Birdie: Advancing State Space Models with a Minimalist Architecture and Novel Pre-training Objectives

Anonymous ACL submission

Abstract

State Space Models (SSMs) are emerging as alternatives to Transformers but struggle with tasks needing long-range interactions, such as text copying and multi-query associative recall. Most improvements in SSMs focus on internal architecture rather than exploring diverse pre-training objectives. This paper introduces the Birdie model, a minimalist SSM architecture, with novel pre-training objectives.. Experimental evaluations demonstrate that combining this minimalist architecture designed with refined control over recurrence parameterization with pre-training objectives like infilling, copying, and deshuffling significantly improves performance in practical generative tasks, achieving higher average metric scores and win rates. The findings offer valuable insights for optimizing SSMs to compete with Transformers.[1]

1 Introduction

Thanks to their scaling properties (Hoffmann et al., 2022) and in-context learning (Garg et al., 2023), large transformer models are now prominent in natural language processing (NLP) and have made possible effective performance on natural language generation tasks (NLG), including language modeling, machine translation, and question answering (Yue et al., 2022; Xie et al., 2022; Kumar et al., 2021). Attention, the key to the success of transformers, is also their limitation. As the attention layer computes similarity scores between all pairs of tokens in the input sequence, its computational demands grow quadratically with sequence length.

State Space Models (SSMs) are emerging as promising alternatives due to their subquadratic time complexity. SSMs belong to a class of dynamic models that represent the state of a system at each time step as a linear combination of previous states and the input signal. Originally popular in control theory and time series analysis, SSMs have recently been adapted for discrete data, such as natural language. A flurry of research activity has introduced various SSMs, such as S4 (Gu et al., 2022), S5 (Smith et al., 2023), LRU (Orvieto et al., 2023), Mamba (Gu and Dao, 2023), Hawk and Griffin (De et al., 2024), and others, as covered in some detail in Section 2.

Methodological innovations in SSMs aim to improve performance on long-range dependency tasks. Evaluations show varied success. SSMs demonstrate considerable strength in traditional log-likelihood evaluations, where the objective is to read and choose between multiple predetermined choices for a given question (Gu and Dao, 2023; De et al., 2024).

This objective, however, does not capture what a model would generate in a natural chat setting. When it comes to practical, generative tasks, requiring a sharp probability distribution, such as long-range interactions, text copying, and multi-query associative recall, SSMs fall short of expectations. For instance, work in (Jelassi et al., 2024) demonstrates that SSMs fall short of transformer-based models on copying and selective tasks. These tasks are critical in NLP, where the ability to maintain and manipulate long-term dependencies is crucial for generating coherent text, following directions, copying sequences, and responding accurately to multiple queries.

This paper addresses some of these challenges. It advances SSMs along four main directions.

(1) Current SSMs predominantly utilize causal language modeling and ignore the diversity of pre-training objectives. In contrast, the first contribution this paper makes is the design of **novel pre-training objective mixtures** to bias the model towards learning functions for long-range interactions, specifically focusing on tasks where SSMs currently underperform compared to Transformer models. We introduce reinforcement learning into the pre-training process.

[1] All code and pre-trained models will be publicly released.

(2) We demonstrate that pre-training objective mixtures can elicit superior performance, contrary to current thinking on SSMs. However, figuring the right mixture ratios is an important consideration. A second contribution of this paper is to propose **a dynamic mixture of pre-training objectives via reinforcement learning**. Our results show that Birdie obtains competitive performance against Transformer models with such a dynamic mixture and additionally becomes the first SSM to solve the outstanding phone retrieval task.

(3) The paper proposes a **novel model, Birdie, which implements a generalized SSM architecture with direct control over the recurrence parameterization**. This architecture is informed by the goal of improving performance in the presence of long-range interactions without suffering from decaying state intermediates over the sequence length; such decay is observed in (Gu et al., 2022; Gu and Dao, 2023; De et al., 2024).

(4) By narrowing themselves in the causal language modeling objective, current research on SSMs ignores potentially interesting dynamics between pre-training objectives and the model architecture. The fourth contribution this paper makes is to show the **emergence of non-trivial dynamics** resulting from the combination of a minimalist SSM architecture in the proposed Birdie model and the proposed novel pre-training objectives.

2 Background and Related Work

We relate background concepts and related work.

2.1 SSMs

SSMs maintain an online state that stores information seen in previous time steps. Due to the state's finite dimensionality, SSMs inherently suffer from a limited and fixed memory capacity (Gu and Dao, 2023; Wen et al., 2024). This balances memory efficiency with computational demands of processing long sequences. Gu and Dao (2023) argue that an effective state can be maintained via selective gating which is capable of blocking unnecessary inputs to the state. Here we argue that bidirectionality is also necessary for maintaining a healthy state in terms of intermediate magnitude, as it can help the selective mechanisms identify what information to maintain. We report improved performance when enabling bidirectionality.

A Brief History of SSMs S4 (Gu et al., 2022) was the first popular SSM for NLP that through clever reparameterization of the state space matrices was able to dynamically maintain information across long context windows with few resources. In S5 Smith et al. (2023) reimplement S4, moving its application from a bank of independent single-input, single-output time-invariant SSMs applied using the fast fourier transform, to a multi-input, multi-output SSM, leveraging efficient parallel scans and matching the computational efficiency of S4, while maintaining superior performance on toy tasks measuring long range performance (Smith et al., 2023). Linear Recurrent Units (Orvieto et al., 2023) further simplify S5 through linearizing and diagonalizing the recurrence, using better parameterizations, initializations, and proper normalization of the forward pass.

Later research on SSMs predominantly focuses on the selective mechanism to selectively propagate or forget information along the sequence length dimension depending on the current token. Mamba and Hawk are two recent SSMS that leverage this selective mechanism. Mamba (Gu and Dao, 2023) integrates selective SSMs into a simplified end-to-end neural network architecture without attention or even MLP blocks. In contrast, Hawk and Griffin (De et al., 2024) are inspired by the LRU model but incorporate an LSTM-like gating mechanism; Griffin adds a sliding window attention over Hawk and so is a hybrid model.

Linear SSMs Given a length L sequence of vector-valued inputs $\mathbf{x}_{1:L} \in \mathbb{R}^{L\times D}$, a general class of linear SSMs with hidden states $\mathbf{h}_{1:L} \in \mathbb{R}^{L\times N}$ and outputs $\mathbf{y}_{1:L} \in \mathbb{R}^{L\times D}$ can be computed as

$$\mathbf{h}_k = \mathbf{A}_k\mathbf{h}_{k-1} + \mathbf{B}_k\mathbf{x}_k$$
$$\mathbf{y}_k = \mathbf{g}(\mathbf{h}_k, \mathbf{x}_k)$$

with state transition matrix $\mathbf{A}_k \in \mathbb{R}^{N\times N}$, input matrix $\mathbf{B}_k \in \mathbb{R}^{N\times U}$ and output function $\mathbf{g}(\cdot)$ to produce the outputs. Many recent models fall within this framework and are divided into *linear time-invariant* (LTI) and *linear time-varying* (LTV) dynamical systems. LTI systems have static dynamics parameters across time, i.e. $\mathbf{A}_k = \mathbf{A}$ and $\mathbf{B}_k = \mathbf{B}\ \forall k$, and include recent works in deep SSMs such as S4, S5, and LRU. LTV systems have varying dynamics parameters that may also be data-dependent and include recent methods such as Liquid-S4 (Hasani et al., 2022), Hierarchical GRU (HGRU; Qin et al., 2023), Hawk (De et al., 2024) and Mamba (Gu and Dao, 2023) as well as previous works in linear RNNs (Balduzzi and

Ghifary, 2016; Martin and Cundy, 2018; Bradbury et al., 2016; Lei et al., 2018). The linear dependencies between time steps in linear SSMs allow for efficient parallelization across the sequence via Fast Fourier Transforms (Gu et al., 2022; Fu et al., 2023) or parallel scans (Blelloch, 1990; Martin and Cundy, 2018; Smith et al., 2023).

Recent SSMs have demonstrated improvement in language modeling perplexity and max-likelihood evaluations. Hawk holds SOTA performance for attention-free, selective SSMs on common max-likelihood evaluations. At its core, Hawk is powered by the Real-Gated LRU (RG-LRU), an update of the original LRU (Orvieto et al., 2023) that add a limited amount of input-dependent gating to its a parameterisation. The mathematical formulation of the RG-LRU is:

$$\mathbf{r}_t = \tau(\mathbf{W}^a \mathbf{x}_t)$$

$$\mathbf{i}_t = \tau(\mathbf{W}^x \mathbf{x}_t)$$

$$\mathbf{a}_t = \mathbf{a}^{c r_t}$$

$$\mathbf{h}_t = \mathbf{a}_t \odot \mathbf{h}_{t-1} + \sqrt{1 - \mathbf{a}_t^2}\mathbf{i}_t$$

where the constant c is set to 8. The forget-gate a is defined as $\tau(\Lambda)$, with Λ being a learnable parameter. Λ is initialized such that a^c is uniformly distributed between 0.9 and 0.999. This initialization strategy causes $\mathbf{a}_t$ to start "open," forcing elements in a sequence to mix together. Notably, when Λ is close to these initial values, the input-dependent projection $\mathbf{r}_t$ has minimal influence over the $\mathbf{a}$, limiting the RG-LRU's ability to forget previous time steps. Under these conditions, the RG-LRU of Hawk can be conceptualized as a relatively weaker form of a data-dependent LTV system, with an LTI system governing the base performance. This observation inspires the reparameterization in the proposed Birdie model. As detailed in Section 3.1, Birdie is a fully data-dependent LTV model.

2.2 Pre-Training Objectives

Pre-training "instills" general-purpose knowledge and abilities (Raffel et al., 2020). While the default choice in NLP for a pre-training objective is *causal language modeling* (CLM), or "next word prediction," there are several empirically usable alternatives that have been shown to significantly improve model performance in settings such as general language tasks (Tay et al., 2022, 2023b; Anil et al., 2023), code generation (Bavarian et al., 2022; Rozière et al., 2024), and multi-modal audio and vision Transformers (Chen et al., 2023). Notably, Reka (Team et al., 2024) 7B and 21B, pre-trained using Mixture-of-Denoisers (MoD), outperform competing frontier models (ChatGPT/Mistral) in their compute class on many multi-modal tasks.

Alternatively to CLM, *masked language modeling* (MLM) includes objectives where certain tokens are replaced with a mask token, and the model must predict the original tokens. In its original conception with BERT (Devlin et al., 2019), each mask token represented one obfuscated input token. *Span corruption* (SC) extends BERT's MLM objective to generative models. For a given input, several spans of tokens are replaced with unique sentinel tokens. The model then generates the masked tokens and their respective sentinel tokens.

Fill-in-the-Middle encompasses elements from both SC and CLM (Bavarian et al., 2022) and fuses them into one objective. Fill-in-the-Middle masks out a large middle span and moves it to the end of the sequence. This objective enforces a standard language modeling loss on all tokens, including the prefix, intended for processing by a fully-causal model. This is a popular training objective, but did not appear helpful during our pilot runs on SSMs (data not shown).

Prefix language modeling (PLM) modifies this approach slightly. A loss is specifically not calculated on a prefix and the model is allowed a bidirectional view of the context. During pre-training, input sequences are randomly split in two, with the prefix functioning as context and the suffix being the target for the direct loss computation (Raffel et al., 2020).

In this paper, we consider the above representative pre-training objectives and integrate them in a mixture-of-denoisers setting. As described in Section 3, we add new objectives and *dynamic* mixtures to improve SSMs.

2.3 Benchmarking SSMs

Jelassi et al. (2024) benchmarks two representative NLP models, Mamba (as a SOTA SSM) and Pythia (a decoder-only transformer model) on the same dataset. The paper shows serious deficiencies in SSMs. Two key concerns in NLP include being able to copy text outside of the training length and recall via extracting a phone number from a phonebook. In both scenarios, Jelassi et al. (2024) shows that Pythia outperforms Mamba by a significant margin. Similarly, Hawk and Griffin (Hawk with an added sliding window attention layer) both significantly struggle with the same phone number recall task. Notably, both Mamba and Hawk have been shown to solve synthetic copying (Arjovsky et al., 2016) and induction head (Olsson et al., 2022) tasks, which are designed to measure the ability of models to correctly memorize tokens and recall them after reading noise. Similarly contrasting differences between synthetic associative recall and closer-to-real-world task performance are evaluated by Arora et al. (2023). Informed by the deficiencies in current SSMs, as demonstrated in (Jelassi et al., 2024), to address these issues with "real" models (i.e., not a synthetic model trained on synthetic data), in this paper we utilize pre-training objectives to enable an SSM to learn to manage long-range dependencies and copy/retrieve information from earlier in the sequence.

3 Methods

3.1 Birdie

Birdie is defined as follows:

$$\mathbf{z}_t = \tau(\mathbf{W}^K \mathbf{x}_t) \in \mathbb{R}^N$$
$$\mathbf{f}_t = \sigma(\mathbf{W}^f \mathbf{x}_t) \in \mathbb{R}^N$$
$$\mathbf{h}_t = \mathbf{f}_t \odot \mathbf{h}_{t-1} + \mathbf{z}_t$$
$$\mathbf{y}_t = \mathbf{W}^{out} \mathbf{h}_t,$$

where σ is the standard logistic sigmoid function, $\mathbf{x}_t$ is a normalized input, and $\mathbf{y}_t$ is added to a residual connection. In particular, Birdie is a gated linear RNN. Unlike the Gated Impulse Linear Recurrent Layer in (Martin and Cundy, 2018), where $\mathbf{h}_t = \mathbf{f}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{f}_t) * \mathbf{z}_t$, Birdie removes the coupling between $\mathbf{1} - \mathbf{f}_t \odot \mathbf{z}_t$ and all bias terms.

In recent SSM literature, one observes a tendency to add more terms and couplings to the parameterized recurrence equations. In Birdie we take the opposite approach, simplifying and decoupling. Partially driven by work in (Tay et al., 2023a), which shows that minimalist transformer models outperform more complex architectures, our intuition in designing Birdie was to investigate a core architecture and its dynamics with a variety of pre-training objectives.

Birdie can move information along the sequence without numerical decay; that is, data can move forward perfectly, for an infinite amount of time. In contrast, previous SSMs like S4, S5, MEGA, and H3 induce a non-correctable fixed decay during the propagation of information (note: $h(t) = A * h(t-1) + Bx(t)$, where $A < 1.0$), meaning that regardless of what the model has learned, raw information cannot physically flow past a certain distance. This does not happen in Birdie. We have empirical evidence that f_t reaches 1, and this is prominent during span corruption, shown in Section 5.3. In Section A.5 we relate additional observations that the parameterizations in Birdie are dynamic, with different pre-training objectives inducing different behaviors. This dynamic behavior is observed in the presence of a fixed state, suggesting that Birdie is fully utilizing state capacity independent of sequence length.

4 Pre-training Objectives and Mixtures

Table 1 lists classic and new objectives and mixtures we investigate. We include pre-training objectives and tasks that we hypothesize will enhance an SSMs' abilities to handle long-range dependencies, potentially improving performance on downstream tasks. Novel pre-training objectives proposed in this table, highlighted in bold font, include: Selective Copying, Full Span Corruption with Deshuffling (a fixed-ratio MoD), PT5 (a fixed-ratio MoD over objectives listed in Table 1), and RL-Mod, a dynamic MoD with optimal ratios determined via the proposed reinforcement learning approach described later in this paper. The Selective Copying pre-training objective proposed here is inspired by work in (Olsson et al., 2022), but we note that Olsson et al. (2022) introduce it as a synthetic induction head task to guide the design of potentially improved small models.

Span Corruption (SC): This is the standard objective from T5, where a span of tokens is corrupted and the model must generate the masked spans, as illustrated in Table 1. It is worth noting that for an SSM, span corruption takes on a significantly more difficult form, since all relevant information must be compressed into a single state, so we include it in the list of pre-training objectives we investigate. In addition, we explore two variations:

Full Span Corruption (FSC): The model must generate the entire de-noised sequence. This is similar to BERT's MLM task, but the model generates the entire sequence rather than filling in masked tokens in-place. This objective was named BERT-style and was included in an ablation in T5. This tasks the model with maintaining a state where it can simultaneously copy from a context while generating new text conditioned on the context.

FSC with Deshuffling (FSC-D): This is a novel variation we introduce in this paper. It builds over

FSC but shuffles the non-corrupted spans so that it is more difficult for the model to only pay attention to surrounding tokens and so find the right context and copy large amounts of text.

- **Text**: `Bird songs fill the early morning air`

Objective	**Example**
CLM	In: – Tgt: `Bird songs fill the early morning air`
PLM	In: `Bird songs fill` Tgt: `the early morning air`
SC	In: `Bird [mask] the early [mask]` Tgt: `songs fill [mask] air [mask]`
Deshuffling	In: `the early [mask] Bird [mask]` Tgt: `Bird songs fill the early morning air`
Copying	In: `the early [mask] Bird [mask]` Tgt: `Bird songs fill the early morning air`
Selective Copying	In: `the early [mask] Bird [mask]` Tgt: `Bird songs fill the early morning air`
	Mixture of Denoisers (MoD)
FSC	In: `Bird [mask] the early [mask]` Tgt: `Bird songs fill the early morning air`
FSC-D	In: `the early [mask] Bird [mask]` Tgt: `Bird songs fill the early morning air`
UL2	Fixed-ratio mixture of PLM and SC (Tay et al., 2023b).
PT5	A new mixture of CLM, PLM, SC, Deshuffling, and Copying at fixed ratios found via ablations.
RL-MoD	A novel dynamic mixture of CLM, PLM, SC, Deshuffling, Copying, and Selective Copying at optimal ratios found via reinforcement learning.

Table 1: CLM: Causal language modeling. PLM: Prefix language modeling. SC: Span corruption. FSC: Full SC. FSC-D: FSC with deshuffling. In: input; Tgt: target. New pre-training objectives and MoDs are **bolded**.

Deshuffling: The model is given an input sequence with shuffled tokens. The model must deshuffle the tokens to recreate the original sequence. We use two variations: one where 50% of the input tokens are shuffled, and a second where all inputs tokens are shuffled.

Copying: We include two pre-training tasks that do not involve denoising an input, inspired by recent work (Jelassi et al., 2024) that highlights challenges with SSMs in copying tasks. In Copying, the model must recreate the input sequence. This is a key component of many tasks requiring long-range dependencies. We introduce here a novel variation, Selective Copying, in which the model is given beginnings and endings of spans in the context and then must find and copy these spans to the output. The model copies specific spans of text from the input. This tasks differs from standard copying in that not all text is copied and the spans to copy are not necessarily found in order in the context. This can be seen as an analog to the downstream phone book look-up task.

4.1 Optimal Mixtures with Objective Sampling via Reinforcement Learning

Model architecture and size has a significant effect on pre-training objective performance. As shown in Figure 1 for the selective copying task, we observe that Birdie at 400M parameters achieves accuracy only in the high 30%s but reaches 93% at 1.4B parameters when trained to only 32B tokens, approaching a nearly 100% accuracy by a 6-layer transformer at 141M parameters.

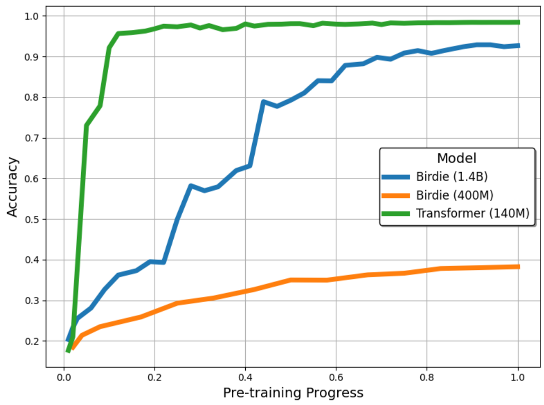

Figure 1: Accuracy on the selective copying task during pre-training on a shared validation set from The Pile (Gao et al., 2020) across Birdie (1.4B and 400M parameters) and a transformer of 141M parameters.

This suggests that static pre-training objective mixes, like those in BERT and UL2 (Devlin et al., 2019; Tay et al., 2023b), may not suit all model architectures. Despite the limited details shared by Team et al. (2024) and (PAL) on their use of curriculum-based techniques, the challenge remains in optimally scheduling and adjusting mixture rates for varying model configurations, as noted by (Tay et al., 2022).

To address this, we propose a dynamic, automated curriculum that adapts pre-training task mixtures according to the evolving needs of the model and exploits synergies between objectives. Our approach utilizes a critic model, which predicts rewards for each objective, given past actions and their observed outcomes.

Overall, this forms a classic multi-armed bandit framework and is related to a recent Gaussian Process approach for masking rates in MLM (Urteaga et al., 2023), which we found ineffective for our diverse objectives. We adopt a two-layer Transformer to directly predict per-objective rewards based on historical data. Our optimization method is akin to Direct Preference Optimization (DPO) (Rafailov et al., 2023), avoiding drawbacks from fixed policies and empirically follows the model's drift during training.

This approach, visualized in Appendix Figure 4, shows that training on Full Span Corruption occasionally boosts Copying and Deshuffling objectives to the extent that their sampling can be drastically reduced. Interestingly, Span Corruption benefits from Selective Copying early in training even without a direct copying component.

Mathematically, we define the reward estimation process as $R_t^i = \mathbb{E}[r(\mathbf{o}_t^i|\theta_{t-1})]$ for $i = 1, 2, \ldots, N$, where R_t^i is the estimated reward for objective i at time step t, r is the reward function, $\mathbf{o}_t^i$ represents the objective i at time step t, and θ_{t-1} denotes the model parameters at the previous time step. We then select the top M actions with the highest rewards $\mathbf{O}_t = \arg\max_M\{R_t^1, R_t^2, \ldots, R_t^N\}$. Finally, at the next evaluation step, the model is trained on the selected actions: $\theta_t = \theta_{t-1} + \eta\nabla\frac{1}{M}\sum_{i\in\mathbf{O}_t} r(\mathbf{o}_t^i|\theta_{t-1})$, where η is the learning rate.

We find our approach consistently improves performance across all tested pre-training objectives. The dynamic MoD resulting from this approach is referred to as RL-MoD in our experiments.

4.2 Experimental Setup

We carry out three lines of investigation. First, we relate a comprehensive comparison that pitches Birdie against transformer-based models at various configurations (base versus instruction fine tuning, 400M versus 1.4B parameters, and various pre-training objectives) over 14 max-likelihood tasks from the EleutherAI LM Harness (`https://github.com/EleutherAI/lm-evaluation-harness`). These tasks are listed in Appendix A. Second, we relate a detailed analysis over the phone number retrieval task, which has been shown to be a particularly challenging task for SSMs (Jelassi et al., 2024), and we show here Birdie being the first SSM to solve this task. Third, we showcase interesting dynamics from the combination of a minimalist architecture in Birdie and pre-training objectives.

Pretraining: We train Hawk and Birdie on The Pile (Gao et al., 2020) dataset using sequence packing and proper masking to prevent sample interference. We investigate 400M-parameter models trained for 16,000 steps with a batch size of 260 versus 1.4B-parameters models trained for 32,000 steps with a batch size of 520. As in recent literature, all models were pre-trained with a sequence length of 2048. Following recommendations by Chowdhery et al. (2022), we pre-train slightly over Chinchilla optimal scaling laws (Hoffmann et al., 2022) – 20-25x tokens per parameter.

Instruction Tuning: For 1.4B parameter models, we loosely follow the progressive learning fine-tuning procedure from Orca 2 (Mitra et al., 2023) and integrate common instruction-tuning procedures from FLAN (Longpre et al., 2023), Zephyr (Tunstall et al., 2023), and Tulu (Wang et al., 2023). We extend the sequence length to 4096 and 8192. More details on pre-training and fine-tuning can be found in Sections A.1-A.2.

5 Results

We present here our main findings.

5.1 Comparative Performance and Ablation Study on Max-likelihood Tasks

Table 2 relates the average task accuracy for various model configurations (model architecture - Birdie versus transformer, size – 1.4B versus 400M), objectives (fixed MoD objectives, such as UL2 and PT5, versus our RL-resulting dynamic MoD RL-MoD, and base versus instruction-tuned. Section A.4 in Appendix A describes each of these tasks as well as relates the performance of each model configuration on each task.

Several observations can be made from Table 2. First, in each setting, whether base versus instruction fine-tuned and whether at 1.4B or 400M, Birdie elicits competitive performance. It is worth noting that inclusion of the additional objectives and copying tasks do not hurt average downstream performance: instead, we observe that Birdie training with a mixtures-of-denoisers is always among the top performers. This is an important finding, as current literature on generative SSMs default to CLM as the pre-training objective. Another interesting finding suggested by the results in Table 2 is that RL-MoD elicits the best performance with growing model size. Birdie at 1.4B parameters with RL-MoD begins to pulls away from the other models with an average task performance of 2.5% higher than the next model, the attention-based model with CLM. We note that CLM is currently the most popular pre-training objective for Attention-based models in literature.

Model	Objective	Avg Task Accuracy
Instruction Tuned, 1.4B		
Birdie (RL-MoD)	RL-MoD	**45.5%**
Attention (CLM)	CLM	43.0%
Birdie (PT5)	PT5	42.5%
Birdie (CLM)	CLM	40.9%
Base Models, 1.4B		
Birdie (PT5)	PT5	**41.0%**
Birdie (CLM)	CLM	40.9%
Birdie (RL-MoD)	RL-MoD	40.6%
Attention (CLM)	CLM	40.1%
Instruction Tuned, 400M		
Birdie (UL2)	UL2	**40.3%**
Attention (UL2)	UL2	40.2%
Hawk (PT5)	PT5	39.3%
Attention (CLM)	CLM	39.2%
Hawk (CLM)	CLM	38.4%
Base Models, 400M		
Birdie (CLM)	CLM	**40.3%**
Birdie (RL-MoD)	RL-MoD	40.1%
Attention (CLM)	CLM	39.7%
Birdie (UL2)	UL2	39.5%
Birdie (PT5)	PT5	39.3%
Attention (UL2)	UL2	39.2%
Hawk (PT5)	PT5	38.8%
Hawk (CLM)	CLM	38.4%

Table 2: Average task accuracy over 14 EleutherAI LM Harness tasks (listed in Appendix A) for various model configurations (model architecture, size), objectives, and base versus instruction-tuned. FLAN-style (Longpre et al., 2023) templates and accuracy normalized by target token length is used. The full chart is in Table 2.

Table 3 relates compute cost between models (Birdie 1.4B with RL-MoD, Transformer 1.4B, and reference models for Hawk 1.4B and Flash Attention 2 (Dao, 2023) with causal masking), going beyond parameter count comparisons. On our hardware (Nvidia A100 and Google TPUv3 and TPUv4's), Hawk required significantly more time to train as compared Birdie, primarily from its high-dimensional 1D convolution operation, so we do not fully pre-train it. It is worth noting that Birdie at 1.4B is the most compute efficient, which can allow for practical savings and a longer model training for a given compute budget.

Backend	Model	GPU Hrs (A100)	Sec / Step	Seq Length	Tokens / sec / A100
JAX	Birdie 1.4B	5,600	3.5	N/A	15,214
JAX	Hawk 1.4B	7,680	4.8	N/A	11,093
JAX	Transformer 1.4B	10,016	6.3	2048	8,506
Torch	Flash Attn. 2	7,011	4.4	2048	12,152

Table 3: Comparison of observed model training speeds.

5.2 Analysis on Phone Number Retrieval

Inspired by work in (Jelassi et al., 2024), which shows that SSMs cannot pull the phone numbers out of the textbook, we compare the performance of various model configurations on the phone number retrieval task. We design a more difficult variant by increasing the number of phone book entries from 200 to 800 and using a variety of name formats (more closely resembling a phone book in which names can be more complex than simple first and last names). For example, an entries in (Jelassi et al., 2024) may be "Firstname Lastname 123-456-7890" and may not encaspulate a typical phone book. To improve realism, we generate random names using the most common first and last names of individuals from the US Census. The evaluation is limited to an exact match.

Table 4 compares model configurations over growing phone book entries (200–1600) and sequence length in pre-training (2048 versus 16384). Given Hawk's compute demands, we only include it at 400M parameters. All are base models lightly fine-tuned for 100 steps (arbitrarily chosen) due to the Transformer's compute demands. We observe that when trained with RL-MoD, Birdie 1.4B is able to reliably retrieve a phone number during this task, making it the first known SSM to solve this single number retrieval task.

In Fig. 2 we dig deeper and show accuracy versus sequence length for the phone number retrieval task. We observe that when trained with RL-MoD, Birdie achieves Attention-class performance on this task, even at the longest sequence lengths tested. In contrast, Birdie with CLM, the conventional NLP pre-training objective (particularly for SSMs), scores 0% accuracy everywhere except on the shortest sequence length. Note that we expect the Transformer's performance to return if fine-tuned longer as permitted by a larger computational budget.

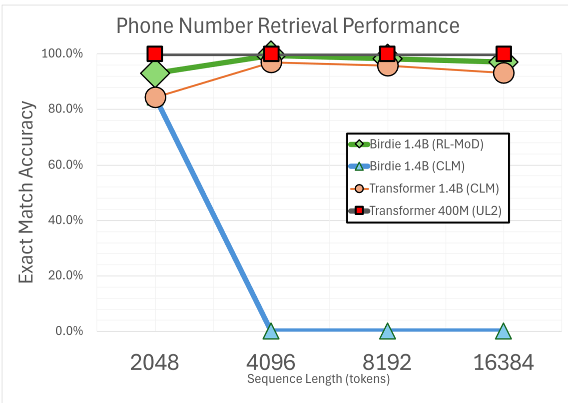

Figure 2: Phone number retrieval exact match accuracy versus sequence length for Birdie (with CLM versus RL-Mod) and a Transformer (with CLM versus UL2) at 400M and 1.4B parameters.

Model	Entries: Seq. Length:	100 (2048)	200 (4096)	400 (8192)	800 (16384)
1.4B Base Models					
Birdie 1.4B (RL-MoD)		93.0%	99.4%	98.1%	96.9%
Transformer 1.4B (CLM)		84.4%	96.9%	95.6%	93.1%
Birdie 1.4B (CLM)		84.4%	0.0%	0.0%	0.0%
Birdie 1.4B (MoD)		88.0%	98.9%	92.0%	91.3%
400M Base Models					
Birdie 400M (RL-MoD)		1.8%	0.0%	0.0%	0.0%
Hawk 400M (CLM)		0.0%	0.0%	0.0%	0.0%
Hawk 400M (PT5)		0.0%	0.0%	0.0%	0.0%
Transformer 400M (UL2)		100.0%	100.0%	100.0%	100.0%
Transformer 400M (CLM)		83.9%	98.9%	94.4%	90.0%
Birdie 400M (CLM)		3.4%	0.0%	0.0%	0.0%
Birdie 400M (UL2)		0.0%	0.0%	0.0%	0.0%

Table 4: Exact match accuracy on the phone number retrieval task for various model configurations across different numbers of entries and sequence lengths. We finetune each model for 100 steps to allow for Transformers to adjust to new positional encodings and the SSMs to only adjust slightly.

5.3 Architecture-Objectives Dynamics

During span corruption, we observe the recurrence parameterization in Birdie abruptly resets along a significant portion of state dimensions at, as shown in Figure 3. This shows f_t reaching 1 during span corruption, perfect data transfer during UL2, and a unique relationship between z_t and f_t during copying.

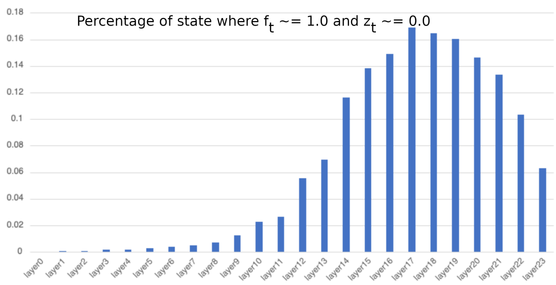

Figure 3: A sizable portion of the state in later layers in Birdie can be seen performing lossless data transfer along the sequence during span corruption tasks. These intermediates were taken from a 400M Birdie trained on UL2, running on validation data from The Pile with span corruption noise applied to them.

6 Conclusion

This work contributes to the ongoing discourse on the enhancement of SSMs. It makes the case that, while architectural innovations are undeniably valuable, pre-training objectives can be equally, if not more, pivotal in advancing capabilities, especially in areas where SSMs traditionally fall short of attention-based models. In particular, our findings suggest that the conventional CLM pre-training objective may not be optimally aligned with the inherent strengths and limitations of SSMs. By exploring bidirectional pre-training and integrating objectives tailored to improve infilling, copying, and handling of long-range dependencies, we demonstrate the potential for significant performance improvements. This approach not only reevaluates the role of pre-training in model development but also posits that SSMs can achieve enhanced performance through careful objective selection, thereby offering a new pathway for SSM enhancement beyond architectural tweaks.

We introduced Birdie, a minimalist SSM model that utilizes these objectives well. Birdie exhibits a dramatic improvement with an RL MoD objective, improving performance greatly on downstream tasks and synthetics like retrieval and copying.

Taken altogether, this work presents a compelling case for reevaluating the conventional approaches to enhancing SSMs through a focused examination of pre-training objectives. By demonstrating the significant performance gains achievable through this lens, we advocate for a broader reconsideration of how SSMs are developed and optimized. The introduction of Birdie serves as a tangible example of the benefits this approach can bring, offering a new direction for future research. We hope that our findings will inspire further exploration of pre-training objectives as a critical factor in the advancement of SSMs and their application in solving complex NLP challenges.

7 Limitations

A consequence of the auto-regressive approach is that Birdie must try and determine what information to send forward to be decoded without clear knowledge of what exactly is being currently generated. Alternatively, a diffusion-based approach may enable Birdie's bidirectional capabilities to be leveraged for fetching information relevant to the current generation for the decoder. This is in contrast to Attention, which can better attend to any token at any time.

While our findings suggest Birdie is the first known SSM to solve the long-standing phone number retrieval task, interpolating from Birdie's performance on the selective copying pre-training performance, we would expect performance to drop when asked to copy more numbers simultaneously.

This paper shows dynamics between model architecture and pre-training objective results in performance gains, which is an important first step, but further research is needed to explore these dynamics, especially in the context of longer sequence lengths and more diverse task requirements. Model scaling is an additional concern. While we are limited in terms of compute we can afford on model pre-training, potentially different dynamics may emerge at very large model sizes and significant overtraining, such as grokking. Recent work (Wang et al., 2024) suggests that this is an interesting research direction for advancing language models.

8 Ethics Statement

Our research advances an alternative framework to the transformer for natural language understanding via SSMs. Not relying on attention, SSMs potentially offer a more sustainable computational framework, as their compute demands grow only linearly with input sequence length. This reduced time complexity translates into energy and carbon footprint savings, directly benefiting our society and opening the way to sustainable AI models. Academic labs and small and medium business enterprises, including start-ups, do not have access to the large computational resources that the big tech industry does. Therefore, advancing SSMs also democratizes research in LLMs, and in doing so potentially increases innovation futher, by allowing more researchers to participate in scientific advancement.

References

AI across google: PaLM 2. https://ai.google/discover/palm2/. Accessed: 2023-05-28.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, et al. 2023. Palm 2 technical report. *Preprint*, arXiv:2305.10403.

Martin Arjovsky, Amar Shah, and Yoshua Bengio. 2016. Unitary evolution recurrent neural networks. *Preprint*, arXiv:1511.06464.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. 2023. Zoology: Measuring and improving recall in efficient language models. *Preprint*, arXiv:2312.04927.

David Balduzzi and Muhammad Ghifary. 2016. Strongly-typed recurrent neural networks. *Preprint*, arXiv:1602.02218.

Mohammad Bavarian, Heewoo Jun, Nikolas Tezak, John Schulman, Christine McLeavey, Jerry Tworek, and Mark Chen. 2022. Efficient training of language models to fill in the middle. *Preprint*, arXiv:2207.14255.

Guy E. Blelloch. 1990. Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University.

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2016. Quasi-recurrent neural networks. *Preprint*, arXiv:1611.01576.

Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, Siamak Shakeri, Mostafa Dehghani, Daniel M. Salz, Mario Lucic, Michael Tschannen, Arsha Nagrani, Hexiang Hu, Mandar Joshi, Bo Pang, Ceslee Montgomery, Paulina Pietrzyk, Marvin Ritter, A. J. Piergiovanni, Matthias Minderer, Filip Pavetic, Austin Waters, Gang Li, Ibrahim M. Alabdulmohsin, Lucas Beyer, Julien Amelot, Kenton Lee, Andreas Steiner, Yang Li, Daniel Keysers, Anurag Arnab, Yuanzhong Xu, Keran Rong, Alexander Kolesnikov, Mojtaba Seyedhosseini, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. 2023. Pali-x: On scaling up a multilingual vision and language model. *ArXiv*, abs/2305.18565.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, et al. 2022. Palm: Scaling language modeling with pathways. *Preprint*, arXiv:2204.02311.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *Preprint*, arXiv:2307.08691.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. 2024. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *Preprint*, arXiv:2402.19427.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Daniel Y. Fu, Hermann Kumbong, Eric Nguyen, and Christopher Ré. 2023. Flashfftconv: Efficient convolutions for long sequences with tensor cores. *Preprint*, arXiv:2311.05908.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling. *Preprint*, arXiv:2101.00027.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. 2023. What can transformers learn in-context? a case study of simple function classes. In *NeurIPS*.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 394–398, Montréal, Canada. Association for Computational Linguistics.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *Preprint*, arXiv:2312.00752.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. In *ICLR*.

Ramin Hasani, Mathias Lechner, Tsun-Hsuan Wang, Makram Chahine, Alexander Amini, and Daniela Rus. 2022. Liquid structural state-space models. *Preprint*, arXiv:2209.12951.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, et al. 2022. Training compute-optimal large language models. In *NeurIPS*.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. 2024. Repeat after me: Transformers are better than state space models at copying. *Preprint*, arXiv:2402.01032.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. *Preprint*, arXiv:1909.06146.

Sachin Kumar, Antonios Anastasopoulos, Shuly Wintner, and Yulia Tsvetkov. 2021. Machine translation into low-resource language varieties. In *ACL-IJNLP*, Online. Association for Computational Linguistics.

Tao Lei, Yu Zhang, Sida I. Wang, Hui Dai, and Yoav Artzi. 2018. Simple recurrent units for highly parallelizable recurrence. *Preprint*, arXiv:1709.02755.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *13th International Conference on the Principles of Knowledge Representation and Reasoning, KR 2012*, Proceedings of the International Conference on Knowledge Representation and Reasoning, pages 552–561. Institute of Electrical and Electronics Engineers Inc. 13th International Conference on the Principles of Knowledge Representation and Reasoning, KR 2012 ; Conference date: 10-06-2012 Through 14-06-2012.

Wing Lian, Guan Wang, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, "Teknium", and Nathan Hoos. 2023. Slimorca dedup: A deduplicated subset of slimorca.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. *Preprint*, arXiv:2301.13688.

Eric Martin and Chris Cundy. 2018. Parallelizing linear recurrent neural nets over sequence length. *Preprint*, arXiv:1709.04057.

Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, Hamid Palangi, Guoqing Zheng, Corby Rosset, Hamed Khanpour, and Ahmed Awadallah. 2023. Orca 2: Teaching small language models how to reason. *Preprint*, arXiv:2311.11045.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2022. In-context learning and induction heads. *Preprint*, arXiv:2209.11895.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. 2023. Resurrecting recurrent neural networks for long sequences. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa : A large-scale multi-subject multi-choice dataset for medical domain question answering. *Preprint*, arXiv:2203.14371.

Mohammad Taher Pilehvar and José Camacho-Collados. 2018. Wic: 10, 000 example pairs for evaluating context-sensitive representations. *CoRR*, abs/1808.09121.

Zhen Qin, Songlin Yang, and Yiran Zhong. 2023. Hierarchically gated recurrent neural network for sequence modeling. *Preprint*, arXiv:2311.04823.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code. *Preprint*, arXiv:2308.12950.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. WINOGRANDE: an adversarial winograd schema challenge at scale. *CoRR*, abs/1907.10641.

Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. 2023. Simplified state space layers for sequence modeling. In *ICLR*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Samira Abnar, Hyung Chung, William Fedus, Jinfeng Rao, Sharan Narang, Vinh Tran, Dani Yogatama, and Donald Metzler. 2023a. Scaling laws vs model architectures: How does inductive bias influence scaling? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12342–12364, Singapore. Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, et al. 2023b. Ul2: Unifying language learning paradigms. In *ICLR*.

Yi Tay, Jason Wei, Hyung Won Chung, Vinh Q. Tran, David R. So, Siamak Shakeri, Xavier García, Huaixiu Steven Zheng, Jinfeng Rao, Aakanksha Chowdhery, Denny Zhou, Donald Metzler, Slav Petrov, Neil Houlsby, Quoc V. Le, and Mostafa Dehghani. 2022. Transcending scaling laws with 0.1% extra compute. In *Conference on Empirical Methods in Natural Language Processing*.

Reka Team, Aitor Ormazabal, Che Zheng, Cyprien de Masson d'Autume, Dani Yogatama, Deyu Fu, Donovan Ong, Eric Chen, Eugenie Lamprecht, Hai Pham, Isaac Ong, Kaloyan Aleksiev, Lei Li, Matthew Henderson, Max Bain, Mikel Artetxe, Nishant Relan, Piotr Padlewski, Qi Liu, Ren Chen, Samuel Phua, Yazheng Yang, Yi Tay, Yuqi Wang, Zhongkai Zhu, and Zhihui Xie. 2024. Reka core, flash, and edge: A series of powerful multimodal language models. *Preprint*, arXiv:2404.12387.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of lm alignment. *Preprint*, arXiv:2310.16944.

Iñigo Urteaga, Moulay-Zaïdane Draïdia, Tomer Lancewicki, and Shahram Khadivi. 2023. Multi-armed bandits for resource efficient, online optimization of language model pre-training: the use case of dynamic masking. *Preprint*, arXiv:2203.13151.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing *and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Boshi Wang, Xiang Yue, Yu Su, and Huan Sun. 2024. Grokked transformers are implicit reasoners: A mechanistic journey to the edge of generalization. *Preprint*, arXiv:2405.15071.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. How far can camels go? exploring the state of instruction tuning on open resources. *Preprint*, arXiv:2306.04751.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

Kaiyue Wen, Xingyu Dang, and Kaifeng Lyu. 2024. Rnns are not transformers (yet): The key bottleneck on in-context retrieval. *Preprint*, arXiv:2402.18510.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, et al. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. In *EMNLP*.

Xiang Yue, Ziyu Yao, and Huan Sun. 2022. Synthetic question value estimation for domain adaptation of question answering. In *ACL*.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369, Hong Kong, China. Association for Computational Linguistics.

A Appendix

A.1 Pretraining

We train all models on the same amount of data from The Pile.(Gao et al., 2020)[2]. The Pile is a collection of several datasets, and includes books, code, web scrapes, emails, and question-answer instruction formatted examples.

During all training and finetuning, we always use sequence packing and proper masking for all models, preventing samples from interfering with each other. For Hawk, we add spacing between samples to prevent the Conv1D layer from leaking out information. Models in the 400M class trained for 16,000 steps with a batch size of 260. Models with approximately 1.4B parameters trained for 32,000 steps, with a batch size of 520. All models wre pre-trained with a sequence length of 2048.

We pre-train on The Pile and train slightly over Chinchilla optimal scaling laws (20-25x tokens per parameter), for a maximum of 32B pre-training tokens for the 1.4B parameter models. We count both context and target tokens as tokens "seen" by the model. This provides a fair comparison among different pre-training objectives. This diverges from other approaches, which do not always consider context tokens in their total count of tokens on which the model was trained (Tay et al., 2023b).

We use the same hyperparameters for all models, using the same settings, such as learning rates and batch sizes, as models found in Mamba (Gu and Dao, 2023). We augment our Transformer with Llama 2 Long's positional encodings.

A.2 Instruction Tuning

For 1.4B parameter models, we largely follow the progressive learning fine-tuning procedure from Orca 2 (Mitra et al., 2023), as immediately jumping into relatively difficult, small datasets, such as SlimOrca-Dedup (Lian et al., 2023) ended up hurting performance. We follow common instruction-tuning procedures from FLAN (Longpre et al., 2023), Zephyr (Tunstall et al., 2023), and Tulu (Wang et al., 2023) with dropout, cosine decay learning rate, and no weight decay. We use all training, validation, and test sets as provided by the original authors.

We first finetune using the same hyperparameters as in FLAN's paper, but since we use AdamW and not AdaFactor, we need a different learning rate to compensate for the lack of AdaFactor's parameter-scaled updates. We simply use a gentle 3e-4 peak cosine LR as in Zephyr (Tunstall et al., 2023) over 4 epochs. For FLAN, we extend the sequence length to 4096 and use a batch size of 20 to keep the number of tokens per batch equal with the original publication. A motivation for choosing such a relatively lengthy fine-tuning procedure was to show if different pre-training objectives maintained differences between the models after finetuning. Based on our results, the differences hold. and the models are discernible.

A.3 Hardware

We present models from 400M to 1.4B parameters. We train using 5 machines with 4 Nvidia A100 80GB's each, and also perform some finetuning and evaluations using TPU V4-32. PT5 was found by training small 110M Birdie and Attention models with random mixtures and continously evolving from there. Birdie exhibited drastically different performance based on its size, so we ensured that our ratio worked well for the 1.4B model, also. This took over 50 iterations of training the 110M model, which took roughly 5 hours each.

A.4 EleutherAI LM Harness Tasks for Downstream Performance Evaluation

Table 5 shows the performance over each of the above tasks for various model configurations.

A.5 Interesting Dynamics in Birdie Pre-training

The main article relates interesting dynamics regarding stop of information flow during span corruption. Here we relate additional observations that the parameterizations in Birdie are dynamic, with different pre-training objectives inducing different behaviors. Figure 4 shows how the reinforcement learning adjusts the pretraining objective mixtures in Birdie 1.4B.

[2]We use the full version of The Pile, last available mid-2023

EleutherAI LM Harness Downstream Tasks	**Description**
arc_easy	The 'Easy' portion of a multiple-choice question-answering dataset, containing questions from science exams from grade 3 to 9 (Clark et al., 2018).
arc_challenge	The Challenge portion of the dataset, containing the more difficult questions that require reasoning (Clark et al., 2018).
medmcqa	A large-scale, Multiple-Choice Question Answering (MCQA) dataset designed to address real-world medical entrance exam questions (Pal et al., 2022).
winogrande	A large-scale dataset of 44k problems, inspired by the original Winograd Schema Challenge (WSC) design (Levesque et al., 2012), but adjusted to improve both the scale and the hardness of the dataset (Sakaguchi et al., 2019).
wic	A large-scale Word in Context dataset based on annotations curated by experts for generic evaluation of context-sensitive representations (Pilehvar and Camacho-Collados, 2018).
sst2	The Stanford Sentiment Treebank, a corpus with fully labeled parse trees for a complete analysis of the compositional effects of sentiment in language (Socher et al., 2013).
sciq	Crowd-sourced science exam questions about Physics, Chemistry, Biology, etc, in multiple-choice format with 4 answer options and an evidence-supporting paragraph for the correct answer for most questions (Welbl et al., 2017).
qnli	The Question-answering Natural Language Inference dataset is automatically derived from the Stanford Question Answering Dataset v1.1 (SQuAD) of question-paragraph pairs, where one of the sentences in the paragraph (drawn from Wikipedia) contains the answer to the corresponding question (written by an annotator). (Wang et al., 2018).
pubmedqa	A Yes/No biomedical question answering dataset collected from PubMed abstracts (Jin et al., 2019).
mnli	Often also referred to as multi-nl, this Multi-Genre Natural Language Inference (MultiNLI) corpus is a dataset to test sentence understanding; it offers data from ten distinct genres of written and spoken English–enabling evaluation on nearly the full complexity of the language and on cross-genre domain adaptation. (Williams et al., 2018)
mc_taco	13K question-answer pairs that require temporal commonsense comprehension on (1) duration of an event, (2) order of events, (3) time when event occurs, (4) event frequency, and (5) stationarity (whether a state is maintained for a very long time or indefinitely). (Zhou et al., 2019)
mathqa	A large-scale dataset of math word problems (Amini et al., 2019).
copa	The Choice Of Plausible Alternatives (COPA) dataset consists of 1000 questions composed of a premise and two alternatives, with the task being to select the alternative that more plausibly has a causal relation with the premise (Gordon et al., 2012).
boolq	A question answering dataset for Yes/No questions containing 15942 examples; each example is a triplet of (question, passage, answer), with the title of the page (from google search engine where questions are collected) as optional additional context (Clark et al., 2019).

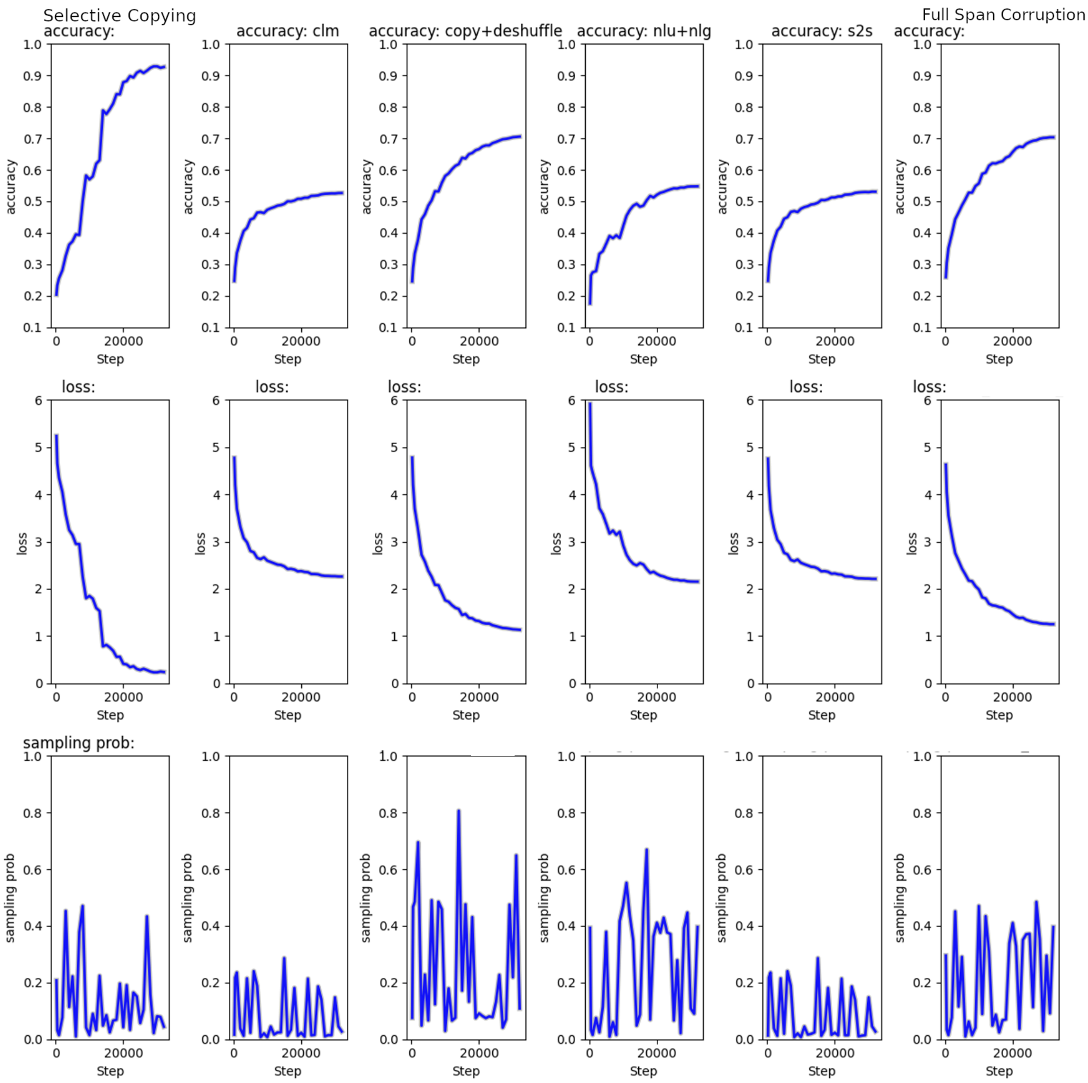

Figure 4: These plots shows how the reinforcement learning adjusts the pretraining objective mixtures in Birdie 1.4B. Objectives are arbitrarily grouped together.

Model (Objective)	avg	arc_challenge	arc_easy	medmcqa	winogrande	wic	sst2	sciq	qnli	pubmedqa	mnli	mc_taco	mathqa	copa	boolq
1.4B, Instruction Tuned															
Birdie (RL-MoD)	45.5	29.7	28.97	29.15	50.17	51.25	82.11	31.95	53.40	52.00	31.82	64.67	25.3	47.1	58.7
Attention (CLM)	43.0	31.1	29.34	28.02	49.17	50.63	87.61	38.85	51.00	40.51	31.82	40.99	26.7	46.1	50.1
Birdie (MoD)	42.5	30.2	29.09	28.87	50.59	52.35	77.64	36.40	53.16	35.29	31.82	42.47	25.2	41.2	61.4
Birdie (CLM)	40.9	29.1	28.26	27.27	50.09	50.16	54.47	35.75	49.46	48.14	31.82	34.06	25.8	47.1	61.5
1.4B, Base Models															
Birdie (MoD)	41.0	29.1	27.98	26.74	48.41	50.00	59.17	36.15	50.58	38.06	31.82	66.13	24.0	46.5	39.1
Birdie (CLM)	40.9	29.4	28.49	28.38	49.69	50.00	56.88	33.54	49.28	46.34	31.84	34.13	24.3	48.4	62.0
Birdie (Bandit)	40.6	29.6	27.61	25.96	48.08	50.00	57.91	34.95	52.13	39.00	31.83	62.79	24.3	45.7	38.8
Attention (CLM)	40.1	30.4	29.68	26.58	48.90	50.00	50.00	30.78	49.42	40.09	31.84	33.88	26.4	50.6	62.1
400M, Instruction Tuned															
Birdie (UL2)	40.3	28.6	28.39	28.30	47.85	49.84	51.03	41.31	49.37	43.14	31.82	34.58	24.0	43.9	62.0
Attention (UL2)	40.2	30.0	28.50	27.19	48.92	49.84	50.92	28.27	49.70	43.89	31.82	47.42	24.3	42.2	59.2
Hawk (PT5)	39.3	26.5	25.34	24.73	49.98	50.00	50.92	24.42	49.46	44.26	35.33	33.87	22.1	52.4	60.8
Attention (CLM)	39.2	29.4	28.54	28.54	49.36	49.06	49.08	27.28	49.42	38.97	31.82	38.35	25.7	41.0	61.8
Hawk (CLM) 38.4	26.2	26.64	26.46	50.64	50.00	50.92	23.24	49.46	37.00	31.82	33.87	21.7	47.8	62.2	
400M, Base Models															
Birdie (CLM)	40.3	27.5	27.12	28.40	49.14	50.00	49.08	39.64	49.46	47.97	31.49	33.87	23.8	44.8	62.1
Birdie (Bandit)	40.1	27.3	27.29	27.66	48.95	47.96	57.68	48.45	49.59	38.29	31.82	39.98	24.2	44.2	47.9
Attention (CLM)	39.7	29.6	28.02	27.99	49.44	48.75	58.26	32.03	50.36	44.69	31.82	34.77	24.6	43.2	53.0
Birdie (UL2)	39.5	27.8	26.94	27.96	47.72	50.63	49.08	38.27	49.55	47.74	31.81	40.52	23.2	42.2	50.0
Birdie (PT5)	39.3	27.1	27.61	27.65	49.16	51.10	49.43	38.64	50.17	37.77	31.82	44.98	24.6	40.0	49.7
Attention (UL2)	39.2	28.8	27.66	28.96	48.49	50.00	47.94	29.02	50.54	36.71	31.82	66.06	23.0	41.8	37.7
Hawk (PT5)	38.8	24.7	26.51	25.98	50.66	50.00	50.80	21.99	49.46	36.86	32.84	33.87	20.6	57.1	62.2
Hawk (CLM)	38.4	25.7	26.30	23.63	50.15	50.00	49.08	23.33	49.46	36.86	35.33	33.87	22.8	49.3	62.2

Table 5: Various model configurations (model architecture - Birdie versus transformer, size – 1.4B versus 400M), objectives (fixed MoD objectives, such as UL2 and PT5, versus our RL-resulting dynamic MoD RL-MoD, and base versus instruction-tuned, listed in Column 1, are evaluated on each of the 14 EleutherAI LM Harness tasks (listed in Columns 3-16). Column 2 shows average performance over tasks. All scores are percentages.

B