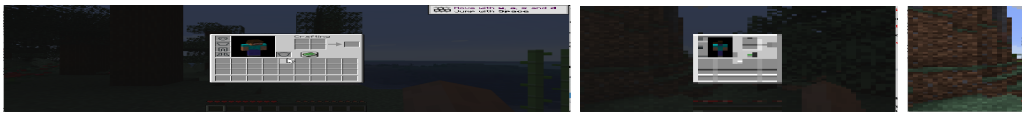

Figure 12: **(Left)** Sample of a Minecraft frame in the original resolution (64
GUI open. The mouse cursor can be seen in the center of the image. This p
player's inventory and can be used to craft very basic items. **(Middle)** We
128x128 for computational reasons. Shown is a downsampled observation v
crafting. This is the resolution used by our models. **(Right)** A 128x128 ob
models without in-game GUI. The health, hunger, hotbar overlays, and agen
lower part of the image.

game, in which the server (or the "world") always runs at 20Hz and the client
can complete (typically at 60-100Hz), in our version the client and server
at the same frequency. This allows us to run the environment slower or fas
avoiding artifacts like missing chunks of the world. The action and observ
to those of MineRL environments and are described in more detail in the
The environment also returns diagnostic information, such as in-game stats
inventory, whether any in-game GUI is open, etc., which we use for tracking
inputs to the models. The episode length is 10 minutes for RL experiments
model evaluations. The agent can "die" in a number of ways, such as staying
and drowning, being killed by hostile mobs, or falling from a tall structure.
episode on agent "death". Instead, just as for humans in the regular Minecra
all its items when it dies and respawns at a random spot close to the initial sp
Minecraft world. The policy state is not masked on death, so the model can
has died and act accordingly.

C.1 Observation space

The environment observations are simply the raw pixels from the Minec
would see. Unlike MineRL, we do not remove overlays like the hotbar, he
animation of a moving hand shown in response to the attack or "use" actio
70 degrees, which corresponds to the Minecraft default. GUI scale (a param
of the in-game GUI) is set to 2, and brightness is set to 2 (which is not a M
very frequently used in online videos). The rendering resolution is 640x360
to 128x128 before being input to the models. We empirically found 128
resolution for which in-game GUI elements are still discernible, and then
compute costs. Whenever an in-game GUI is open, we additionally rend
cursor at the appropriate mouse position to match what a human player's ope
12).

C.2 Action space

Our action space includes almost all actions directly available to human pla
mouse movements, and clicks. The specific binary actions we include are si

One difference between the human action space and our agent's is that we
letters, which is only useful for entering text into the search bar of the crafti
can either do that or browse the recipe book with the mouse, the latter of v
do. However, because we do allow the agent to press letters that are also s
outside of the GUI, the "W" key triggers the `forward` action) agents are a
within the GUI (W, A, S, D, E, Q) that produce letters if the recipe book se
have not seen agents attempt to search the recipe book with these letters. Inst
the recipe book with the mouse or craft by dragging items around the crafti

Action	Human action	Description
forward	W key	Move forward.
back	S key	Move backward.
left	A key	Strafe left.
right	D key	Strafe right.
jump	space key	Jump.
inventory	E key	Open or close inventory and the
sneak	shift key	Move carefully in current direct GUI it acts as a modifier key: wh it moves item from/to the inven bar, and when used with craft i number of items possible instead
sprint	ctrl key	Move fast in the current direction
attack	left mouse button	Attack; In GUI, pick up the stack stack of items in a GUI cell; w click (attack - no attack - attack items of the same kind present in stack.
use	right mouse button	Place the item currently held or u is looking at. In GUI, pick up the a single item from a stack held b
drop	Q key	Drop a single item from the stac is currently holding. If the play it drops the entire stack. In the happens except to the item the m
hotbar.[1-9]	keys $1-9$	Switch active item to the one in a

Table 3: Binary actions included in the action space. https://minecraf Controls has more detailed descriptions of each action.

In addition to the binary (on/off) keypress actions, our action space also incl As with human gameplay, when in-game GUIs are not open, mouse X an agent's yaw and pitch, respectively. When a GUI is open, camera actions Mouse movements are relative (i.e. they move the mouse or camera relative and thus their effect depends on the current position).

Inventory interaction in Minecraft requires fine-grained mouse movements crafting and smelting, while mining and navigating the world can be achie action. To be able to achieve both with the same action space, we impleme as a set of discrete actions with foveated binning along each axis (Fig. 1 experiments we found to improve crafting performance.

D Inverse Dynamics Model Training Details

D.1 IDM Architecture

The IDM model has approximately 0.5 billion trainable weights. The i consecutive image frames (128 frames of video), each of which has dimensi IDM is tasked with predicting the action at each frame. All image pixel by 255.0 such that they lie within the range $[0, 1]$. The first layer of the ID with 128 learnable filters with a temporal kernel width of 5 and spatial k convolution is non-causal, meaning that embeddings at time index t are fun times $t-2$, $t-1$, t, $t+1$, and $t+2$. We found this layer to be extremely in

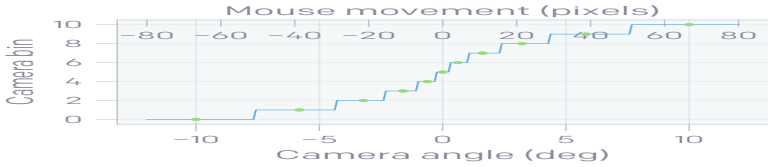

Figure 13: Relative camera angle or mouse movement in pixels vs. action b
used for both X and Y coordinates. The binning is foveated, meaning that binn
for smaller movements and more coarse-grained for larger movements. Th
axis (X and Y). The center of each bin (indicated with green circles) is use
movements (that is, when converting from an action expressed as a bin to a
movement).

as it incorporates neighboring temporal information immediately, and we
IDM performance with and without this layer in Figure 14. This comparison
(1962-hour) IDM dataset.

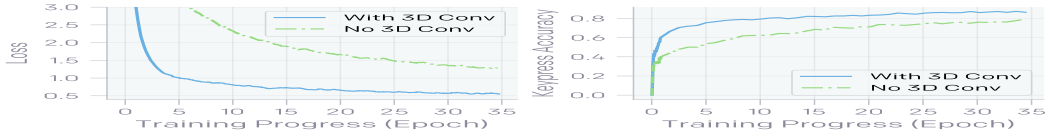

Figure 14: Effect of 3-D Convolution in the IDM Archite

This initial temporal convolutional layer is followed by a ResNet[62] image pro
part of the model, no extra temporal information is shared between neighb
since each frame was first processed with the temporal convolution, som
is present at this stage. The ResNet image processing network is compri
stacks with widths $W = \{64, 128, 128\}$. Each stack is comprised of, in
convolutional layer with 1-pixel zero padding at the embedding boundary
embedding dimensions are the same as the incoming embedding dimension)
(2) a 3x3 max pooling with stride 2 and padding 1 such that the embeddi
halved, and (3) two classic ResNet blocks as defined in He et al. [62] with ea
output channels.

The output of the ResNet stack is flattened into a 1-dimensional vector of
vector for each frame in the video) such that at this stage there are 128 vecto
vector is independently processed with two frame-wise dense layers with 25
then 4096 output activations, respectively. The result is then fed through 4
(unmasked) residual transformer[69] blocks. Each block first has an unmasked at
may attend to future frames, with 32 attention heads of dimension 128 each ar
connection that skips this layer. The embedding is then passed through a
with output dimension 16384 and another with output dimension returning t
connection skips past this pair of frame-wise dense layers (not skipping pa
but skipping the pair). All dense layers have their weights tied through tim
video is processed with the same weights.

Finally, independent dense layer heads for each action are pulled from the
class on/off categorical parameterized with a softmax for each available

categorical for both the discretized horizontal and vertical mouse movement
details on the action space).

Each dense layer or convolutional layer in the network is preceded by a layer
a ReLU non-linearity. Weights are initialized with Fan-In initialization[71] and
zero.

D.2 IDM Training

The total loss for the network is the sum of each independent action predi
key and one for both mouse directions). Each independent loss is the negat
correct action. We use the ADAM[72] optimizer with a linear learning rate
learning rate of 0.003, a batch size of 128 (where each item in the batch is
frames), and a weight decay of 0.01. Hyperparameters were tuned in prelim
IDM is trained on our contractor collected dataset for 20 epochs. This took 4

We add data augmentation to each video segment; augmentations are rand
segment such they are temporally consistent. Using the Pytorch[73] transfor
hue by a random factor between -0.2 and 0.2, saturation between 0.8 and
0.8 and 1.2, and contrast between 0.8 and 1.2. We also randomly rotate the
degrees, scale it by a random factor between 0.98 and 1.02, shear it betwe
translate it between -2 and 2 pixels in both the x and y dimensions.

Due the large computational cost of running all of the experiments in thi
are from one run of training (for IDM, BC, and RL training): this non-ide
because deep learning training tends to be low variance[74,75] and because w
from sweeps (e.g. on dataset size) that suggest overall trends.

D.3 Generating Pseudo Labels with the IDM

Section 4.1 shows that inverse dynamics modeling is a much easier task
because IDMs can be non-causal. The IDM is trained to simultaneously pr
each video sequence, so the IDM will effectively be causal for frames at t
because future frames are not included in the sequence. For this reason, w
video using a sliding window with stride 64 frames and only use the pse
frames 32 to 96 (the center 64 frames). By doing this, the IDM prediction
video clip is never used except for the first and last frames of a full video.

E Foundation Model Behavioral Cloning

E.1 Foundation Model Architecture

The behavioral cloning model architecture is the same as the IDM architectu
D.1 except that we modify the architecture so that it is causal (i.e. cannot see
predictions). This means the BC architecture does not have the initial non
IDM has (this layer is omitted completely). Furthermore, the residual tra
causally masked (as is standard in language modeling) and we do Transfor
where frames can attend to keys and values from past batches within the sa
Transformer-XL-style relative attention position embedding.

E.2 Null Action Filtering

The most common action humans take is the null action (no keypresses or m
accounts for 35% of all actions they take. Among other reasons, a player ma
wait for something in the game to finish, to pause between actions, or to tak
of water. Early on in the project we found that the BC model would take a n
35% of null actions, often upwards of 95%. In order to prevent this behavior
null actions from the dataset. We compare a few different treatments: we
been 1, 3, or 21 frames of consecutive null actions, and include a treatmen
any null filtering. Null action filtering generally helps, increasing all crafti

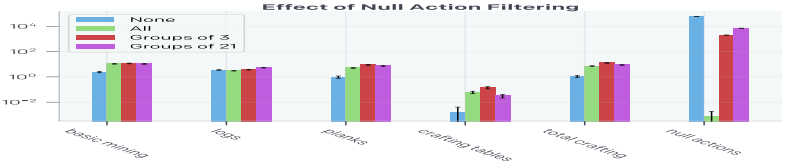

Figure 15: **(Left)** Effect of Null Action Filtering during training. We comp
and number of sampled null action during rollouts (rightmost group of col
treatments: no null action filtering (blue), filtering all null actions (green),
3 or more null actions (red), and filtering only groups of 21 or more null a
Hierarchical versus Factored Action Spaces.

Filtering only groups of 3 performed slightly better than filtering all null actio
experiments indicated that filtering all null actions was better; however, af
and after we had already trained our largest models, we found that filtering
null actions performed best. Due to compute constraints we were not able
with this setting, but doing so would be a reasonable choice for any future v

E.3 Joint Hierarchical Action Space

We originally worked with a factored action space, where each keypress co
or off, and this choice was independent of whether the mouse was being m
issues for modeling the human behavior distribution exactly. Say for a given s
50% probability (a) move forward and attack or with 50% probability (b) m
item. The best a factored distribution can do is to assign 50% probability to
actions because it chooses to press each button simultaneously and independ
for details on the entire action space.

For this reason, we implemented a joint distribution over actions; however, t
over 20 binary buttons and two mouse movement dimensions discretized i
result in in $2^{20} \times 11^2 \approx 1.2 \times 10^8$ possible combinations. This is far too lar
the final layer from the transformer stack with a dimension of 4096 would ne
combination resulting in $4096 \times 1.2 \times 10^8 \approx 5.2 \times 10^{11}$ parameters for t
order to reduce this we noted that many buttons in Minecraft have no effe
pressed; for example, if a player tries to move forward and backward at the sa
place. Below we list the the sets of mutually exclusive actions. Furthermore
exclusive with all other buttons and mouse movement.

Mutually Exclusive Actions
`forward, back`
`left, right`
`sprint, sneak`
`hotbar.[1-9]`

Even reducing the joint action space to reflect these mutually exclusive comb
huge action space when combined with the discretized mouse movements, i.e.
5.2×10^5. This calculation results from 3^3 for the 3 sets of 2 mutually excl
taking neither in the set is an option, $\times 10$ for the 9 hotbar keys or no hotba
remaining binary 4 keys: `use`, `drop`, `attack`, and `jump`, $\times 11^2$ for mouse
$+1$ for the `inventory` button which is mutually exclusive with all other acti
quite large so we chose to implement a hierarchical binary action for camer
this action is on, then there is a secondary discrete action head with 121 class
of mouse movements because each discretized mouse direction has 11 bins

to move the mouse. If the hierarchical action is off, then there is no mo
the secondary mouse movement action is masked during training, and the
need not be sampled during evaluations. While this no longer models the fu
quite a bit better than the factored action space since dependencies betwee
whether or not to move the mouse (although not which mouse movement) a
resulting action space has dimension $3^3 \times 10 \times 2^4 \times 2 + 1 = 8461$ (the 11
for camera movement has been replaced by a multiplier of 2 here, correspo
for whether or not to move the mouse) with an additional 121-dimension h
movements. In the future it would be interesting to implement sequential co
more completely model the joint distribution.

In Figure 15 (right) we compare environment rollout performance betwe
hierarchical joint action space and with the factored action space. Environ
comparable; however, we see that the factored action space model sample
This is an important example of the factored action space failing to correctly
the dataset because, due to null action filtering, there are 0 null actions in t
train on. Despite this, the factored model samples many null actions becaus
key is not conditioned on other keypresses.

E.4 Foundation Model Training

The foundation model training is similar to the IDM training, with the ex
IDM-generated pseudo labels. The hyperparameters used for foundation mo
Table 4.

Hyperparameter	Value
Learning rate	0.002147
Weight decay	0.0625
Epochs	30
Batch size	880

Table 4: Hyperparameters for foundation model traini

F Behavioral Cloning Fine-Tuning

Behavior cloning fine-tuning is similar to the foundation model training, excep
subset of all the videos (`early_game` dataset, described in A.3) with pseudo
(`contractor_house` dataset, described in B.4) with ground-truth labels. Th
for behavior cloning fine-tuning are listed in Table 5. We used 16 A100 GPU
fine-tuning on `contractor_house` dataset, and 16 A100 GPUs for about
on `early_game` dataset.

Hyperparameter	Value
Learning rate	0.000181
Weight decay	0.039428
Epochs	2
Batch size	16

Table 5: Hyperparameters for behavior cloning fine-tu

G Reinforcement Learning Fine-Tuning

G.1 Reinforcement Learning Fine-Tuning Training Details

RL experiments were performed with the phasic policy gradient (PPG) algor
based on the proximal policy optimization (PPO) algorithm[77] that increas
performing additional passes over the collected data to optimize the value

auxiliary value function. These algorithms have been described extensivel
so here we describe them only briefly. A major inefficiency when training
that, to remain on-policy, one can only take a single gradient step before
to be gathered to continue optimization. To alleviate the potentially destr
multiple optimization steps in a single iteration, PPO prevents the policy f
in a single step by clipping the loss when the difference between the curre
before the update becomes too large.[77] We also use generalized advantage e
can speed-up credit assignment by looking more than 1 step into the futur
advantage of an action, with the look-ahead being determined by hyperpara

PPG improves the sample efficiency of PPO when the policy and value f
network by following different optimization processes for the policy, the v
shared representation. PPG splits optimization in two phases: a wake phase
wake phase, the policy and value function are optimized as in normal PPO
exception being that every sample is used at most once, which prevents the p
these samples. In the sleep phase PPG optimizes the value function and an
(which is optimized with the exact same loss as the regular value function, bu
during training), while keeping a Kullback-Leibler (KL) divergence loss to t
of the sleep phase to ensure that the policy does not change. Because the
in this step, PPG does allow samples to be reused multiple times in this
behind optimizing the value function during the sleep phase is that value f
less sensitive to being trained multiple times on the same sample. Optimi
function does not directly affect either the value function or the policy, but it
representation of both functions (the assumption being that predicting the
encoding all features that are important for distinguishing states). The c
losses (value function loss, auxiliary value function loss, and KL loss) are
experiments a single iteration consists of two sleep cycles and one wake cyc

Because the value and auxiliary value functions are not optimized durin
are initialized at the start of RL fine-tuning. Each value function is implem
connected layer on top of the last residual transformer block of the pretrained
The weights of the auxiliary value function are randomly initialized while th
value function are initialized with zero weights, which appeared to prevent d
in training that could happen with a randomly initialized value function. To pr
loss from having gradients that depend greatly on the magnitude of the re
value-function target by subtracting the mean and dividing by the standa
estimated through an exponentially weighted moving average.

To prevent catastrophically forgetting the skills of the pretrained network w
apply an auxiliary KL divergence loss between the RL model and the frozen
loss is defined as:

$$L_{klpt} = \rho \mathrm{KL}(\pi_{pt}, \pi_\theta)$$

Where π_θ is the the policy being trained, π_{pt} is the frozen pretrained po
Kullback-Leibler divergence between the policy being trained and the pret
coefficient to weight this loss relative to other losses.

In the fine-tuning experiments, this KL divergence loss replaces the commo
loss, which is often added to RL experiments to encourage exploration.[79,80]
maximization is that, when all actions appear to have equal value, such as
learned about the next reward, it should maximize its entropy to increase the
the next reward. Blindly exploring by maximizing entropy is effective wh
spaces are sufficiently small or the reward is sufficiently dense, but becom
state and action spaces are large and rewards are sparse, which is the case
task. Instead of blindly exploring through uniform-random actions, we ass
policy has an action distribution that is much more likely to take sequen
to interestingly new states, and thus, in states where the agent assigns eq
actions, it should mimic the action-distribution of the pretrained policy inste
action distribution. In experiments with a randomly initialized policy we
maximization loss with a coefficient of 0.01, which has been an effective se
work.[30] Empirically, we found that a high coefficient ρ for this KL diverge
the agent from properly optimizing the reward function while a low coeffic

Hyperparameter	Value
Learning rate:	2×1
Weight decay:	0.04
Batch size:	40
Batches per iteration:	48
Context length:	128
Discount factor (γ):	0.999
GAE λ:	0.95
PPO clip:	0.2
Max Grad norm:	5
Max Staleness:	2
PPG sleep cycles:	2
PPG sleep value-function coefficient:	0.5
PPG sleep auxiliary value-function coefficient:	0.5
PPG sleep KL coefficient:	1.0
PPG sleep max Sample Reuse:	6
KL divergence coefficient ρ:	0.2
Coefficient ρ decay:	0.999

Table 6: Hyperparameters for RL experiments. These are the hyperparamete
two exceptions. First, when fine-tuning from the early-game model witho
in addition to the KL divergence loss being set to 0, the learning rate was s
setting out of a sweep over 5 different learning rates), as we found that perfo
lower with the standard learning rate of 2×10^{-5} and the agent did not ev
We suspect that the reason that the learning rate needed to be lowered wh
KL loss is that the KL loss prevents making optimization steps that change
single step, especially in early iterations when the value function has not
the KL loss thus makes it possible to optimize with a higher learning rate.
RL from a randomly initialized policy there is no KL divergence loss or K
instead we use an entropy bonus of 0.01, which reportedly worked well in p

protecting the learned skills of the pretrained policy and preventing catastro
we start with a relatively high coefficient ρ and decay it by a fixed factor afte
This method protects policy skills in early iterations while guaranteeing that
maximize the reward function, regardless of how different its behavior has
the pretrained policy.

For the reward function we estimated the rough quantities of each item tha
gather when trying to craft a diamond pickaxe, and we reward the model
quantity for each item. We started these estimates by iterating over the techno
a diamond pickaxe and adding the requirements for each item to the rewar
added a diamond pickaxe to the reward function, then we added the 3 diamon
for crafting a diamond pickaxe, then we added the 1 iron pickaxe require
and so on). Then we added coal and torches to the reward function, with
when smelting iron and for crafting torches while the torches themselves
prevent enemies from spawning. Finally, we reward the model for bringing a
required to craft all items in the reward function, but we reward up to 8 log
fuel or crafted into a crafting table or sticks if the agent runs out. In practice
the additional logs, places the torches, or uses coal as fuel when smelting,
was based on human expectations on what would be useful to execute this ta
around how an RL model behaves after training. Finally, to encourage th
diamonds and crafting diamond pickaxes after it has crafted its first diamond
a limit on the number of diamonds or diamond pickaxes that would be rewa

The rewards for the different items are separated into 4 tiers, roughly depend
would usually get the relevant item. The first tier consists of all wooden an
base reward of 1, the second tier consists of all items requiring coal with a ba
tier consists of all items requiring iron with a base reward of 4, and the fina
base reward of 8. Thus items later in the sequence of items towards a dia

Item	Quantity rewarded	Reward per
Log	8	
Planks	20	
Stick	16	
Crafting table	1	
Wooden pickaxe	1	
Cobblestone	11	
Stone pickaxe	1	
Furnace	1	
Coal	5	
Torch	16	
Iron ore	3	
Iron ingot	3	
Iron pickaxe	1	
Diamond	inf	
Diamond pickaxe	inf	

Table 7: Reward per item and total quantity rewarded

give a higher reward. To make sure that the agent does not over-value ite
be gathered in bulk (e.g. the agent is rewarded for up to 20 planks but onl
which can cause the agent to focus on planks at the expense of creating a c
the base reward of each item by the total quantity that the agent gets rewar
of determining the reward, the total quantity for diamonds is 3 and the tot
pickaxes is 1, even though we did not put a limit on the number of these ite
example, the agent is rewarded for 3 iron ore, which has a base reward of 4
and up to 3 blocks of iron ore are rewarded, thus the reward per block of iron
and reward for each item are listed in Table 7.

While every item in the sequence towards a diamond pickaxe is rewarded, th
sparse and, in some cases, even deceptive. The sparsity comes from the fact t
of actions to find the next reward, even after the agent has acquired all the
(e.g. human players often take more than 10,000 actions to find a diamon
pickaxe). The reward function can be deceptive when the most efficient met
can make it far more difficult to get the next item. For example, a good s
craft a stone pickaxe quickly is to mine (i.e. spend a few seconds to pick up
crafting a wooden pickaxe, such that the agent has immediate access to a cr
has collected enough cobblestone. However, the fastest way to get a reward f
is to mine down immediately after crafting a wooden pickaxe, while leaving
Thus following the optimal strategy for gathering cobblestone makes it more
a stone pickaxe.

Experiments ran for approximately 6 days (144 hours) on 80 GPUs (for p
56,719 CPUs (mostly for collecting rollouts from Minecraft). In this time t
roughly 4,000 optimization iterations and collected roughly 1.4 million Mine
of 12,000 frames each, for a total of 16.8 billion frames.

G.2 Reinforcement Learning Fine-Tuning Additional Data

Additional figures that are helpful for understanding the main results of the RL
are presented in this section. First, we show the items-over-training figure w
the early-game model without a KL loss (Fig. 16). When training withou
only learns to obtain the four items that the early-game model is capable of
are logs, planks, sticks, and crafting tables. Second, we present preliminar
we directly compare RL fine-tuning from the house-building model and
early-game model (Fig. 17). These experiments differ from the main expe
treatments shown here, the KL loss coefficient was set to 0.4, the learning
and the reward for each item was $1/quantity$ for all items (i.e. items closer
did not have an increased reward). While RL fine-tuning from the house-

worked better than RL fine-tuning from the early-game model, fine-tuning fro
worked better after 800,000 episodes and showed signs of smelting iron i
early-game model was chosen for the main experiments.

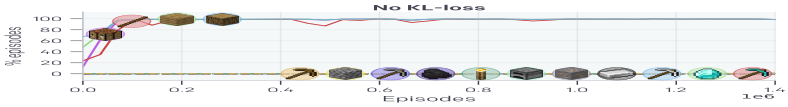

Figure 16: Items obtained when RL fine-tuning from the early-game model
model learns to obtain all items that the early-game model can craft zero-sho
sticks, and a crafting table. In contrast to the treatment with a KL-penalty, it
beyond these initial four, likely because skills that are not performed zero-
model thus does not initially see any reward, are catastrophically forgotten
are learned.

Figure 17: Preliminary experiments when RL fine-tuning from the early-ga
RL fine-tuning from the house-building model. **(Left)** While reward initial
fine-tuning from the house-building model, fine-tuning form the early-game
a slightly higher reward. **(Right)** RL fine-tuning from the early-game mode
of smelting an iron-ingot, which is why the early-game model was chosen f
experiments.

H Foundation Model Scaling

In early experiments we found that increasing model size led to models
learning regime longer into training.[63] Here we compare the 0.5B model de
both a 248M and 71M parameter model. Both of these models are trained fo
to the 30 epochs the 0.5B model trained for. These models have the same
model but each layer in the 248M parameter model has 1/2 the width and
parameter model 1/3 the width. The 71M model was trained with an initial le
batch size of 480, and weight decay of 0.044506. The 248M model had a
0.001831, batch size of 640, and weight decay of 0.051376.

In Figure 18 we show validation loss on `web_clean` with IDM pseudo-labe
dataset used to train the IDM with ground truth labels collected during contr
environment performance for the 71M, 248M, and 0.5B models. While la
validation loss on `web_clean`, these results do not tell the clear story that
than its smaller counterparts. The 71M model has the lowest contractor data
highest `web_clean` loss, and it also has the best zero-shot environment perf
that the 71M model even had non-zero wooden tool crafting (Fig. 18 botton
also appears to be better at crafting than the 0.5B, and also has lower contra

While the zero-shot results suggest smaller models are better, fine-tuning te
fine-tuning to `contractor_house`, model size rank ordering reverses an
performs best both in validation loss (Fig. 19 left) and in environment perf

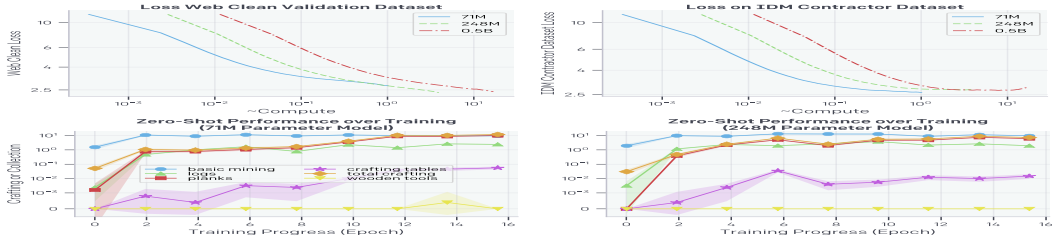

Figure 18: Training and Zero-Shot Performance versus Model Scale. In the is compute normalized to that used by the 71M parameter model, such that af the 71M model has used 1 "compute". The 248M parameter model and the 7 the same amount of data (15 epochs), and the 0.5B parameter model is train **(Top Left)** Loss on the `web_clean` validation dataset. **(Top Middle)** Loss dataset; note that these models were trained only on `web_clean` and not **(Top Right)** Zero-shot environment rollout performance at the end of traini environment rollout performance over training for the 71M model (bottom le middle), and 0.5B model (bottom right).

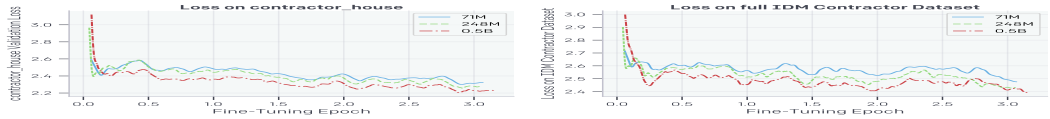

Figure 19: `contractor_house` fine-tuning performance versus model s `contractor_house` holdout validation set. **(Middle)** Loss on the full con to train the IDM; this dataset is disjoint from `contractor_house`. **(Righ** performance at the end of fine-tuning.

followed by the 248M model and then the 71M model. Environment mode using the same game engine that we use to collect contractor data, which c from videos taken from the web. It is plausible that the larger models peculiarities in web data during pretraining since they have worse contract middle), and this causes them to perform more poorly in the environment hypothesize that because the `contractor_house` dataset we fine-tune to is engine, the larger models that are a better overall Minecraft prior (as indicat validation loss in Fig.18 top left) can quickly shift their low level features t coming from our game engine, resulting in better environment rollout perfo is further supported by Fig. 19 (middle) showing loss on the contractor da training, which has no overlap with `contractor_house`. After just a few `contractor_house`, all models quickly improve in loss on the full IDM larger models now performing best. While not conclusive, we believe thi some intuition for future studies of model scaling for sequential decision m

I Text Conditioning

Goal-conditioned policies[81,82] make it possible for a single agent to per
goals in a single environment, which is particularly relevant in open-ende
Minecraft. In recent work, goal specification has increasingly taken the f
languages[83], or even natural language[84,85]. The benefits of language-co
tremendous, especially *natural*-language-conditioned agents, as their goa
variety of potentially very complex tasks. Text conditional models have sho
perform tasks zero-shot (or learn them few-shot) including generalizing in
compositional and combinatorial possibilities allowed by natural language (
2^{86}). We hypothesize that we should expect similar capabilities to emerge
conditioned virtual agents, if they are similarly trained on enormous amount
a natural language description to a sequence of actions that completes the
section we take preliminary steps toward that future. Our preliminary exper
that it is possible to pretrain a natural-language-conditioned model for Min
approach presented in this paper (VPT) plus conditioning on the speech
videos.

In online videos, the human actor sometimes indicates their intent in th
(e.g. "Let's go chop some trees to make a wooden axe" or "now let's lear
Photoshop"). Conditioning on this closed caption data could produce a *stee*
i.e., it may later be possible to condition the model with text such as "I am
pickaxe" or "I am going to build a house," and have the agent perform those
than simply follow typical human behavior (as was investigated in the rest of
way to produce a steerable agent is via RL fine-tuning, which we could h
by adding a bit indicating the task to be completed, as has been done in
conditioning on natural language offers many benefits over that approach
powerful, being able to express any task. Second, one does not need to pr
be completed ahead of time. This would allow for general, capable, zero-s
extending those capabilities to embodied tasks such as completing tasks on c
3D worlds. Third, text conditioning can be used even when tasks are diffic
functions (e.g. "Let's build a house" or—if the agent is capable of doing it—m
"I will now build a castle surrounded by a moat"). In the limit, VPT+text co
powerful, capable, natural-language-conditional agents with the powers of G
instructions, and complete tasks zero or few shot, but in the form of agen
worlds, complete tasks on computers, and in other similar embodied seque
We do not reach those lofty goals in this work, but we began a first step to
direction.

Many Minecraft videos feature audio commentary from the player. This co
present in the form of closed captions for the videos, or could be extracted p
speech recognition (ASR).[87] Our dataset features about 17k hours of conte
captions.

We fine-tuned the 220 million parameter VPT foundation model used in
periments (chosen vs. 0.5B for the same reason: to reduce compute co
text-conditioning input on the subset of our data for which closed captions are
conditioning input, we first split videos into 30 second chunks. The same text
frame in a given chunk, and is made up of all the closed captions occurring w
as the line of text preceding and following the chunk (if any). Because th
95%) of our closed caption data lacks capitalization and punctuation, it is pu
library[88]. We then obtain a text embedding vector of length 4,096 from the Op
which is processed by a randomly initialized multi-layer perceptron (MLP) w
size 2,048. The resulting activations are added for each frame to the pretr
before the transformer layers (`pretransformerActivations += mlp(t`
model is fine-tuned for four epochs.

Our model shows evidence of steerability. When conditioned on sentences
explore (such as "I'm going to explore" and "I'm going to find water") the a
farther from its spawn point (Figure 20a). Additionally, we can steer the agen

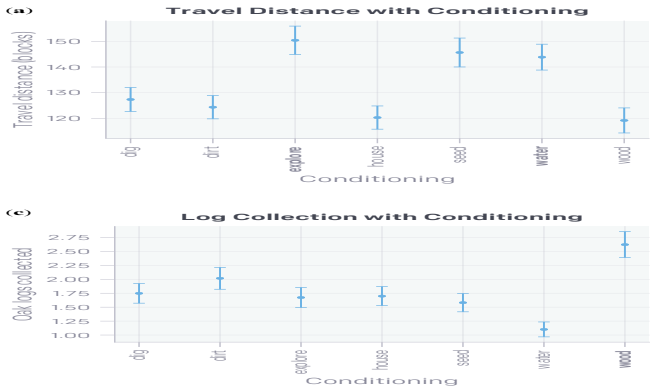

Figure 20: Evidence for conditioning. In each plot, the variants expected to
bold. The strings corresponding to each variant are shown in Table 8. Statis
minute episodes. **(a)** Distance traveled by the agent . Both "explore" and "w
encourage a steerable agent to move more than when doing other tasks, whi
(which is needed to get seeds) is not present in all biomes, which is likely w
produces more travel (as the agent sometimes needs to move to a biome
distance is the Euclidean distance from the spawn point to the farthest point
the episode on the horizontal (x-z) plane. **(b)** Collection of wheat seeds. Th
substantially more than other variants, as expected of a steerable agent. **(c**
most common type of wood) logs. The "wood" variant collects significantl
be expected of a steerable agent (we speculate that the "water" variant collec
no trees in water). **(d)** Collection of dirt. The "dirt" and "dig" variants collec
the variants that are (indirectly in the case of "dig") conditioned to collect dir
aim at the ground rather than at grass or trees when collecting seeds or woo
the slightly higher amount of dirt collected by these variants. In all cases
confidence intervals of the mean, over 1,000 episodes per conditioning varia
the bars in each bar plot do not overlap are statistically significantly differe

Variant name	String
dig	I'm going to dig as far as possible
dirt	I'm going to collect dirt
explore	I'm going to explore
house	I'm going to make a house
seed	I'm going to collect seeds
water	I'm going to find water
wood	I'm going to chop wood

Table 8: Strings corresponding to each conditioning var

early game items such as seeds, wood, and dirt by conditioning with text such
seeds/chop wood/collect dirt'' (Figure 20b,c,d).

While our results show some level of steerability, more work is required to i
we were not able to successfully steer agents to gather flowers or to hunt, bo
in the early game, but less common (and, in the case of hunting animals, m
gathering dirt, wood, or seeds. Likewise, an experiment in which the ag
crafting window and various resources, and conditioned to craft a given i
craft a wooden axe'') failed to show that the conditioning had a significant e
crafted. Instead, it seemed the agent was more influenced by the prior, un
of what human players would craft next given the resources available, wh
since in Minecraft, especially in the early game, there is a relatively cons
resources in a specific order go produce more powerful tools (Fig. 6). For e
the resources to make a stone pickaxe and we asked it instead to make a (we
often would make the stone pickaxe anyway. Finally, looking at videos of a
convince us that the ''house'' conditioning causes the agents to take more s
house than other variants.

Thus, our results show that it is possible to train a somewhat steerable natur
agent. However, its steerability is still too weak to be practically useful, an
believe could be accomplished with more research, data, and training com
research direction is to have the model predict future text as well as just the