Learning to Learn: Curriculum Selection via Reinforcement Learning for Sample-Efficient Language Models

Scott Viteri

1 Overview

We propose an algorithm that enables language models to learn what data to learn from. Rather than training on randomly sampled data, the model takes actions that select training examples, receiving reward based on how much each selection improves prediction on held-out data. The core hypothesis is that learned curriculum selection will yield significantly better sample efficiency than random or heuristic curricula.

2 Technical Approach

2.1 Setup

- Language model (GPT-2) with parameters θ
- Off-the-shelf embedding model ϕ: text $\to \mathbb{R}^{d_\phi}$ for encoding streaming data (e.g. sentence-transformers)
- Streaming data source, partitioned into training candidates and held-out evaluation
- Value vector $w \in \mathbb{R}^{d_\phi}$, living in the same embedding space as ϕ
- Boltzmann temperature $\beta > 0$ controlling exploration
- Learning rate α for the LM gradient step; learning rate η for the value regression step
- Reward scaling factor c (raw rewards $r_k \sim O(10^{-3})$ are multiplied by c before value regression)
- Gradient clipping threshold G (max global norm; all LM gradient steps are clipped to $\|\nabla\| \leq G$)

2.2 Data Streaming

Data streams in continuously. At each outer step, a fresh batch of context windows arrives and is partitioned:

- $\sim$900 become training candidates (the set S_t at outer step t)
- $\sim$400 are reserved for held-out evaluation (the set D_t at outer step t)

Each incoming data point x is embedded via an off-the-shelf embedding model: $e_x = \phi(x)$. These embeddings are used for the selection mechanism described below.

Each element of S_t or D_t is conceptually one full transformer context window of text.

2.3 Algorithm

The algorithm alternates two update steps—coordinate descent on the free energy. At each outer-loop stream step t, we refresh S_t, D_t from newly arrived data and hold them fixed for the duration of the inner loop. We then run K inner-loop steps on fixed S_t, D_t.

The policy over training candidates is the Boltzmann distribution:

$$\pi(x) = \mathrm{softmax}\Big(\frac{w \cdot \phi(x)}{\beta}\Big), \qquad x \in S_t, \tag{1}$$

where $w \in \mathbb{R}^{d_\phi}$ is a learned value vector in embedding space and β is the temperature. The estimated value of candidate x is $V(x) = w \cdot \phi(x)$.

At initialisation, $w = 0$, so the policy is uniform over S_t. The value vector is updated by online linear regression toward observed rewards, causing the policy to concentrate on candidates predicted to yield high held-out improvement.

Each inner step k proceeds as follows:

1. **Select a training example (E-step).** Sample $x_k \sim \pi$ from the candidates S_t using the current value vector w.
2. **Language-modeling update (M-step).**
$$\theta_{k+1} = \theta_k + \alpha \nabla_{\theta_k} \log p_{\theta_k}(x_k). \tag{2}$$
This is a standard gradient ascent step on the log-likelihood of the selected example. Note that we only need to cache the held-out log-probability $\frac{1}{|D_t|} \log p_{\theta_k}(D_t)$ from the previous step, not the full parameter snapshot θ_k.
3. **Reward.** The reward is the one-step held-out improvement:
$$r_k = \frac{1}{|D_t|} \log p_{\theta_{k+1}}(D_t) - \frac{1}{|D_t|} \log p_{\theta_k}(D_t). \tag{3}$$
In practice, r_k is scaled by a constant c (since raw $r_k \sim O(10^{-3})$) before the value update.
4. **Value vector regression.** Update w by online linear regression toward the observed (scaled) reward:
$$w \;\leftarrow\; w - \eta \big(w \cdot \phi(x_k) - c\, r_k\big)\, \phi(x_k). \tag{4}$$
This drives the predicted value $w \cdot \phi(x_k)$ toward the observed reward, so that future sampling via π favours candidates whose embeddings are similar to those that previously yielded high held-out improvement.

Key simplifications. This formulation requires no Q-function, no value baseline, no Bellman loss, no parameter snapshots, and no backpropagation through the selection policy. The value vector w is updated by a closed-form regression step that is decoupled from the LM optimiser. The only neural-network gradient computation is the standard LM training step (step 2).

2.4 Interpretation as Free-Energy Minimisation

The Boltzmann policy $\pi(x) \propto \exp(w \cdot \phi(x)/\beta)$ can be viewed as the solution to a free-energy minimisation problem:

$$\pi^* = \arg\min_{\pi} \Big[-\mathbb{E}_{x\sim\pi}\big[V(x)\big] + \beta\, H(\pi) \Big], \tag{5}$$

where $H(\pi)$ is the entropy of the selection distribution. The first term encourages selecting high-value candidates; the second (controlled by β) encourages exploration. The value regression step (step 4) updates $V(x) = w \cdot \phi(x)$ to better predict which candidates yield held-out improvement, and the Boltzmann form automatically adjusts the policy accordingly.

3 Evaluation

3.1 Primary Metric: Sample Efficiency

The central question is: how many training examples does the model need to reach a given performance level?

We will measure:

- Perplexity on a fixed evaluation set as a function of training examples seen
- Performance on downstream tasks (e.g., MMLU, HellaSwag) as a function of training examples
- Comparison to human learning efficiency as an aspirational benchmark

3.2 Baselines

- **Random curriculum:** Uniform sampling from candidates
- **Loss-based curriculum:** Prioritize high-loss examples
- **Uncertainty-based curriculum:** Prioritize examples with high model uncertainty
- **Competence-based curriculum:** Examples ordered by difficulty (requires difficulty labels)

3.3 Analysis

Beyond aggregate metrics, we will examine:

- **Curriculum structure:** Does the learned curriculum exhibit interpretable patterns? Developmental stages? Topic clustering?
- **Exploration dynamics:** How does the policy entropy evolve over training? Does the Boltzmann temperature β produce reasonable exploration?
- **Value vector interpretability:** What directions does w learn? Can we understand what makes a training example "valuable" by examining $w \cdot \phi(x)$ across candidates?

4 Relation to Prior Work

4.1 Curriculum Learning

Graves et al. (2017) use multi-armed bandits to select tasks; Jiang et al. (2018) train a separate "mentor" network to weight examples. Our approach differs by embedding selection in the model's forward pass and using RL to optimize for held-out improvement directly.

4.2 Meta-Learning

MAML (Finn et al., 2017) learns initializations for fast adaptation. We share the computational structure—reasoning about the effect of gradient updates—but learn *what* to train on rather than *where* to start.

4.3 Intrinsic Motivation

Our objective relates to compression progress (Schmidhuber) and active inference, but avoids the memory requirements of the former and the dark room problem of the latter by using streaming held-out data as a proxy for predictive success.

4.4 Connection to My Prior Work

This proposal builds directly on my recent work on Markovian Transformers for Informative Language Modeling (`https://arxiv.org/abs/2404.18988`). That work introduces a framework for training language models to generate Chain-of-Thought reasoning that is *causally load-bearing*—the CoT must contain all information needed to predict the answer, as the model cannot attend back to the original question.

Both projects share a common structure: use RL to learn intermediate representations that improve prediction on held-out data. In the Markovian Transformers work, we learn CoTs:

$$\text{Question} \to \text{CoT} \to \text{Answer}$$

In the current proposal, we learn curricula:

$$\text{Model State} \to \text{Selected Data} \to \text{Improved Predictions}$$

The Markovian Transformers work demonstrates that this general approach is tractable and yields large gains on reasoning benchmarks (e.g., GSM8K: 19.6% → 57.1%). The current proposal extends this framework from learning *what to say* to learning *what to study.*

5 Broader Motivation

Language models trained to predict text develop remarkable capabilities from a simple objective. Yet they require extensive post-training to behave agentically and arguably lack a kind of global coherence. One hypothesis: the training process is purely observational—the model never takes actions that affect what it observes.

This project is a stepping stone toward studying whether learned curriculum selection produces qualitatively different agents. The immediate goal is demonstrating sample efficiency gains. The longer-term question is whether controlling one's own learning process contributes to the coherence and agency that current models seem to lack.

5.1 Long-Term Direction: Formalizing Homeostasis

Biological agents are shaped by survival pressures. Hunger, pain, and fatigue are not arbitrary reward signals—they are tied to the organism's continued existence. Current approaches to intrinsic motivation (curiosity, empowerment, compression progress) capture aspects of adaptive behavior but lack this grounding in self-preservation.

A long-term goal of this research program is to formalize homeostasis and survival into a simple, biologically plausible metric that could serve as a foundation for intrinsic motivation in artificial systems. The current project—learning to select data that improves future prediction—is a minimal step in this direction: the agent takes actions that maintain its predictive capacity, a kind of epistemic homeostasis. Understanding what this simple case yields will inform more ambitious formulations.

6 Timeline and Resources

6.1 Compute Resources

- **Initial experiments:** Stanford Sherlock cluster (covered by existing allocation)
- **Final scaling experiments:** Cloud compute (Runpod or similar), approximately $10K for H100/H200 time to validate results on larger models (1B+ parameters)

Phase	**Duration**
Infrastructure and baseline implementation	6 weeks
Core algorithm implementation and debugging	8 weeks
Initial experiments and hyperparameter tuning	6 weeks
Scaling experiments on larger models	4 weeks
Analysis and writeup	4 weeks
Total	**6 months**

7 Expected Outcomes

1. Empirical demonstration of sample efficiency gains (or well-documented negative result)
2. Open-source implementation of the algorithm
3. Analysis of learned curriculum structure
4. Foundation for future work on agency, homeostasis, and learned exploration in language models

8 About the Applicant

Scott Viteri is a final-year PhD student at Stanford working on machine learning and reinforcement learning, with a focus on continual learning in transformers. His relevant prior work includes:

- **Markovian Transformers for Informative Language Modeling** (`https://arxiv.org/abs/2404.18988`): Introduces a framework for training language models with RL to produce causally load-bearing Chain-of-Thought reasoning. Demonstrates large gains on reasoning benchmarks (GSM8K, ARC-Challenge, MMLU). This work establishes the technical foundation for the current proposal.
- **Epistemic Phase Transitions in Mathematical Proofs** (Cognition, 2022): Studies how belief formation operates in mathematical reasoning using network models, showing how modular structure and bidirectional inference enable certainty to emerge despite local error rates.
- **ARC Eliciting Latent Knowledge Prize winner**: Recognized for work on methods to extract truthful information from language models.
- Experience with category-theoretic approaches to machine learning, collaborating with researchers at the Topos Institute.