ASH: Agents that Self-Hone in Long-Horizon Worlds

Benjamin Schneider
University of Waterloo
benjamin.schneider@uwaterloo.ca

Xavier Schneider
University of Waterloo
xschneider@uwaterloo.ca

Victor Zhong
University of Waterloo
victor.zhong@uwaterloo.ca

Sun Sun
National Research Council Canada
sun.sun@nrc-cnrc.ca

Abstract

Long-horizon visuomotor tasks remain a fundamental challenge in AI, as current methods rely on hand-engineered rewards or action-labeled demonstrations, neither of which scales. We introduce **ASH**, an agentic system that learns a long-horizon policy from unlabeled, noisy internet video, without reward shaping or expert annotation. ASH follows a self-improvement loop; when it gets stuck, ASH learns an Inverse Dynamics Model (IDM) from its own trajectories, and uses its IDM to extract supervision from relevant internet video. ASH uses unsupervised learning to identify key moments from large-scale internet video and retains them as long-term memory — allowing it to tackle long-horizon problems. We evaluate ASH on two complementary environments demanding multi-hour planning: *Pokémon Emerald*, a turn-based RPG, and *The Legend of Zelda: The Minish Cap*, a real-time action-adventure game. In both games, behavioral cloning, retrieval-augmented and zero-shot foundation-model baselines plateau, while ASH sustains progression across our 8-hour evaluation. ASH reaches an average of $11.2/12$ milestones in Pokémon Emerald and $9.9/12$ in Legend of Zelda, while the strongest baseline gets stuck in both environments at an average of $9.3/12$ and $7.8/12$ milestones, respectively. We demonstrate that self-improving agents are a scalable recipe for learning in long-horizon worlds. Our project page is https://toolbox-hq.github.io/ASH.

1 Introduction

Developing agents capable of mastering long-horizon, open-ended tasks remains a fundamental challenge in artificial intelligence [1, 2]. Unlike agents for short-horizon planning, long-horizon agents must navigate complex, open-ended worlds over millions of steps. This requires multi-hour planning, fine-grained motor control, and the ability to adapt to new visual contexts.

Despite progress in visuomotor learning, current approaches struggle in these long-horizon settings. Behavioral Cloning (BC) methods [3] typically learn a policy from massive offline datasets. At this scale, assembling high-quality data is a significant challenge, requiring domain knowledge to curate data and extensive human annotation [4]. Moreover, much of the available data may be irrelevant to the agent's current objectives, increasing both training compute and susceptibility to noise. Long-horizon tasks exacerbate these challenges [5]: a single trajectory may span hours, making it intractable to model dependencies across the full sequence. As a result, BC policies typically condition on short-term memory and struggle when a current decision depends on old information [6].

While Reinforcement Learning (RL) has demonstrated success in complex environments, it remains heavily dependent on dense reward engineering [7, 8, 4]. In a domain spanning tens of hours with diverse sub-tasks, manually designing a reward function for every intermediate milestone is brittle, domain-specific, and labor intensive. Like BC, modeling dependencies across hour-long sequences using RL is intractable. Instead, a common approach is to use short-term memory combined with

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



Figure 1: ASH self-improves over the course of a multi-hour playthrough by retrieving and learning from relevant internet video. In Pokémon Emerald, ASH learns to battle trainers, catch Pokémon, and heal its team at Pokémon centers. In Zelda, ASH discovers how to wield a sword and apply game mechanics to solve puzzles. These skills compound, allowing ASH to achieve sustained progression.

"next sub-objective" conditioning [8, 4], which amounts to a human manually laying out a path for the agent to follow. These approaches have difficulty learning skills and knowledge beyond their human-engineered objectives, making them unsuitable for open-ended environments.

To address these limitations, we introduce **ASH**, an agentic system that follows a **dynamic bootstrap** loop. When ASH gets stuck, it searches for helpful demonstrations on the internet, and learns from them. ASH uses its current trajectories to retrieve relevant examples from online video and to update its Inverse Dynamics Model (IDM). ASH uses unsupervised learning to identify key moments from the retrieved video. ASH bootstraps its policy using these key moments as **long-term memory** and the retrieved video demonstrations labeled by the IDM.

We evaluate ASH in two complementary long-horizon environments: *Pokémon Emerald*, a turn-based RPG combining strategic battling with open-ended exploration, and *The Legend of Zelda: The Minish Cap*, a real-time action-adventure game that requires puzzle solving and fine-grained motor control. Both require dozens of hours of decision-making across millions of distinct visual states, which makes them useful as long-horizon testbeds. ASH interacts with the games as a human does: by observing the screen and pressing buttons.

In our evaluations, ASH progresses further than behavioral cloning [4], retrieval-augmented [9], and zero-shot foundation-model [10] baselines. Baselines plateau on intermediate milestones, while ASH continues to acquire new skills—battling trainers, catching Pokémon, exploring areas, and defeating enemies—across multi-hour horizons. After 8 hours, ASH achieves an average of $11.2/12$ progression milestones in Pokémon Emerald and $9.9/12$ milestones in Legend of Zelda. The strongest baseline gets stuck in both environments, achieving an average of 9.3 and 7.8 milestones, respectively.

ASH's main contributions are as follows: **(1)** We introduce dynamic bootstrapping for long-horizon learning. **(2)** We propose long-term memory created from salient moments in large-scale internet video. **(3)** Through experimentation, we demonstrate that dynamic bootstrapping and long-term memory create a scalable recipe for long-horizon learning — capable of multi-hour planning, fine-grained motor control, and the ability to adapt to new visual contexts.

2 Related Work

Internet-scale behavioral cloning. A large body of work trains visuomotor policies via BC on observation-only video at scale. VPT [4] creates pseudo-actions for thousands of hours of Minecraft

video with an IDM trained on contractor-collected gameplay [3], and remains the strongest reported Minecraft BC backbone [11]. A parallel *imitation-from-observation* literature recovers actions without an explicit IDM [12, 13, 14, 15]. ASH departs from these approaches in two ways: it does not rely on a single statically curated training corpus or a fixed action-labeling model. Instead, ASH is an agentic system that uses the trajectories of its agents to update its IDM and retrieve data for policy bootstrapping.

Retrieval-augmented decision-making. Retrieval has been used to give agents relevant context [16, 17, 18], to condition policies on prior experience [19, 20], and to guide planning [21]. In these methods, retrieval selects a small number of relevant documents, trajectories, or examples to condition the agent during inference. ASH uses retrieval differently: it retrieves a task-relevant training set from a much larger internet-scale corpus, then updates its weights using that data.

Self-improvement from model-generated data. A growing line of work bootstraps a learner from data produced by an earlier version of itself: STaR [22] and ReST [23] for reasoning, self-play for games [24, 25], and self-distillation in robotics [26]. ASH fits this template, but its improvement comes from retrieved internet video rather than from an external verifier or reward model, which are often unavailable in open-ended environments [27].

Long-horizon control in visuomotor agents. Long-horizon control requires agents to retain information that may become useful much later. However, full-history conditioning is computationally intractable, while short fixed-length memory is efficient but can lose information needed for planning [6]. BC and RL are dominated by bounded-memory policies, including recurrent policies that compress history into a fixed-size state and transformers that condition on a fixed context window [28, 4, 8]. Memory-augmented language architectures [29, 30, 31] and skill-library agents such as Voyager [32] offer alternatives. Rather than being authored by an LLM or hand-engineered, ASH autonomously discovers key moments from internet data, and retains those moments as long-term memory when it encounters them in the environment.

RL in open-ended environments. RL can achieve strong results when dense rewards, self-play objectives, or programmatic verifiers are available [33, 25, 34, 35, 8]. However, these assumptions are hard to satisfy in open-ended environments. For example, Pleines et al. [7] train a PPO agent on Pokémon Red by manually creating a trail of rewards for the agent to follow. They report high failure rates when the agent encounters situations not anticipated by the reward engineer, and must continually adapt their rewards to correct for these failure modes. By contrast, ASH's training signal comes from unlabeled demonstrations, a scalable approach to open-ended long-horizon learning.

Foundation-model agents and scaffolded play. A separate line of work uses pretrained foundation models with extensive scaffolding as agents. Voyager [32] uses an LLM and a learned skill library in a Minecraft code-as-action loop, and Karten et al. [36], ClaudePlaysPokemon [37] apply pretrained models to Pokémon Emerald with scaffolds that read underlying game memory. These systems demonstrate emergent behavior but rely on privileged state access to make the games interpretable by an LLM [38]. Unlike these approaches, ASH offers a path to long-horizon learning beyond environments where privileged state is available.

Long-horizon games as testbeds. Prior testbeds for long-horizon learning include Minecraft [4, 8, 32, 27], NetHack [39], and Pokémon [7, 36]. We add *The Legend of Zelda: The Minish Cap* as a complement to Pokémon Emerald. Zelda contains real-time action and puzzle solving, while still offering long-horizon objectives. This contrasts with the turn-based, strategic nature of Pokémon.

3 ASH

Preliminary. An agent interacts with an environment $\mathcal{E}$ that, given action $a \in \mathcal{A}$, returns observation $o \in \mathcal{O}$. We write *observation-action trajectories* as $\tau = (o_1, a_1, \ldots, o_T, a_T)$ and *observation-only trajectories* as $\tau^{obs} = (o_1, \ldots, o_T)$. A policy specifies a distribution over the agent's next action given the information available to it, written $\pi \sim p_\theta(a_t \mid \cdot)$. We use end-exclusive slicing $A_{i:j} = (A_i, \ldots, A_{j-1})$. A detailed symbol glossary appears in Appendix G.

Behavioral cloning (BC) [3] trains π via supervised learning using cross-entropy loss on observation-action trajectories. However, scalable observation-only sources such as YouTube [4] lack action labels. We use an IDM to supply pseudo-actions from these observation-only trajectories; $\text{IDM}(\tau^{obs}) = (\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_T)$. The IDM is trained via supervised learning on a dataset of observation-action

trajectories. We note that training an IDM is strictly easier than training a policy, as the IDM can use future information to predict pseudo-actions [4].

We use HDBSCAN [40] clustering to discover recurring key moments because (1) the number of clusters (key moments) is unknown a priori, (2) it rejects outliers as noise rather than forcing assignment, and (3) HDBSCAN's approximate-predict supports online assignment without refitting.

Overview. ASH is analogous to a human that plays the game, gets stuck, and finds helpful video on the internet. We deploy N agents under a single shared policy π conditioned on short- and long-term memory, and collect trajectories $\tau_{1\sim N}$. A key-moment classifier $\mathcal{K}(\tau^{obs}, t)$, derived from internet data, is used to determine if the observation at t is a new key moment. When an observation is classified as a new key moment, it gets added to an agent's long-term memory and resets a per-agent stuck-timer. When any timer exceeds threshold Δ, ASH *dynamically bootstraps*: it uses the agents' trajectories to update the IDM, and to retrieve visually similar internet video. These observation-only videos are used to update $\mathcal{K}$. The IDM then labels the videos with pseudo-actions, producing trajectories that are used to update π. Full ASH pseudo-code can be found in Appendix B.

Algorithm 1 ASH

Inputs: initial policy π, IDM, key-moment classifier $\mathcal{K}$, internet corpus $\mathcal{D}^I$

Step 1: Inference

Infer N agents in parallel under shared policy π, conditioning each action on per-agent short-term and long-term memory.
On each step, use $\mathcal{K}$ to detect new key moments and append them to that agent's long-term memory.
Halt when any agent fails to find a new key moment for Δ steps (*stuck*).

Step 2: Retrieval

Embed agent trajectories $\tau_{1\sim N}$ with DINOv2.
Retrieve top-k visually similar videos from $\mathcal{D}^I$ per trajectory, union into $\mathcal{D}^R$.

Step 3: Bootstrapping

Update $\mathcal{K}$ on $\mathcal{D}^R$ to discover new key moments.
Update IDM on $\tau_{1\sim N}$.
Update π on $\mathcal{D}^R$, using the IDM for pseudo-actions and $\mathcal{K}$ to construct long-term memory.
`GoTo` **Step 1**.

Architecture. In the long-horizon setting, a decision can hinge on information from many steps ago. ASH resolves this with a *dual memory*: a short-term memory of the w_s most recent observation-action pairs for reactive control, and a long-term memory of observations ρ that are the w_l most recent *key moments*. The policy is therefore $\pi \sim p_\theta(a_t \mid \tau_{t-w_s:t},\ \rho)$. ASH uses an image tokenizer ϕ (SigLIP [41]), an IDM with VPT's bidirectional-attention architecture [4], and the causal transformer block architecture from Yang et al. [42], repeated 28 times, for $\mathcal{T}$.

Training. We train the policy π via gradient descent on observation-only trajectories τ^{obs} of internet data. Each τ^{obs} is divided into non-overlapping samples of length w_s, denoted $\tau^{obs}_{t:t+w_s} = (o_t, o_{t+1}, \ldots, o_{t+w_s-1})$.

$$\tilde{\boldsymbol{A}} = \text{IDM}(\tau^{obs}_{t:t+w_s}), \qquad \boldsymbol{T}^s, \boldsymbol{T}^\ell = \phi(\tau^{obs}_{t:t+w_s}, \rho)$$

$$\boldsymbol{H} = \mathcal{T}\Big(\Big[\boldsymbol{T}^\ell,\ \boldsymbol{T}^s_1, \tilde{\boldsymbol{A}}_1,\ \ldots,\ \boldsymbol{T}^s_{w_s}, \tilde{\boldsymbol{A}}_{w_s}\Big]\Big)$$

$$\mathcal{L} = -\sum_{j=0}^{w_s} \tilde{\boldsymbol{A}}_j^\top \log \text{softmax}(\boldsymbol{W}_o \boldsymbol{H}_{w_l+2j+1})\,.$$

Algorithm 2 Inference

Require: Policy π, trajectories $\tau_{1\sim N}$, memory banks $\mathcal{M}_{1\sim N}$, classifier $\mathcal{K}$, memory lengths w_s, w_l, threshold Δ, environments $\mathcal{E}_{1\sim N}$

$C_{1\sim N} \leftarrow 0$
while $\max_n C_n < \Delta$ **do**
  for $n = 1$ to N **do**
    $\rho \leftarrow \mathcal{M}_{n,\,-w_l:}$
    $a \sim \pi(\cdot \mid \tau_{n,\,-w_s:},\ \rho)$
    $o \leftarrow \mathcal{E}_n(a)$
    $\tau_n \leftarrow \tau_n \cup \{(o, a)\}$
    if $\mathcal{K}(\tau_n,\ |\tau_n|)$ **then**
      $\mathcal{M}_n \leftarrow \mathcal{M}_n \cup \{o\}$
      $C_n \leftarrow 0$
    end if
    $C_n \leftarrow C_n + 1$
  end for
end while
return $\tau_{1\sim N}$, $\mathcal{M}_{1\sim N}$

First, we use the IDM to create pseudo-actions for each observation in the sample $\tau^{obs}_{t:t+w_s}$, which are encoded as one-hot vectors. These actions are used for both teacher-forcing and output labels while training π. ϕ is used to tokenize short-term ($\tau^{obs}_{t:t+w_s}$) and long-term (ρ) observations to create $\boldsymbol{T}^s$ and $\boldsymbol{T}^\ell$, respectively. The resulting tokens are concatenated into a unified sequence where the long-term memories $\boldsymbol{T}^\ell$ act as a prefix to the interleaved short-term observations and pseudo-actions. This combined sequence is passed through the causal transformer $\mathcal{T}$, allowing the model to attend to important memories while predicting local actions. We use linear layers to project $\boldsymbol{T}^s, \boldsymbol{T}^\ell$ and $\tilde{\boldsymbol{A}}$ into the hidden dimension of $\mathcal{T}$. The output layer $\boldsymbol{W}_o$ maps the hidden output at the observation token position of the transformer $\boldsymbol{H}$ to the action classes. The policy π is

trained using $\mathcal{L}$, the cross-entropy loss between predictions generated by ASH and pseudo-actions generated by our IDM.

Inference. Algorithm 2 describes ASH's inference loop. Each agent n maintains a per-agent stuck timer C_n, initialized to 0, that counts how many consecutive steps have elapsed since the agent last observed a new key moment. At each step, agent n reads the w_l most recent key moments from its memory bank $\mathcal{M}_n$ as ρ (long-memory), combines them with the last w_s observation-action pairs in τ_n (short-memory), and samples an action from π. After executing the action, the agent appends the resulting observation-action pair to τ_n. The classifier $\mathcal{K}$ determines whether the observation is a new key moment. If so, the observation is appended to $\mathcal{M}_n$ and the timer C_n is reset to zero; otherwise C_n is incremented. The loop runs until any agent's timer exceeds Δ, at which point ASH treats that agent as stuck, halts inference for all agents, and bootstraps their shared policy. After the bootstrap, agents resume inference from their current observation.

Retrieval. When an agent gets stuck, ASH needs training data targeted at its *current roadblock*. To find this data, ASH retrieves a dataset $\mathcal{D}^R$ from internet video $\mathcal{D}^I$ by selecting videos that are similar to the agent's current observations. We index our internet dataset using DINOv2 [43] by embedding video frames sampled at 2 second intervals. Each internet video is represented as a sequence of L2-normalized visual embeddings, forming a matrix $\boldsymbol{E} \in \mathbb{R}^{e \times d}$, where e denotes the number of sampled frames and d is the embedding dimension. For each agent trajectory τ_n, we similarly compute a trajectory embedding matrix $\boldsymbol{Q} \in \mathbb{R}^{q \times d}$. For each internet video τ^{obs}, we compute a similarity matrix $\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{E}^\top$. Retrieval searches for the most similar contiguous segment of length w_r in each internet video. We score each temporal window $\boldsymbol{S}_{:,\, t:t+w_r}$ by greedy one-to-one matching — each internet video frame in the window claims its highest-similarity trajectory frame, and each trajectory frame is consumed at most once. An internet video's score is its best temporal window score. We retrieve the top-k videos per agent trajectory and union them into $\mathcal{D}^R$. This matching scheme is compared to simpler baselines in Section 4.4.

Algorithm 3 Retrieval

Require: Trajectories $\tau_{1\sim N}$, indexed corpus $\mathcal{D}^I$, window w_r, number of videos retrieved k, steps since last retrieval δ

$\mathcal{D}^R \leftarrow \emptyset$
for $n = 1$ to N **do**
  $\boldsymbol{Q} \leftarrow \textsc{Embed}(\tau_{n,-\delta:})$
  scores $\leftarrow \emptyset$
  for $\tau^{\text{obs}} \in \mathcal{D}^I$ **do**
    $\boldsymbol{E} \leftarrow \textsc{Embed}(\tau^{\text{obs}})$
    $e \leftarrow \textsc{Rows}(\boldsymbol{E})$
    $\boldsymbol{S} \leftarrow \boldsymbol{Q}\boldsymbol{E}^\top$
    $z^\star \leftarrow -\infty$
    for $t \in \{1, \ldots, e - w_r + 1\}$ **do**
      $s \leftarrow \boldsymbol{S}_{:,\, t:t+w_r}$
      $z \leftarrow 0$
      for $j = 1$ to w_r **do**
        $r^\star \leftarrow \arg\max_r s_{r,j}$
        $z \leftarrow z + s_{r^\star,j}$
        $s_{r^\star,:} \leftarrow -\infty$
      end for
      $z^\star \leftarrow \max(z^\star, z)$
    end for
    scores $\leftarrow$ scores $\cup \{(\tau^{\text{obs}}, z^\star)\}$
  end for
  $\mathcal{D}^R \leftarrow \mathcal{D}^R \cup \{\text{top-k(scores)}\}$
end for
return $\mathcal{D}^R$

Discovering key moments. $\mathcal{K}$ is built by applying the HDBSCAN [40] clustering algorithm over the embeddings of $\mathcal{D}^R$, and all previously clustered embeddings. We further filter the resulting clusters by requiring each cluster contain observations from a minimum of $c_{\min}$ distinct trajectories in $\mathcal{D}^R$. This ensures that the clusters capture key visual moments that are commonly occurring between many trajectories. Each individual cluster therefore aims to be a *single key moment* in the environment. Details on cluster quality can be found in Section 4.5.

Key moments to long-term memory. $\mathcal{K}(\tau^{obs}, t)$ is a binary classifier: it returns *true* if the observation at timestep t is a *new* key moment—one whose corresponding cluster has not yet been matched by any earlier observation in τ^{obs}—and *false* otherwise. This is done by embedding τ^{obs}_t using DINOv2 [43] and applying HDBSCAN's approximate predict function to determine if it belongs to a cluster. These new key moments serve a dual purpose: firstly, they are appended to an agent's memory bank $\mathcal{M}_n$, and the last w_l entries are used as long-term memory for the agent. Secondly, they are used for our dynamic bootstrapping algorithm, where a bootstrap is triggered when an agent does not encounter a new key moment for Δ steps.

Bootstrapping. Each bootstrap (Algorithm 4) updates $\mathcal{K}$, the IDM and π. First, $\mathcal{K}$ is updated using the newly retrieved $\mathcal{D}^R$. For IDM data, ASH uses the observations from $\tau_{1\sim N}$ as IDM inputs, and the corresponding actions as labels. To ensure coverage of new environmental dynamics, we supplement the IDM training dataset with samples obtained from the environment using a random policy.

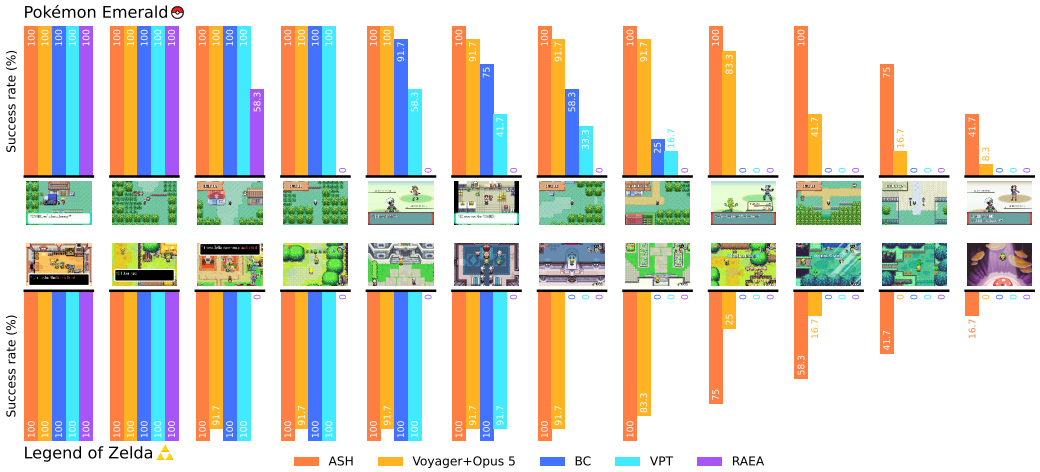


Figure 2: Agent progress in **Pokémon Emerald** (top) and **Legend of Zelda** (bottom), measured using milestone completion rates. ASH is able to adapt and continue progressing throughout the 8-hour gameplay period. While all methods are able to complete early milestones, only ASH can adapt to new areas, objectives, and mechanics. See Appendix E.1 for standard deviation.

ASH uses $\mathcal{K}$ and the IDM to update the policy π. Since τ^{obs} is internet video, it only contains a sequence of observations, not actions. Therefore, we use the IDM to create pseudo-actions for each sample from τ^{obs}. Then, $\mathcal{K}(\tau^{obs}, k)$ is used to construct memories (ρ) from timesteps $k < t$—before the start of the sample—which are used for training π's long memory. The resulting sample, consisting of w_s observation-action pairs and w_l long-term memories, is added to π's training dataset $\mathcal{D}^\pi$. π is then trained using $\mathcal{D}^\pi$. For both π and the IDM, we use 10% of samples as a held-out set to prevent overfitting. Once π has been bootstrapped, ASH resumes agent inference. Through this self-improvement loop, ASH hones its policy—adapting to new aspects of the environment and refining the skills it has acquired.

Algorithm 4 Bootstrapping

Require: Agent trajectories $\tau_{1\sim N}$, retrieved dataset $\mathcal{D}^R$, memory lengths w_s, w_l

Update $\mathcal{K}$ using $\mathcal{D}^R$
Update IDM on $\tau_{1\sim N}$
$\mathcal{D}^\pi \leftarrow \emptyset$
for $\tau^{obs} \in \mathcal{D}^R$ **do**
  for $t = 1$ to $|\tau^{obs}| - w_s + 1$ step w_s **do**
    $\tilde{\boldsymbol{A}} \leftarrow \text{IDM}(\tau^{obs}_{t:t+w_s})$
    $\bar{\rho} \leftarrow [\tau^{obs}_i : i < t, \mathcal{K}(\tau^{obs}, i)]$
    $\mathcal{D}^\pi \leftarrow \mathcal{D}^\pi \cup \{(\tau^{obs}_{t:t+w_s}, \tilde{\boldsymbol{A}}, \bar{\rho}_{-w_l:})\}$
  end for
end for
Update π on $\mathcal{D}^\pi$
return π, IDM, $\mathcal{K}$

4 Experiments

We evaluate ASH on two complementary long-horizon environments: Pokémon Emerald, a strategic turn-based RPG with a deep skill tree and dozens of hours of content, and *The Legend of Zelda: The Minish Cap*, an action-adventure game that demands real-time dexterity and puzzle solving. Both span millions of distinct environment observations. We provide additional background on our environments in Appendix D. Through our experiments, we investigate the following:

- How does ASH compare to other adaptive and non-adaptive baselines? (Section 4.1)
- What effect do long-term memory and dynamic bootstrapping have on ASH's policy? (Section 4.2)
- What does bootstrapping do to the IDM? (Section 4.3)
- Does ASH retrieve relevant, high-quality examples from internet data? (Section 4.4)
- Does ASH's long-term memory improve long-horizon decision-making? (Section 4.5)
- Can ASH's final policy replay the environment offline? (Section 4.6)

Milestones. We evaluate our agents using *milestones*, which are human-defined markers of progress in our environments. We derive our milestones from online walkthroughs [44, 45], descriptions of each milestone can be found in Appendix C.

Settings and implementation details. In our experiments, ASH uses $N = 4$ agents. We use a timestep of 0.25 seconds to sample observations from the environment and input actions. We set

w_s and w_l to 64 and 10, respectively. At each timestep, we use w_s previous observations, with a sampling interval of 0.5 seconds, as an agent's short-term memory. This allows the agent to maintain a short-term memory of 32 seconds, while interacting with the environment every 0.25 seconds. For retrieval, we set $k = 100$ and $w_r = 540$. We set our HDBSCAN [46] hyperparameter, $c_{\min}$, to 30. We scrape our internet datasets ($\mathcal{D}^I$) from YouTube using a list of keywords (Appendix I). Our dataset for Pokémon Emerald contains approximately $22,000$ unique videos, whereas our Legend of Zelda dataset contains $17,000$ videos. For dynamic bootstrapping, we set Δ to 4800 which is 20 minutes of real-time (the median video length in $\mathcal{D}^I$). ASH contains 1.2 billion parameters, 100 million for vision encoder ϕ and 1.1 billion for the transformer $\mathcal{T}$. We use fp32 precision for ϕ and bf16 for $\mathcal{T}$. We find that initializing each run from random weights is inefficient, so we use checkpoints trained for 3 and 9 GPU hours to initialize the IDM and π. The data for these short training runs is obtained from the environment's starting state using a uniformly random policy over the action space (for the IDM) and Algorithm 3 (for π). We detail architectural, inference, and training hyperparameters for ASH in Appendix F.

Baselines. We compare ASH with the following baselines. **Voyager+Opus 5** [32]: We scaffold Opus 5 [47] with a self-editable code skill library. Voyager+Opus 5 interacts with the game by sending button press actions and receiving frames as observations. **VPT** [4]: VPT requires an existing IDM, so we use the strongest IDM checkpoint produced by ASH, as measured by performance on a validation set; see Section 4.3. We find that direct offline training on $\mathcal{D}^I$ creates a weak policy that does not progress past the first milestones, so we manually curate 400 hours of high-quality training data. **Offline BC**: ASH's architecture without long-term memory or bootstrapping, trained using the same IDM and data as VPT. **Retrieval-augmented Embodied Agent (RAEA)** [9]: We follow the recipe of Zhu et al. [9] and augment offline BC with cross-attention on retrieved examples. We split our manually curated data into 350 hours of training data and 50 hours of retrievable examples, labeled by our IDM. For both inference and training, we run ASH retrieval every 32 seconds with $w_r = 16$ and $k = 1$ but return the best matching segment, rather than the whole video. Therefore, at every step, RAEA is conditioned on 32 seconds of context and a 32 second policy example. For compute parity, we allocate up to 288 GPU hours (4 days on 4×H100 SXM 80GB GPUs) total for each method, or until overfitting is detected on held-out data.

4.1 Performance on Milestones

For our baselines, we train one policy for each environment and perform 12 runs each. Since ASH infers multiple agents at once, we perform 3 training runs per environment with $N = 4$. This gives us 12 trajectories per-method. We report average milestone completion rate in Figure 2. We find that:

Self-improvement is necessary for sustained progression. Over the 8-hour evaluation, ASH reaches milestone 12 in both environments, while no baseline exceeds milestone 8 in Pokémon or 6 in Zelda. VPT and offline BC plateau once the games introduce dynamics that are underrepresented in their training distribution (for example, seldom encountered random events, or areas that are only briefly visited by players). RAEA, designed for short-horizon tasks, struggles in our long-horizon setting since it retrieves a different policy example every 32 seconds, causing instability. Finally, Voyager+Opus 5 often loses track of its current objective, causing it to get stuck attempting the wrong objectives.



Figure 3: Outside (top) vs. inside (bottom) of the Zelda castle: offline policies collapse on this dynamics shift; ASH bootstraps and continues.

ASH adapts to changing dynamics. The Zelda castle entrance is the cleanest example of a distribution-shift moment (Figure 3): the visual dynamics change abruptly, and VPT and offline BC, frozen at training time, devolve into random button-mashing and never exit the castle. ASH's policy also falters, but after a bootstrap is able to continue making progress. The same pattern appears in Pokémon the moment the agent leaves the starting town and encounters the Pokémon battling interface.

Bootstrapping evolves new skills. ASH's initial checkpoint does not exhibit any ability to battle with Pokémon, and therefore gets stuck on the second milestone (which requires battling). However, through bootstrapping, ASH becomes an adept battler. After hour 4, ASH wins 87% of the Pokémon battles it engages in. In the Legend of Zelda, ASH kills enemies

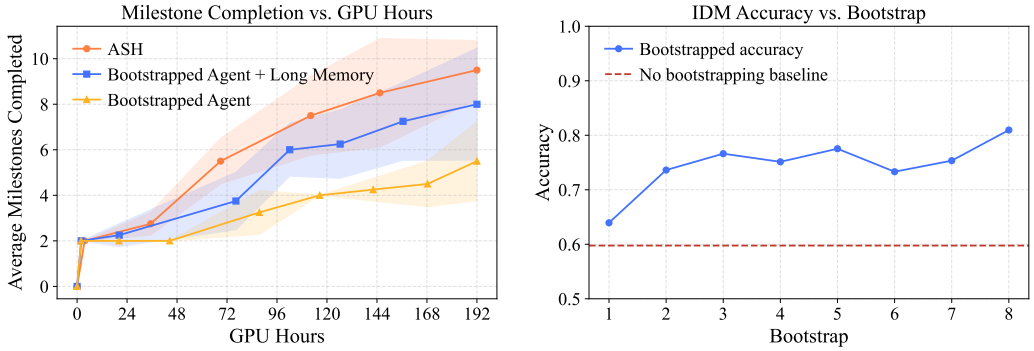


Figure 4: **(Left)** Component ablation on Pokémon Emerald: each addition (long-term memory, dynamic bootstrapping) yields a clear gain in milestones completed per GPU hour of online training. Shaded regions are one standard deviation over 4 trajectories per method. **(Right)** IDM accuracy across bootstraps, evaluated on a test set. The dashed line is the pre-bootstrap initialization checkpoint.

with its sword in 75% of trajectories. This capability is absent from its initial checkpoint and is not observed in any of our baselines. We provide examples of ASH evolving skills in Appendix A.

Across both environments, gameplay has a horizon beyond which a fixed policy struggles to succeed: the environments continually introduce new mechanics, regions, and visual contexts. Methods without an update mechanism plateau at that horizon. ASH's self-improvement loop allows it to move beyond this horizon.

4.2 Component Ablations

In Figure 4 (left), we ablate across the components of ASH and measure their contribution to milestone completion in Pokémon Emerald. **Bootstrapped Agent** removes both the long-term memory and dynamic bootstrapping, fixing the bootstrap interval to Δ. **Bootstrapped Agent + Long Memory** reintroduces the agent's long-term memory but keeps the fixed bootstrapping schedule. **ASH** adds dynamic bootstrapping. Because each variant uses different amounts of compute for inference and training, we plot milestone completion against *total GPU hours of online training*. Each method is averaged over 4 trajectories, and the shaded regions show the standard deviation. We find that the introduction of each component of ASH results in a clear increase in performance. By the end of the evaluation, ASH's long-memory component contributes 2.5 average milestones. Finally, dynamic bootstrapping allows ASH to deploy compute more efficiently by only training when it gets stuck, contributing an additional 1.5 milestones.

4.3 IDM Adaptation

As an agent progresses through the environment, new visual dynamics emerge, and the IDM must correctly label these new dynamics for policy bootstrapping to be effective. To measure how well the IDM captures evolving dynamics, we take the checkpoint from each bootstrap of an ASH playthrough of Pokémon Emerald, and evaluate it on a test set of trajectories from a separate ASH playthrough. We report the per-bootstrap IDM accuracy in Figure 4 (right). We find that the IDM becomes better at labeling environment dynamics across bootstraps. The initialization IDM checkpoint (no bootstrapping baseline) has just 60% accuracy on our test set, whereas our final IDM checkpoint achieves 81%. We limit our analysis to the four (up, down, left, right) movement action classes since some actions like the `A` and `B` buttons are often visually indistinguishable.

We find that IDM bootstrapping is essential for progression. In Pokémon Emerald, training the VPT and BC baselines using our curated dataset and the no bootstrapping baseline IDM yielded 0% completion rates on milestones 3 and above.

4.4 Retrieval Quality

For bootstrapping to be effective, it is crucial that ASH retrieval creates a *relevant and low-noise* training dataset $\mathcal{D}^R$ from our corpus of YouTube data, $\mathcal{D}^I$. To measure the effectiveness of ASH retrieval, we conduct a blind evaluation. For each query trajectory, an annotator sees the videos retrieved by each method and performs two tasks. First, they label each retrieved video as relevant

training data (yes/no). Second, given one video per method, they rank the four videos using the following criteria, in priority order:

1. Does the retrieved video correspond to the same part of the game as the trajectory?
2. Is the retrieved video visually similar to the trajectory?
3. Does the retrieved video contain future milestones?

Using the above criteria, the annotators rank the videos from 1 (most relevant) to 4 (least relevant). For each method considered, we use 10 trajectories sampled from bootstrapping runs of Pokémon Emerald (Section 4.1). We judge the top 20 videos per trajectory, for a total of 800 videos judged.

We compare **ASH retrieval** with three baselines. **Random** selects videos uniformly from the entire dataset. **All-to-All** ranks videos by the average similarity between all trajectory and internet video embeddings. **Max Single Match** replaces the sliding window score of ASH retrieval with the maximum score of any given trajectory embedding with an internet video embedding. We report our findings in Table 1. As Pokémon is a long-horizon game, we find a randomly retrieved video has only an 8% chance of being relevant training data, highlighting the challenges of naively training a policy over $\mathcal{D}^I$. By contrast, ASH retrieves relevant videos 94% of the time, achieving an average rank of 1.35 between methods. We find that Max Single Match occasionally makes spurious matches, for example matching a black screen between videos, and All-to-All matching seems to lose all signal, consistently matching to the same internet video, irrespective of the query.

	ASH	MSM	All2All	Rand
Binary Rel. ↑	**94%**	69%	3%	8%
Ave. Rank ↓	**1.35**	1.81	3.63	3.29

Table 1: Annotator judgments of ASH retrieval and baselines: binary relevance (yes/no) and average rank across methods (lower is better).

4.5 Long-term Memory Analysis

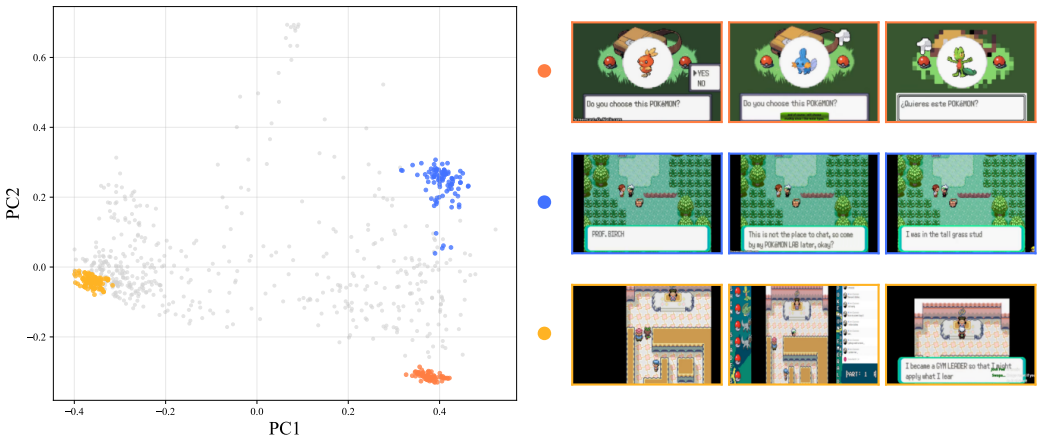


Figure 5: Visualization of 3 HDBSCAN [40] clusters, as well as 500 uniformly sampled outlier points reduced to 2 dimensions via principal component analysis. Each of these clusters represents a key moment in Pokémon Emerald; choosing a starter Pokémon (orange), saving the professor (blue) and challenging a gym leader (yellow). Grey dots are outliers that are not assigned to a cluster.

The effectiveness of ASH's long-term memory depends on the quality of the clusters produced by HDBSCAN [40]. A cluster should be composed of data from a single *key moment* — an observation that can be used to contextualize a player's actions within the broader trajectory. Examples of key moments include reaching a milestone, selecting your starter Pokémon, or battling a particular enemy.

To evaluate how well HDBSCAN [40] clusters these key moments, we evaluated 400 clusters sampled uniformly at random from a pool of 2,200 clusters generated by ASH across all bootstraps during an eight-hour run in Pokémon Emerald. For each cluster, annotators were shown five of its constituent internet video frames and answered two yes/no questions:

1. Are all frames *key moments*?
2. Do all frames correspond to the *same* key moment?

If the answer to both questions is yes, we consider the cluster to be a key moment. For comparison, we also evaluate 400 randomly sampled frames from the agent's trajectory. We find that 48% of clusters identified by HDBSCAN [40] correspond to key moments, compared to just 0.5% of randomly sampled frames.

Long-term memory contents	Avg. milestones
Random trajectory frames	6.00
Key moments (ours)	**7.75**

Table 2: Average milestones achieved for key moments versus random frames as the contents of long-term memory, in Pokémon Emerald. Each value is an average over 4 runs of 4 hours.

To validate the usefulness of these HDBSCAN identified key moments, we replace the w_l key moments with w_l frames sampled uniformly at random from the agent's trajectory during inference. Since we cannot perform dynamic bootstrapping with randomly sampled long memory, we compare against Bootstrapped Agent + Long Memory from Section 4.2 averaging over 4 trajectories of 4 hours each in Pokémon Emerald. We report results in Table 2.

We find that replacing key moments with random frames reduces average achieved milestones by 1.75 over the same evaluation period. The benefit of long-term memory therefore comes from *which* observations ASH keeps, indicating the usefulness of HDBSCAN identified key moments.

4.6 Offline Replay

Catastrophic forgetting is a phenomenon in lifelong learning where an agent will forget previously known skills and knowledge when its policy is updated [48]. The result is an agent that can progress through the latter stages of an environment but can no longer accomplish early milestones. We examine whether ASH's final policy is able to play through the environment from the beginning without bootstrapping, or if it exhibits catastrophic forgetting.

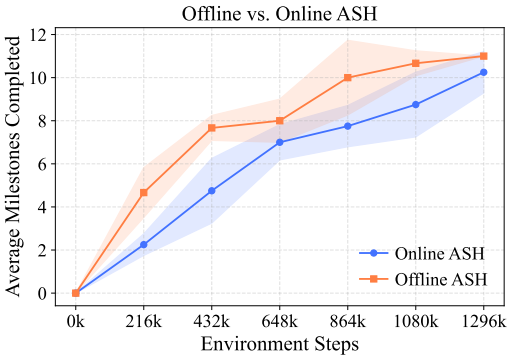


Figure 6: Offline replay of the final ASH checkpoint vs. the original online run.

We compare the final ASH checkpoint (offline ASH) with the online run that created it (online ASH). Both methods are run for 1.3 million environment steps (6 hours) in Pokémon Emerald (Figure 6). Results are averaged over 4 trajectories and the shaded regions show the standard deviation. We find that ASH does not exhibit catastrophic forgetting. As the offline ASH checkpoint has already learned (and remembers) the environment, it progresses through milestones faster than the online learning agent. This is analogous to a player replaying the environment, who, due to accumulated expertise, is faster than the beginner.

5 Limitations and Conclusion

We introduced ASH, an agent that self-improves by retrieving and learning from relevant internet video, without reward engineering or human annotation. Across Pokémon Emerald and The Legend of Zelda, ASH continues to adapt beyond the point where behavioral cloning, retrieval-augmented, and foundation-model baselines stall, developing new skills such as battling, catching Pokémon, and using a sword and shield. These results show that self-improving agents are a scalable path toward open-ended, long-horizon learning.

Although ASH improves over fixed policies in long-horizon settings, it depends on the availability of useful internet video and may be harder to apply in environments where demonstrations are scarce. It also relies on pseudo-actions from an Inverse Dynamics Model, which will struggle to label actions that are difficult to infer from pixels. Furthermore, as ASH was evaluated in 2D environments with discrete action spaces, it remains unclear how well ASH generalizes to 3D environments, or to those with continuous action spaces. Finally, since our approach is not specific to Pokémon or Zelda, we hope ASH can inspire future work in creating long-horizon agents in other domains and environments.

Acknowledgments and Disclosure of Funding

This research was supported by the New Beginnings program of the National Research Council of Canada (INBR6-001156), as well as the Natural Sciences and Engineering Research Council of Canada (NSERC) through Victor Zhong's Discovery Grant, Research Tools and Instruments grant, and by the Canada Foundation for Innovation (CFI) John R. Evans Leaders Fund with matching support from the Ontario Research Fund. Furthermore, this research was enabled by computing resources provided by Compute Ontario and the Digital Research Alliance of Canada.

References

[1] Wei-Chieh Huang, Weizhi Zhang, Yueqing Liang, Yuanchen Bei, Yankai Chen, Tao Feng, Xinyu Pan, Zhen Tan, Yu Wang, Tianxin Wei, Shanglin Wu, Ruiyao Xu, Liangwei Yang, Rui Yang, Wooseong Yang, Chin-Yuan Yeh, Hanrong Zhang, Haozhen Zhang, Siqi Zhu, Henry Peng Zou, Wanjia Zhao, Song Wang, Wujiang Xu, Zixuan Ke, Zheng Hui, Dawei Li, Yaozu Wu, Langzhou He, Chen Wang, Xiongxiao Xu, Baixiang Huang, Juntao Tan, Shelby Heinecke, Huan Wang, Caiming Xiong, Ahmed A. Metwally, Jun Yan, Chen-Yu Lee, Hanqing Zeng, Yinglong Xia, Xiaokai Wei, Ali Payani, Yu Wang, Haitong Ma, Wenya Wang, Chenguang Wang, Yu Zhang, Xin Wang, Yongfeng Zhang, Jiaxuan You, Hanghang Tong, Xiao Luo, Xue Liu, Yizhou Sun, Wei Wang, Julian McAuley, James Zou, Jiawei Han, Philip S. Yu, and Kai Shu. Rethinking memory mechanisms of foundation agents in the second half: A survey, 2026. URL https://arxiv.org/abs/2602.06052.

[2] Yafei Hu, Quanting Xie, Vidhi Jain, Jonathan Francis, Jay Patrikar, Nikhil Keetha, Seungchan Kim, Yaqi Xie, Tianyi Zhang, Hao-Shu Fang, Shibo Zhao, Shayegan Omidshafiei, Dong-Ki Kim, Ali akbar Agha-mohammadi, Katia Sycara, Matthew Johnson-Roberson, Dhruv Batra, Xiaolong Wang, Sebastian Scherer, Chen Wang, Zsolt Kira, Fei Xia, and Yonatan Bisk. Toward general-purpose robots via foundation models: A survey and meta-analysis, 2024. URL https://arxiv.org/abs/2312.08782.

[3] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4950–4957. International Joint Conferences on Artificial Intelligence Organization, 7 2018. doi: 10.24963/ijcai.2018/687. URL https://doi.org/10.24963/ijcai.2018/687.

[4] Bowen Baker, Ilge Akkaya, Peter Zhokhov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos, 2022. URL https://arxiv.org/abs/2206.11795.

[5] Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/ross10a.html.

[6] Shalev Lifshitz, Keiran Paster, Harris Chan, Jimmy Ba, and Sheila A. McIlraith. STEVE-1: A generative model for text-to-behavior in minecraft. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=YkBDJWerKg.

[7] Marco Pleines, Daniel Addis, David Rubinstein, Frank Zimmer, Mike Preuss, and Peter Whidden. Pokemon red via reinforcement learning, 2025.

[8] Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models, 2025. URL https://arxiv.org/abs/2509.24527.

[9] Yichen Zhu, Zhicai Ou, Xiaofeng Mou, and Jian Tang. Retrieval-Augmented Embodied Agents . In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17985–17995, Los Alamitos, CA, USA, June 2024. IEEE Computer Society. doi: 10.1109/CVPR52733.2024.01703. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.01703.

[10] Qwen Team. Qwen3.5: Accelerating productivity with native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[11] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. Optimus-2: Multimodal minecraft agent with goal-observation-action conditioned policy. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2025.

[12] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation, 2018. URL https://arxiv.org/abs/1805.01954.

[13] Ashley Edwards, Himanshu Sahni, Yannick Schroecker, and Charles Isbell. Imitating latent policies from observation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1755–1763. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/edwards19a.html.

[14] Dominik Schmidt and Minqi Jiang. Learning to act without actions. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[15] Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Maria Elisabeth Bechtle, Feryal Behbahani, Stephanie C.Y. Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando De Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 4603–4623. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/bruce24a.html.

[16] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

[17] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[18] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

[19] Peter Conway Humphreys, Arthur Guez, Olivier Tieleman, Laurent Sifre, Theophane Weber, and Timothy P Lillicrap. Large-scale retrieval for reinforcement learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Ya9lATuQ3gg.

[20] Anirudh Goyal, Abram Friesen, Andrea Banino, Theophane Weber, Nan Rosemary Ke, Adrià Puigdomènech Badia, Arthur Guez, Mehdi Mirza, Peter C Humphreys, Ksenia Konyushova, Michal Valko, Simon Osindero, Timothy Lillicrap, Nicolas Heess, and Charles Blundell. Retrieval-augmented reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 7740–7765. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/goyal22a.html.

[21] Danyang Zhang, Lu Chen, Situo Zhang, Hongshen Xu, Zihan Zhao, and Kai Yu. Large language models are semi-parametric reinforcement learning agents. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=ZcJa1R6j3v.

[22] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STar: Bootstrapping reasoning with reasoning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_3ELRdg2sgI.

[23] Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling, 2023. URL https://arxiv.org/abs/2308.08998.

[24] Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 5366–5376, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[25] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017. URL https://arxiv.org/abs/1712.01815.

[26] Michael Ahn, Debidatta Dwibedi, Chelsea Finn, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Karol Hausman, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Sean Kirmani, Isabel Leal, Edward Lee, Sergey Levine, Yao Lu, Isabel Leal, Sharath Maddineni, Kanishka Rao, Dorsa Sadigh, Pannag Sanketi, Pierre Sermanet, Quan Vuong, Stefan Welker, Fei Xia, Ted Xiao, Peng Xu, Steve Xu, and Zhuo Xu. Autort: Embodied foundation models for large scale orchestration of robotic agents, 2024.

[27] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=rc8o_j8I8PX.

[28] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021. URL https://arxiv.org/abs/2106.01345.

[29] Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TrjbxzRcnf-.

[30] Aydar Bulatov, Yuri Kuratov, and Mikhail Burtsev. Recurrent memory transformer. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Uynr3iPhksa.

[31] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego De Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/borgeaud22a.html.

[32] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

[33] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

[34] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Joseph Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, L. Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander Sasha Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom Le Paine, Caglar Gulcehre, Ziyun Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy P. Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575:350 – 354, 2019. URL https://api.semanticscholar.org/CorpusID:204972004.

[35] OpenAI, :, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d. O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning, 2019. URL https://arxiv.org/abs/1912.06680.

[36] Seth Karten, Jake Grigsby, Stephanie Milani, Kiran Vodrahalli, Amy Zhang, Fei Fang, Yuke Zhu, and Chi Jin. The pokeagent challenge: Competitive and long-context learning at scale. In *NeurIPS Competition Track*, April 2025.

[37] ClaudePlaysPokemon. ClaudePlaysPokemon. https://www.twitch.tv/claudeplayspokemon, 2026. [Accessed 01-05-2026].

[38] SIMA team, Adrian Bolton, Alexander Lerchner, Alexandra Cordell, Alexandre Moufarek, Andrew Bolt, Andrew Lampinen, Anna Mitenkova, Arne Olav Hallingstad, Bojan Vujatovic, Bonnie Li, Cong Lu, Daan Wierstra, Daniel P. Sawyer, Daniel Slater, David Reichert, Davide Vercelli, Demis Hassabis, Drew A. Hudson, Duncan Williams, Ed Hirst, Fabio Pardo, Felix Hill, Frederic Besse, Hannah Openshaw, Harris Chan, Hubert Soyer, Jane X. Wang, Jeff Clune, John Agapiou, John Reid, Joseph Marino, Junkyung Kim, Karol Gregor, Kaustubh Sridhar, Kay McKinney, Laura Kampis, Lei M. Zhang, Loic Matthey, Luyu Wang, Maria Abi Raad, Maria Loks-Thompson, Martin Engelcke, Matija Kecman, Matthew Jackson, Maxime Gazeau, Ollie Purkiss, Oscar Knagg, Peter Stys, Piermaria Mendolicchio, Raia Hadsell, Rosemary Ke, Ryan Faulkner, Sarah Chakera, Satinder Singh Baveja, Shane Legg, Sheleem Kashem, Tayfun Terzi, Thomas Keck, Tim Harley, Tim Scholtes, Tyson Roberts, Volodymyr Mnih, Yulan Liu, Zhengdong Wang, and Zoubin Ghahramani. Sima 2: A generalist embodied agent for virtual worlds, 2025. URL https://arxiv.org/abs/2512.04797.

[39] Eric Hambro, Roberta Raileanu, Danielle Rothermel, Vegard Mella, Tim Rocktäschel, Heinrich Küttler, and Naila Murray. Dungeons and data: A large-scale nethack dataset, 2023. URL https://arxiv.org/abs/2211.00539.

[40] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, pages 160–172, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-37456-2.

[41] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023. URL https://arxiv.org/abs/2303.15343.

[42] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

[43] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

[44] Bulbapedia. Walkthrough:pokémon emerald, Jun 2025. URL https://bulbapedia.bulbagarden.net/wiki/Walkthrough:Pok%C3%A9mon_Emerald.

[45] Zelda Dungeon. The minish cap walkthrough. https://www.zeldadungeon.net/the-minish-cap-walkthrough/, 2026. [Accessed 30-04-2026].

[46] Sebastian Raschka, Joshua Patterson, and Corey Nolet. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *arXiv preprint arXiv:2002.04803*, 2020.

[47] Anthropic. Claude opus 5 system card. Technical report, Anthropic, 2026. URL https://platform.claude.com/docs/en/resources/overview.

[48] Zhenyi Wang, Enneng Yang, Li Shen, and Heng Huang. A comprehensive survey of forgetting in deep learning beyond continual learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 47(3):1464–1483, March 2025. ISSN 0162-8828. doi: 10.1109/TPAMI.2024.3498346. URL https://doi.org/10.1109/TPAMI.2024.3498346.

[49] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

A ASH Playthrough Examples

Examples of how ASH uses **dynamic bootstrapping** and **long-memory** to overcome roadblocks.



Figure 7: **Dynamic bootstrapping example.** To complete milestone 2, the player must rescue the Professor from a wild Zigzagoon **(Panel 1)**. To accomplish this, the player must use their starter Pokémon to defeat the Zigzagoon in battle **(Panel 2)**. However, ASH's initial policy does not know how to use the battle interface to command their Pokémon. After being stuck for Δ steps (20 minutes), ASH **dynamically bootstraps**, and finds relevant examples of battling from $\mathcal{D}^I$ **(Panel 3)**. After updating its policy, ASH is able to attack and defeat the Zigzagoon **(Panels 4 & 5)**. After learning how to battle, ASH is **unstuck** and continues on its journey **(Panel 6)**.

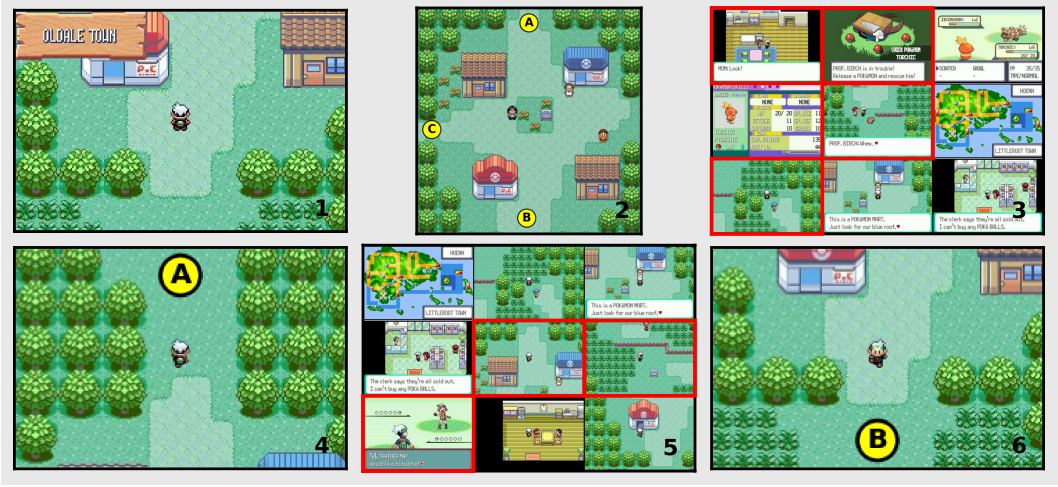


Figure 8: **Long-term memory example.** When the player arrives in Oldale town **(Panel 1)**, they are presented with 3 possible next paths. **Option `A`:** The player heads north to Route 103 to meet their rival, May. This is the correct choice if the player has just obtained their starter Pokémon from the Professor and been tasked with bringing May back to the lab. **Option `B`:** The player has already met May, and should head back to the lab to meet the professor. Or **Option `C`:** The player has obtained the Pokédex at the lab from May and the Professor, and should proceed to Route 102. However, **short-term memory is insufficient for this decision**, as short-term observations are nearly identical. However, from inspecting the agent's **long-term memory (Panel 3)**, it is clear that the player has just obtained their starter Pokémon, and therefore ASH chooses **Option** `A` unambiguously **(Panel 4)**. After meeting May, the player's long-term memory has been updated with their encounter **(Panel 5)**, so it knows to choose **Option** `B` and head back to the lab **(Panel 6)**.

B ASH

Algorithm 5 ASH

Require: Number of agents N, short-term memory length w_s, long-term memory length w_l, max steps without new key moment Δ, retrieval match length w_r, number of videos retrieved k, internet dataset $\mathcal{D}^I$

Step 0: Initialization

$\pi \leftarrow$ Initialize ASH policy
$\mathcal{K} \leftarrow$ Initialize new key moment classifier
IDM $\leftarrow$ Initialize IDM
$\mathcal{E}_{1\sim N} \leftarrow$ Initialize environments ▷ All agents start at the beginning of the environment.
$\tau_{1\sim N} \leftarrow \emptyset$ ▷ Per agent environment trajectories.
$\mathcal{M}_{1\sim N} \leftarrow \emptyset$ ▷ Key moment memory per agent.

Step 1: Inference

$\delta \leftarrow 0$ ▷ Steps since last retrieval.
$C_{1\sim N} \leftarrow 0$ ▷ Per-agent stuck counters
while $\max_n C_n < \Delta$ **do**
  $\delta \leftarrow \delta + 1$
  for $n = 1$ to N **do**
    $\rho \leftarrow \mathcal{M}_{n,\,-w_l:}$ ▷ Retrieve w_l most recent long-memories from the agent's memory bank.
    $a \sim \pi(\cdot \mid \tau_{n,\,-w_s:},\, \rho)$ ▷ Predict action using short and long-term memory.
    $o \leftarrow \mathcal{E}_n(a)$
    $\tau_n \leftarrow \tau_n \cup \{(o, a)\}$
    if $\mathcal{K}(\tau_n,\, |\tau_n|)$ **then** ▷ Add observation to memory bank if it is a *new key moment*.
      $\mathcal{M}_n \leftarrow \mathcal{M}_n \cup \{o\}$
      $C_n \leftarrow 0$ ▷ Reset steps since last new key moment counter.
    end if
    $C_n \leftarrow C_n + 1$
  end for
end while

Step 2: Retrieval

$\mathcal{D}^R \leftarrow \emptyset$
for $n = 1$ to N **do**
  $\boldsymbol{Q} \leftarrow \textsc{Embed}(\tau_{n,-\delta:})$
  scores $\leftarrow \emptyset$
  for $\tau^{\text{obs}} \in \mathcal{D}^I$ **do**
    $\boldsymbol{E} \leftarrow \textsc{Embed}(\tau^{\text{obs}})$ ▷ $\boldsymbol{E}$ is cached for efficiency.
    $e \leftarrow \textsc{Rows}(\boldsymbol{E})$
    $\boldsymbol{S} \leftarrow \boldsymbol{Q}\boldsymbol{E}^\top$
    $z^\star \leftarrow -\infty$
    for $t \in \{1, \ldots, e - w_r + 1\}$ **do** ▷ Compute scores over windows.
      $s \leftarrow \boldsymbol{S}_{:,\,t:t+w_r}$
      $z \leftarrow 0$
      for $j = 1$ to w_r **do**
        $r^\star \leftarrow \arg\max_r s_{r,j}$
        $z \leftarrow z + s_{r^\star,j}$
        $s_{r^\star,:} \leftarrow -\infty$
      end for
      $z^\star \leftarrow \max(z^\star, z)$
    end for
    scores $\leftarrow$ scores $\cup \{(\tau^{\text{obs}}, z^\star)\}$
  end for
  $\mathcal{D}^R \leftarrow \mathcal{D}^R \cup \{\text{top-k(scores)}\}$ ▷ Unionize videos into a dataset.
end for

Step 3: Policy Bootstrapping

Update $\mathcal{K}$ using $\mathcal{D}^R$
Update IDM on $\tau_{1\sim N}$
$\mathcal{D}^\pi \leftarrow \emptyset$
for $\tau^{obs} \in \mathcal{D}^R$ **do**
  for $t = 1$ to $|\tau^{obs}| - w_s + 1$ step w_s **do**
    $\tilde{\boldsymbol{A}} \leftarrow \text{IDM}(\tau^{obs}_{t:t+w_s})$ ▷ Use IDM to create pseudo-actions for internet data item.
    $\bar{\rho} \leftarrow [\,\tau^{obs}_i \,:\, i < t,\, \mathcal{K}(\tau^{obs}, i)\,]$ ▷ Find key moments in τ^{obs}.
    $\mathcal{D}^\pi \leftarrow \mathcal{D}^\pi \cup \left\{\left(\tau^{obs}_{t:t+w_s},\, \tilde{\boldsymbol{A}},\, \bar{\rho}_{-w_l:}\right)\right\}$ ▷ Add sample to dataset.
  end for
end for
Update π on $\mathcal{D}^\pi$
`GoTo` **Step 1**

C Milestones

Milestone	Icon	Description
Home		Before setting off on their journey, the player must talk to their mother and set the clock in their home.
Save the professor		The player must obtain their starter Pokémon and save the professor from an encounter with a wild Pokémon.
Oldale Town		After progressing through Route 101, the player reaches the first town after their hometown. Oldale Town contains many important optional objectives such as healing your Pokémon at the Pokémon Center and obtaining items at the Poké Mart.
Route 103		Route 103 is the area above Oldale Town. This milestone is achieved when the player first enters the route.
Rival Battle		After meeting them on route 103, the player must defeat a trainer known as the "rival" in a Pokémon battle. This milestone is completed when the player successfully defeats their rival in a battle.
Obtain Pokédex		Upon defeating their rival, the player must backtrack to their hometown to obtain the Pokédex item from the professor. This milestone is completed when the player obtains the Pokédex.
Route 102		With the Pokédex in hand, the player can return to Oldale Town and continue on their journey through route 102. This milestone is achieved when the player first enters route 102.
Arrive in Petalburg City		After traveling through Route 102, the player arrives in Petalburg City, where they meet key characters such as the gym leader (their father) and progress the main story.
Defeat Team Aqua in Petalburg Woods		While passing through Petalburg Woods, the player encounters a Team Aqua member threatening a researcher. This milestone is completed when the player defeats the Team Aqua member in battle.
Route 104		Route 104 connects Petalburg Woods to Rustboro City. This milestone is achieved when the player first enters Route 104 after exiting the forest.
Rustboro City		After traveling through Route 104, the player arrives in Rustboro City, home to the first Gym challenge and several important locations such as the Trainer's School and Devon Corporation.
Defeat the Gym Leader		In Rustboro City, the player can battle the gym leader *Roxanne*. Gym leaders are powerful trainers who act as bosses that the trainer must defeat to progress. The milestone is marked as completed *after* the player wins the battle.

Table 3: Milestones, icons and corresponding descriptions for the Pokémon Emerald environment.

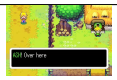
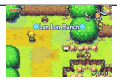
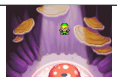
Milestone	Icon	Description
Obtain the sword from the smith		The player must speak to the smith to obtain a sword.
Meet with Zelda		After obtaining the sword, the player must meet up with Zelda to head to the festival. This milestone is marked as completed when the player 'groups up' with Zelda and she follows the player to the festival.
Obtain the small shield		At the festival, the player must win the small shield item as a prize in a contest. The milestone is completed when the player obtains the shield.
Reflect the seed projectile with the shield		After leaving the festival, the player encounters a *Deku Scrub* who is firing seeds and blocking the path. To progress, the player must use their shield to deflect a seed back at the Deku Scrub to stun it.
Enter Hyrule Castle		The player arrives at the entrance to Hyrule Castle. This milestone is achieved when the player reaches the castle exterior.
Talk with the king		The player enters the throne room and has an audience with the King of Hyrule.
Leave the castle		After speaking with the king, the player must leave the castle. This milestone is marked as completed once the player exits the castle. We allow the player to use any of the castle exits for this milestone.
Leave the palace grounds		This milestone is completed when the player leaves the palace grounds area.
Enter Lon Lon Ranch		To get to the Minish Forest, the player must go through the *Lon Lon Ranch* zone, this milestone is completed when the player first enters this new area.
Enter the Minish Woods		After navigating through Lon Lon Ranch, the player can enter the Minish Forest, an important location for the player's quest. This milestone is completed when the player enters the forest.
Obtain the Minish Cap		In the Minish Forest, the player obtains the *Minish Cap*. A new item that allows the player to shrink in size. When the player places the Minish Cap on their head, this milestone is completed.
Solve the Minish Cap Puzzle		Using the new shrinking game mechanic (from the Minish Cap), the player must solve a puzzle that requires using the Minish Cap to fit through narrow spaces (such as the interior of a log). This milestone is completed when the player finishes the puzzle.

Table 4: Milestones, icons and corresponding descriptions for the Legend of Zelda environment.

Milestone distribution in human walkthroughs. Our evaluations are sensitive to milestone choice. If our method made marginally more progress than the next-best baseline, and there were a disproportionate number of milestones in the slice of gameplay separating our method from the next-best, this would make our method look substantially better than it actually was. To evaluate this property of our milestones, we take 5 online walkthroughs of Pokemon Emerald and report the average time at which the players completed each milestone. We report results in Table 5.

#	Milestone	Avg. completion time (min)
1	Home	7.91
2	Save the professor	15.27
3	Oldale Town	22.28
4	Route 103	24.09
5	Rival Battle	26.41
6	Obtain Pokédex	28.72
7	Route 102	36.68
8	Arrive in Petalburg City	49.43
9	Defeat Team Aqua in Petalburg Woods	85.47
10	Route 104	89.74
11	Rustboro City	101.04
12	Defeat the Gym Leader	149.51

Table 5: Average completion time per milestone across 5 human walkthroughs of Pokémon Emerald.

We see that the milestones are more concentrated towards the beginning of the game, suggesting that differences between average milestones completed are more significant when the absolute number is higher.

D Background on Environments

Pokémon Emerald is a third-generation *Pokémon* role-playing game for the Game Boy Advance (GBA). The game follows a largely linear-but-explorable progression: the player travels between towns and routes, completes story events, captures and trains creatures ("Pokémon"), and defeats a sequence of gym leaders to earn badges, culminating in the Elite Four and Champion battles.

Gameplay consists of real-time overworld navigation (movement, interactions with NPCs, and menu-driven inventory/team management) interleaved with turn-based battles. Battles require selecting discrete actions (moves, items, switching Pokémon, or attempting capture) under uncertainty due to hidden information (e.g., opponent move choice, variable damage rolls, status effects). Progress is gated by narrative triggers and abilities acquired over time (e.g., key items and field moves), producing long-horizon dependencies, sparse success signals, and frequent context switches between exploration, dialogue, menus, and combat. Bulbapedia [44] provides a comprehensive overview of progression, game mechanics and storyline.

Pokémon Emerald is a demanding test of long-horizon visuomotor intelligence: it must map high-dimensional visual observations to low-level button sequences while coping with partial observability, delayed consequences, and sparse success signals. The outcomes of many actions are unclear over even long time horizons. For example, your starter Pokémon choice may be a crucial factor in winning a battle dozens of hours into a playthrough.

Legend of Zelda: The Minish Cap is a Game Boy Advance game that combines real-time combat, puzzle solving, and RPG mechanics. The player controls Link, a young hero tasked with rescuing Princess Zelda after she is turned to stone by the evil sorcerer Vaati during the annual Picori Festival. To restore Zelda and prevent Vaati from unleashing further chaos upon the kingdom of Hyrule, Link embarks on a journey to reforge the sacred Picori Blade.

The core gameplay loop consists of exploring a diverse set of zones collecting items to save the princess, defeating enemies with the player's sword and shield, and solving increasingly difficult puzzles using the game's RPG mechanics. The game's central mechanic is the ability to use the Minish Cap to shrink Link. This mechanic (in conjunction with other game mechanics) is combined with environmental puzzles to create settings where the player must reason how to use Link's ability to grow and shrink to progress through novel areas and terrains. This mechanic is also central to the

combat in the game: when Link shrinks enemies that were once trivial become large and imposing. Zelda Dungeon [45] provides a detailed walkthrough of the game, including areas, game mechanics, and the storyline of Link's quest.

Legend of Zelda: The Minish Cap is a fascinating environment for several reasons: (1) As with our previous environment, it is long-horizon, with progression spanning many hours and distinct areas. (2) *Progressing requires that the player demonstrates a variety of skills.* The Legend of Zelda is not a game where the player can simply "walk to the finish line," progressing through each zone requires demonstrating *escalating levels of mastery* over the game's core combat mechanics (centered around a sword and shield), the ability to traverse movement challenges, and utilize newly-introduced game mechanics to solve puzzles. (3) Lastly, *the environment rewards open-ended exploration*. As the player is tasked with a series of progressively harder challenges, they can explore the environment to find power-ups that make these challenges easier. For example, the player can obtain *heart containers* which increase the amount of damage the player can take without dying in combat.

E Extended Experiments

E.1 Average Milestone Results

In Table 6, we report the average number of milestones reached (out of 12) in both environments at the end of the 8-hour evaluation.

Method	Pokémon Emerald	Legend of Zelda
ASH	$\mathbf{11.17 \pm 0.83}$	$\mathbf{9.92 \pm 1.51}$
Voyager+Opus 5	9.25 ± 1.71	7.83 ± 2.04
BC	6.50 ± 1.31	6.00 ± 0.00
VPT	5.50 ± 1.62	5.92 ± 0.29
RAEA	2.58 ± 0.51	2.00 ± 0.00

Table 6: Average milestones reached (out of 12) at the end of the 8-hour evaluation, reported as mean $\pm$ one standard deviation.

E.2 Sensitivity to the Scale and Quality of Internet Video

Corpus size. To measure how sensitive ASH is to the scale of our dataset, we ablate using 10%, 50% and 100% of our corpora in our Pokemon Emerald environment. For each dataset size, we average over 4 runs, each 4 hours in length.

Corpus size	Avg. milestones
10%	6.75
50%	7.75
100%	8.25

Table 7: Effect of internet corpus size on ASH in Pokémon Emerald. Each value is an average over 4 runs of 4 hours.

We report average milestones completed for each dataset size in Table 7. The average milestones completed drops by 1.5 with a 90% reduction in dataset size. Therefore, while ASH does benefit from the availability of data, it can still function in settings with substantially less data.

Corpus quality. To evaluate the quality of our data, we sample 200 videos at random from our internet corpus. For each video, we record if it (1) is from a wrong or modded version of the game, (2) has major editing artifacts (custom overlays, facecams, etc.), (3) has poor video quality (blurry due to low resolution or choppy due to low framerate). If any of these are true, we consider the video to be "dirty". We report the results of this analysis in Table 8.

We see that more than half (52.5%) of the corpus is dirty. ASH's ability to extract high quality supervision from this dirty corpus is one of its unique advantages. When VPT and BC baselines were trained directly on this data, they achieved 0 milestones. This indicates that the ASH algorithm as a whole is more robust to noisy data than other methods.

Issue	% of videos
Wrong or modded game version	16.5
Major editing artifacts	35.5
Poor video quality	19.0
Dirty (any of the above)	52.5

Table 8: Manual quality audit of 200 videos sampled uniformly at random from $\mathcal{D}^I$ for Pokémon Emerald. The three categories are not mutually exclusive.

E.3 Catastrophic Forgetting in the IDM

Section 4.3 shows that IDM accuracy rises from 60% to 81% over 8 bootstraps. Each bootstrap trains the IDM on data that ASH retrieves for the agent's *current* roadblock. The IDM could therefore learn the dynamics of the later game and forget the dynamics of the earlier game. To evaluate if the IDM module forgets previous levels, we evaluate Pokémon Emerald IDM checkpoints from successive bootstraps on data collected during the first bootstrap, and report movement class prediction accuracy in Table 9.

IDM checkpoint (bootstrap #)	1	2	3	4	5	6	7	8
Accuracy on bootstrap-1 data (%)	72.5	78.6	74.0	76.5	78.3	68.9	77.1	74.5

Table 9: Movement class accuracy of successive IDM checkpoints. We evaluate every checkpoint on held-out data from the first bootstrap.

Accuracy on early-game data stays largely flat across the 8 bootstraps, ranging from 68.9% to 78.6% with no downwards trend. The IDM therefore does not forget the earlier parts of the game while it improves.

E.4 Sensitivity to the Stuck-Timer Threshold Δ

Dynamic bootstrapping has one hyperparameter: the stuck-timer threshold Δ, the number of steps that an agent may run without a new key moment before ASH bootstraps. To evaluate ASH's sensitivity to the stuck threshold Δ, we perform an ablation. For each value of Δ, we report average milestones completed for 4 runs, each 96 GPU hours in length (Table 10).

Δ (minutes)	5	10	20	40
Avg. milestones	4.00	6.50	6.25	7.00

Table 10: Sensitivity to the stuck-timer threshold Δ in Pokémon Emerald. Each value is an average over 4 runs of 96 GPU hours of online training. We use $\Delta = 20$ minutes throughout the main paper.

We see that at very low Δ value (5 minutes), the agent does not spend enough time exploring, and bootstraps too frequently. However, above this value, ASH's performance does not seem very sensitive to Δ.

E.5 General Skill Acquisition

Bootstrap	0	1	2	3	4	5
Battles won (%)	0.0	40.0	62.5	80.7	92.1	95.4

Table 11: Success rate in randomized battles across successive bootstraps in Pokémon Emerald.

A method that learns from walkthrough video could simply repeat actions from the trajectory it saw, rather than learning general skills. In Pokémon Emerald, the core skill is Pokémon battling: when the player encounters enemies they must use their team of Pokémon to defeat them. To measure if the

agent is learning this skill, we report the average rate of successful battles in successive bootstraps Table 11. We observe that the rate of success in these random encounters increases over bootstraps. This suggests the agent is learning to battle - which is a general skill, and not simply to copy expert demonstrations.

F Hyperparameters and Training Settings

Hyperparameter	**Setting**
Optimizer	AdamW [49]
Learning Rate	5×10^{-5}
LR Scheduling	Linear
Warm-up Ratio	5%
Weight Decay	1×10^{-5}
Max Epochs	3
Batch Size	4
Trajectories (N)	4
Retrieved Videos per Agent k (for training π)	100
Sampling Temperature	1
Seconds between actions	0.25
Observations in Short-term Memory (w_s)	64
Seconds between observations	0.5
$\mathcal{T}$ hidden dimension	1024
ϕ hidden dimension	768
HDBSCAN minimum samples	30

Table 12: Hyperparameters and Settings.

G Notation Glossary

Symbol	Description
Spaces and environment	
$\mathcal{E}$	Environment
$\mathcal{A}$	Action space
$\mathcal{O}$	Observation space
$a \in \mathcal{A}$	Action
$o \in \mathcal{O}$	Observation
Trajectories	
τ	Observation-action trajectory $(o_1, a_1, \ldots, o_T, a_T)$
τ^{obs}	Observation-only trajectory $(o_1, \ldots, o_T)$
τ_n	Trajectory of agent n
$\tau_{1\sim N}$	Trajectories all N agents: $(\tau_1, \ldots, \tau_N)$
$\tau^{obs}_{t:t+w_s}$	Short-term sample $o_{t:t+w_s}$
Policy and models	
$\pi \sim p_\theta(a_{t+1} \mid \cdot)$	Policy
ϕ	Image tokenizer (SigLIP)
$\mathcal{T}$	Causal transformer backbone
IDM	Inverse dynamics model
$\tilde{\boldsymbol{A}}$	Pseudo-action labels from IDM
$\boldsymbol{W}_o$	Output projection matrix
$\boldsymbol{H}$	Hidden states of $\mathcal{T}$
$\mathcal{L}$	Training loss
Memory	
w_s	Short-term memory window size (number of observations)
w_l	Long-term memory window size (number of key moments)
ρ	Long-term memory (most recent w_l key moments)
$\boldsymbol{T}^s$	Short-term memory tokens from ϕ
$\boldsymbol{T}^\ell$	Long-term memory tokens from ϕ
$\mathcal{K}(\tau^{obs}, t)$	Key-moment classifier
$\mathcal{M}_n$	Memory bank of agent n
Δ	Stuck-timer threshold (steps before bootstrap)
Data and retrieval	
$\mathcal{D}^I$	Internet video dataset
$\mathcal{D}^R$	Retrieved video dataset
$\mathcal{D}^\pi$	Policy update dataset
$\boldsymbol{E} \in \mathbb{R}^{e\times d}$	Per-video DINOv2 embedding matrix
$\boldsymbol{Q}$	Per-trajectory query embedding matrix
$\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{E}^\top$	Similarity matrix
w_r	Retrieval window size
k	Number of retrieved videos per agent
$c_{\min}$	Minimum unique trajectories in a cluster
ASH	
N	Number of parallel agents

Table 13: Notation glossary.

H Human Participants

The human annotators were tasked with labeling a small collection of videos of Pokémon Emerald gameplay, corresponding to a couple hours of work. The human participants in this study did not face any risks.

I Search Keywords

Table 14 lists the search terms used to create our internet dataset. To ensure that we index multi-part walkthroughs, we replace {n} with the values 1-80.

Pokémon Emerald	**Legend of Zelda: The Minish Cap**
pokemon emerald full walkthrough	the minish cap full walkthrough
pokemon emerald longplay	legend of zelda the minish cap full walkthrough
pokemon emerald walkthrough	the legend of zelda the minish cap full walkthrough
pokemon emerald gameplay	the minish cap longplay
pokemon emerald playthrough	legend of zelda the minish cap longplay
pokemon emerald full game	the minish cap walkthrough
pokemon emerald full gameplay	legend of zelda the minish cap walkthrough
pokemon emerald all parts	the minish cap gameplay
pokemon emerald complete	legend of zelda the minish cap gameplay
pokemon emerald full version	the minish cap playthrough
pokemon emerald no commentary	legend of zelda the minish cap playthrough
pokemon emerald 100 percent	the minish cap full game
pokemon emerald part {n}	legend of zelda the minish cap full game
pokemon emerald longplay part {n}	the minish cap full gameplay
pokemon emerald let's play part {n}	legend of zelda the minish cap full gameplay
pokemon emerald episode {n}	the minish cap all parts
pokemon esmeralda guía completa	legend of zelda the minish cap all parts
pokemon esmeralda longplay	the minish cap complete
pokemon esmeralda guía	legend of zelda the minish cap complete
pokemon esmeralda gameplay	the minish cap full version
pokemon esmeralda recorrido	legend of zelda the minish cap full version
pokemon esmeralda juego completo	the minish cap no commentary
pokemon esmeralda gameplay completo	legend of zelda the minish cap no commentary
pokemon esmeralda todas las partes	the minish cap 100 percent
pokemon esmeralda completo	legend of zelda the minish cap 100 percent
pokemon esmeralda versión completa	the minish cap 100%
pokemon esmeralda sin comentarios	legend of zelda the minish cap 100%
pokemon esmeralda 100%	the minish cap part {n}
pokemon esmeralda parte {n}	the minish cap longplay part {n}

Table 14: Search terms used for creating internet dataset.

J Broader Impact

Positive. By enabling agents to acquire new skills from widely available observational data, methods such as ASH could reduce the cost of developing systems for domains including robotics, assistive technologies, simulation-based training, and autonomous exploration, where hand-engineered rewards or large labeled datasets are often impractical. More broadly, scalable long-horizon learning may accelerate the development of adaptive systems that can operate in complex real-world environments.

Negative. Systems capable of autonomously acquiring behaviors from internet-scale demonstrations may inherit biases, unsafe behaviors, or undesirable strategies present in online data. In addition, increasingly capable agents could be applied in ways that reduce human oversight in safety-critical settings or be adopted for harmful applications. Our experiments are limited to video game environments, which provide a safe and controlled testbed, but future deployment in physical domains will require robust safeguards.

K Copyright

The copyright for the videos in the web-scraped dataset belongs to the video creators. We use code from the Transformers library and CuML (which are both provided under the Apache License 2.0 license). As part of our open-source release, we will not distribute any copyrighted work that is not freely provided with an open license.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We claim that our adaptive online method is better than baselines are progresss-ing through long-horizon environments, which we evaluate in Section 4.1. We claim that our dynamic bootstrapping and long-memory mechanisms improve ASH's policy, which we show via ablation in Section 4.2.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in our "limitations and conclusions" section (Section 5).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: We do not include theorectical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We report hyperparameters appendix F, data collection keywords used to create our dataset appendix I and architecture settings section 4. We will also release all code and model weights upon publication.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release all code and model weights freely via a github repository and the project page.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We report hyperparameters appendix F, data collection keywords used to create our dataset appendix I and architecture settings section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report standard deviations over our main results in Appendix E.1 and visualize standard deviations in our ablations in Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report computer resources used in Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The authors believe the paper complies with everything listed in the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader societal impacts of our work in Appendix J.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We believe the models (trained to play Pokémon Emerald and the Legend of Zelda) do not have high likelihood of misuse. As our internet dataset is obtained from YouTube, which flags and removes unsafe content, we believe our dataset of video game walkthroughs is safe.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We provide copyright information related to our work in Appendix K.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: For ease of reproducibility, we will provide documentation with our open-source release.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We detail the instructions given to annotators in Section 4 and the setting in Appendix H.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: No volunteer faced risks in our study (Appendix H).

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were only used for basic document formatting and typo detection in our work. They did not contribute to the core content.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.