ANCHORED ADVERSARIAL FINE-TUNING OF LANGUAGE MODELS WITH GUMBEL-SOFTMAX

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models trained with supervised fine-tuning generate the next token based on ground-truth tokens during training but on their own predictions during inference, introducing a training–inference mismatch that can cause errors to compound over a generated sequence. We investigate generative adversarial fine-tuning as a means of introducing autoregressively generated response segments into language-model training, exposing the model to its own predictions during training. To permit gradient propagation to the generator, we leverage Gumbel-Softmax sampling rather than standard discrete token sampling to obtain a differentiable next-token representation, allowing generative adversarial network-based training. Using a convolutional discriminator, which distinguishes generated responses from reference ones, and a transformer language model as the generator, we attempt to improve training stability and response quality through the use of regularization techniques. In particular, during training we progressively unfreeze the discriminator's convolutional layers and optimize the generator using a weighted combination of adversarial and standard supervised fine-tuning objectives. Using an LLM-based evaluator, our best anchored configuration improves over the supervised fine-tuning baseline in all three evaluated categories, including relative gains of 6.5% and 9.9% in Summarization and Science, respectively. These results provide preliminary evidence that the adversarial generation objective can complement supervised fine-tuning in the studied setting.

1 INTRODUCTION

The fine-tuning of large language models (LLMs) is a vital stage in their training. During generation, LLMs predict the next token conditioned on their previously predicted tokens, unlike during supervised fine-tuning (SFT), which conditions next-token predictions on ground-truth preceding tokens. This creates a training–inference mismatch. Secondarily, previous work shows that supervised fine-tuning can produce highly predictable, formulaic responses Yun et al. (2025); Klypa & Cherednichenko (2026). Motivated by the above, we evaluate the use of autoregressively generated response segments during training as a method for improving response quality.

We seek primarily to address the training–inference mismatch by introducing autoregressive generations during training. In doing so, we attempt to improve the overall generation quality of language models. To optimize the model on these self-generated segments, however, we require a sequence-level training signal. Borrowing intuition from generative adversarial networks (GANs), we score generated text using a learned prediction of the probability that the text is human-written. This provides a sequence-level signal with which to score and optimize generated responses. Indeed, similar ideas have been explored previously, to varying degrees of success and adherence to the original GAN architecture (Li et al., 2017; Zhang et al., 2023; Yu et al., 2017; Wu et al., 2026).

Much of the previous work, such as the SeqGAN framework of Yu et al. (2017) and the SGALM framework of Wu et al. (2026), is shaped by the discrete nature of text sampling. The discrete nature of next-token sampling makes the evaluation of multi-token generated sequences a nontrivial task. In particular, autoregressive generation depends on previously sampled tokens, and discrete token selection interrupts the pathwise gradient from later generated tokens back to the model parameters. Since the token-selection step required for text generation is discrete, standard

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

backpropagation cannot pass directly through that sampling operation. Policy-gradient methods provide one alternative by estimating gradients without differentiating through the sampled token. Such score-function estimators can have high variance; methods such as SeqGAN additionally use Monte Carlo rollouts to estimate intermediate sequence rewards.

To obtain a direct surrogate gradient through discrete token selection, we leverage Gumbel-Softmax sampling (Jang et al., 2017). Gumbel-Softmax sampling provides a smooth and differentiable relaxation from an output logit distribution to a next-token representation. This permits standard backpropagation and can provide a lower-variance gradient approximation than score-function methods such as REINFORCE, at the cost of bias from the relaxation (Maddison et al., 2017). The use of Gumbel-Softmax sampling to permit autoregressive generation during training has been attempted before, particularly in Zhang et al. (2023), which reports substantial training instability. Hence, we also integrate several stabilization techniques into the text-GAN setting, including learning-rate annealing and progressive training of the discriminator. We also use SFT loss anchoring, described further in Section 3. Our contributions can be summarized by the following combination:

1. An adversarial fine-tuning framework for transformer language models that uses Gumbel-Softmax to propagate a surrogate gradient through autoregressive generation.
2. An SFT-anchored generator objective intended to limit potential drift under adversarial training.
3. Integration of stabilization techniques, including learning-rate annealing and progressive discriminator training, into this adversarial fine-tuning setting.

2 Related Work

Our work draws extensively from previous research and more recent innovations, both in natural language processing (NLP) and generative adversarial networks (GANs). Sitting at the intersection of these fields, our work combines recent innovations in both of these fields.

2.1 Language Processing

2.1.1 Generation

Language generation has been one of the most active areas of research in recent history. Much of modern language research, including our work, builds on attention-based models like those of Vaswani et al. (2017) to iteratively relate tokens and produce a probability distribution for the next token. Our work in particular uses the 360M-parameter version of Hugging Face's attention-based language model SmolLM2 Ben Allal et al. (2025). Traditional language models are trained via pretraining and supervised fine-tuning Radford et al. (2018). We similarly use a pretrained language model but investigate an alternative fine-tuning procedure for the latter stage.

2.1.2 Processing and Classification

Similarly, text classification has been a highly active area in modern NLP research. convolutional neural networks (CNNs), due to their structure-preserving nature, have demonstrated great promise in this field of research Johnson & Zhang (2015), Kim (2014). Convolutional networks as discriminators in GANs have been similarly well studied, their multidimensional image-based versions implemented in the work of Radford et al. (2016) and in text by Yu et al. (2017). In particular, we build on the idea of parallel, pooled convolutions for text, whose efficacy was demonstrated by the work of Kim (2014).

2.2 Adversarial Learning Works

Adversarial training in language models is not a completely novel concept, and has been explored previously in detail (Li et al., 2017; Zhang et al., 2023; Wu et al., 2026; Yu et al., 2017). While these works share broad architectural similarities, their specific training procedures differ.
Li et al. (2017) and Yu et al. (2017) use recurrent neural networks as the generator, and Li et al.

(2017) also uses a recurrent neural network in the discriminator. By contrast, other work has focused on reinforcement-style learning with either other LLMs as the discriminator or CNNs (Zhang et al., 2023; Wu et al., 2026; Yu et al., 2017; Che et al., 2017). Li et al. (2017) also combines adversarial and maximum-likelihood training objectives, providing a precedent for retaining supervised training during adversarial optimization. Our formulation incorporates the SFT term directly into a weighted joint generator objective rather than alternating between objectives.

A central challenge in prior work is the difficulty of propagating gradients through discretely sampled tokens during autoregressive generation. Policy-gradient methods, such as SeqGAN (Yu et al., 2017), avoid differentiating through the sampled token but can produce high-variance score-function gradient estimates. SeqGAN additionally uses Monte Carlo rollouts to estimate intermediate sequence rewards. Other work, including Zhang et al. (2023), instead explores differentiable relaxations such as Gumbel-Softmax.

One can instead use a continuous relaxation such as Gumbel-Softmax sampling rather than discrete sampling to select the next token. This preserves a differentiable surrogate path and allows standard backpropagation through the relaxed representation (Jang et al., 2017). However, previous work that attempted this (Zhang et al., 2023; Kusner & Hernández-Lobato, 2016) noted substantial training instability. We therefore implement several stabilization methods, including progressively unfreezing the convolutional layers in the discriminator (Sun et al., 2020), learning-rate annealing, and dropout (Hinton et al., 2012).

3 Gumbel-Softmax Adversarial Fine-Tuning

As in Goodfellow et al. (2014), we use a discriminator's output as a loss criterion. Denote our generator by G_θ with parameters θ. Denote our discriminator by D_ϕ with parameters ϕ. We denote the loss functions $\mathcal{L}_{sft}, \mathcal{L}_{adv}$. Let the reference-text distribution be $\mathbb{T}$.

In our case, G_θ is an attention-based language model that takes prompt P_t and outputs next-token logits z_t. Standard supervised fine-tuning trains G_θ alone to optimize:

$$\mathcal{L}_{sft} = \frac{-1}{N} \sum_{i=1}^{N} \sum_{j=1}^{V} 1_{i,j} \log(G_\theta(P \oplus R_{:i-1})_j).$$

We take $R_{1:i-1}$ to be the first $i-1$ tokens of the ground-truth response, V is the vocabulary size, and $1_{i,j}$ is an indicator variable that is 1 if and only if the ith response token is j. This trains the model using token-level teacher forcing. We hypothesize that the addition of an autoregressive generation component can improve overall response quality by reducing the training-inference mismatch described previously.

To train on the generation component, we require a sequence-level signal for that generated segment. Because response quality is difficult to score directly, we combine the generative adversarial network and large language model architectures to score generated segments by their probability of being human-written, unlike standard token-level SFT. Our discriminator D_ϕ is, as described in Kim (2014), a one-dimensional text-based convolutional neural network with a linear classification head. A writing sample is passed to D_ϕ, which outputs a classification logit. After sigmoid activation, $\sigma(D_\phi(x))$ is the discriminator's estimate of the likelihood that $x \sim \mathbb{T}$. A vanilla implementation of a standard GAN in text can be expressed as follows:

$$\boldsymbol{r_0} = P \; z_0 = S(\text{Softmax}(G_\theta(P)))$$

with sampling function $S(logits) \rightarrow$ Token ID $\in \mathbb{Z}^+_{\leq V}$. In standard LLMs, this sampling function $S \equiv$ argmax when deterministic, or a weighted random sample when not. One can then obtain the recurrence:

$$\boldsymbol{r_t} = \boldsymbol{r_{t-1}} \oplus z_t, \; z_t = S(\text{Softmax}(G_\theta(\boldsymbol{r_{t-1}})))$$

for $t \in 1, 2, 3, ..., N$ for some capped response length N. Then, adversarial loss is computed on the prompt-response pair $\boldsymbol{r_N}$. However, S is a hard categorical selection and is non-differentiable with respect to its input logits. This breaks the direct pathwise gradient through the sampled token,

preventing standard backpropagation through the autoregressive sampling operation.

Following Zhang et al. (2023), we leverage Gumbel-Softmax sampling to produce differentiable relaxations allowing for the computation of biased but low-variance pathwise gradient estimators for training. However, our method differs in its stabilization strategy. In particular, to help regulate instability during training, we propose implementing progressive unfreezing of the convolutions of D_ϕ, performance-based learning-rate annealing, the use of dropout and SFT anchoring. The first is implemented in a manner conceptually aligned to that of Sun et al. (2020), but instead in text. D_ϕ is a convolutional neural network with linear classification head. After pretraining (at the beginning of adversarial training) all convolutional layers of D_ϕ are frozen, while only the classification head remains unfrozen. Let $\sigma(x) = \frac{1}{1+e^{-x}}$ be the sigmoid activation function. When:

$$\text{avg}_{100}(\sigma(D_\phi(x \sim \mathbb{T})) - \sigma(D_\phi(\boldsymbol{r_n}))) < \alpha,$$

for some fixed α (empirically, $\alpha = 0.25$) we unfreeze the convolutional layer with the largest kernel size that remains unfrozen, where avg_{100} is the average over the last 100 examples. This is motivated by the desire for D_ϕ to learn broader n-gram patterns in generated text before finer-grained patterns, while limiting its early capacity relative to G_θ. We similarly describe our learning rate at time step $t+1$ as a function of learning rates at time step t. We distinguish two cases. If D_ϕ is fully unfrozen, then

$$\eta_{G,t+1}, \eta_{D,t+1} = Update_1(\eta_{G,t}, \eta_{D,t}) = \begin{cases} 0.8\eta_{G,t}, 1.2\eta_{D,t} \text{ iff } \text{avg}_{100}(\sigma(D_\phi(x \sim \mathbb{T})) - \sigma(D_\phi(\boldsymbol{r_n}))) < \beta \\ \chi\eta_{G,t}, \chi\eta_{D,t} \text{ iff } \beta < \text{avg}_{100}(\sigma(D_\phi(x \sim \mathbb{T})) - \sigma(D_\phi(\boldsymbol{r_n}))) < \gamma \\ 1.2\eta_{G,t}, 0.8\eta_{D,t} \text{ else.} \end{cases}$$

Else:

$$\eta_{G,t+1}, \eta_{D,t+1} = Update_2(\eta_{G,t}, \eta_{D,t}) = \begin{cases} \chi\eta_{G,t}, \chi\eta_{D,t} \text{ iff } \text{avg}_{100}(\sigma(D_\phi(x \sim \mathbb{T})) - \sigma(D_\phi(\boldsymbol{r_n}))) < \gamma \\ 1.2\eta_{G,t}, 0.8\eta_{D,t} \text{ else.} \end{cases}$$

For fixed constants β, γ, χ. Empirically we use $\beta = 0.25, \gamma = 0.45, \chi = 1 - 10^{-5}$. This adaptive learning-rate schedule is intended to reduce large imbalances between discriminator and generator performance during training. We impose a 100-step cooldown between non-decay learning-rate updates. We also apply this to progressive unfreezing—no two convolutions of D_ϕ are allowed to be unfrozen within 100 steps. We also apply dropout to the discriminator at rate $p = 0.1$.

Additionally, rather than train D_ϕ on token vectors $\boldsymbol{r_N}$, we train D_ϕ on $E(\boldsymbol{r_N})$ where $E : \mathbb{R}^{N*V} \mapsto \mathbb{R}^{N*960}$ is the G_θ's embedding layer. Both reference and generated responses are converted to dense embeddings prior to being passed to D_ϕ. We use this to mitigate the likelihood that D_ϕ learns to discriminate the difference between hard one-hot vectors, present in real responses and sampled vectors.

Our final modification is to integrate the SFT loss into the total generator objective. This is motivated by the fact that D_ϕ is not explicitly trained to judge factual or task-specific correctness. We therefore "anchor" the generator objective to the SFT loss so that adversarial training supplements rather than replaces token-level supervision. Let $\mathcal{R}(x) = \begin{cases} 1 \text{ iff } x \sim \mathbb{T} \\ 0 \text{ else} \end{cases}$. Our loss formulation is shown below, leveraging binary cross-entropy loss for its adversarial component, where B is the batch size.

$$\mathcal{L}_{total,D} = \frac{-1}{B} \sum_{i=1}^{B} \left[\mathcal{R}(batch_i) \log(\sigma(D_\phi(\boldsymbol{r_{N,i}}))) + (1 - \mathcal{R}(batch_i)) \log(1 - \sigma(D_\phi(\boldsymbol{r_{N,i}}))) \right].$$

$$\mathcal{L}_{total,G} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{sft}\mathcal{L}_{sft}$$

$$= \lambda_{adv} \frac{-1}{|B_{fake}|} \sum_{i=1}^{B} (1 - \mathcal{R}(batch_i)) \log(\sigma(D_\phi(\boldsymbol{r_{N,i}}))) + \lambda_{sft} \frac{-1}{NB} \sum_{k=1}^{B} \sum_{i=1}^{N} \sum_{j=1}^{V} 1_{i,j} \log(G_\theta(P \oplus R_{:i-1})_j).$$

Intuitively, the $\mathcal{L}_{adv}$ term pressures G_θ to reproduce characteristics of human-written reference responses, while $\mathcal{L}_{sft}$ provides token-level supervision anchoring the model to the target response distribution.

Algorithm 1 Fine-Tuning Stage Using Regularization and Gumbel-Softmax Token Sampling

Hyperparameters: $\tau = 1$, $\eta_{G_0} = \eta_{D_0} = 2 \cdot 10^{-4}$, Batch size B, maximum response length N
1: Generator G, Discriminator D, losses $\mathcal{L}_{total,G} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{sft}\mathcal{L}_{sft}, \mathcal{L}_{total,D}$
2: **for** Number of Epochs **do**
3: **for** batch in Data with index t **do**
4: Freeze G_θ, Unfreeze D_ϕ. $T = \langle\rangle$. Populate T with B random bits. Set $responses = \langle\rangle$
5: **for** $x \in \mathbb{Z}^{\geq 0} < B$ **do**
6: $P = B_{\text{x,prompt}}$, $R = B_{\text{x,response}}$
7: **if** $T_x = 0$ **then**
8: Generate R' iteratively from P using Gumbel-Softmax token selection, pass each output into embedding and compile into R'
9: $responses \oplus R'$
10: **else**
11: $responses \oplus Embedding(R)$
12: **end if**
13: **end for**
14: Compute $\hat{y} = D_\phi(responses)$ and $\mathcal{L}_{\mathcal{D}} = BinaryCrossEntropy(\hat{y}, T)$.
15: Calculate the parameter update ϕ'. Set $\phi \leftarrow \phi + \phi'$.
16: Freeze $D_{\phi'}$, Unfreeze G_θ. Repeat lines $5 - 13$.
17: Get $D_{\phi'}(responses)$. Calculate $\mathcal{L}_{adv}, \mathcal{L}_{sft}, \mathcal{L}_{total,g}$ as described above
18: Obtain the parameter update θ' from the optimizer and set $\theta \leftarrow \theta + \theta'$
19: **if** D_ϕ is not fully unfrozen **then** $\eta_{G,t+1}, \eta_{D,t+1} = Update_2(\eta_{G,t}, \eta_{D,t})$
20: Unfreeze another layer if $\text{avg}_{100}(\sigma(D_\phi(x \sim \mathbb{T})) - \sigma(D_\phi(\boldsymbol{r_n}))) < \alpha$
21: **else**
22: $\eta_{G,t+1}, \eta_{D,t+1} = Update_1(\eta_{G,t}, \eta_{D,t})$
23: **end if**
24: **end for**
25: **end for**

Empirically, we use Hugging Face's 360M-parameter SmolLM2 model as G_θ (Ben Allal et al., 2025). We use a discriminator architecture with five parallel convolutions of kernel sizes $\{2, 3, 4, 5, 6\}$ and a three-layer linear classifier. To prevent double embedding, already-embedded prompts and generated token embeddings are concatenated and directly passed to the attention layers. A summary of training our complete algorithm using these regularization techniques is provided in Algorithm 1.

4 Theoretical Analysis

We seek to analyze several specific design properties of our approach. In this section, we show that the soft-relaxed adversarial loss path is differentiable, that the fixed-discriminator generator subproblem approaches stationarity in gradient norm under standard assumptions, and that SFT loss anchoring constrains local drift from the SFT optimum. We use the following simplifications:

1. We analyze the soft Gumbel-Softmax relaxation. Empirically, we use the straight-through estimator, whose backward pass uses the corresponding soft relaxation.
2. We assume both models G_θ, D_ϕ produce differentiable outputs with respect to θ, ϕ, respectively.
3. For the stationarity analysis, we assume deterministic gradient descent with a step size satisfying the stated smoothness condition. Empirically, we use Adam.
4. We assume the model has finite initial parameters and finite initial logits.

4.1 Differentiability of Adversarial Loss Path

A summary of the path for loss calculation on any prompt P is as follows:

$$G_\theta(P_t) \to y_t = GumbelSoftmax_\tau(G_\theta(P_t)) \to y_t E \to r_t = r_{t-1} \oplus y_t E, P_{t+1} = P_t \oplus y_t E$$

for $t = 1, 2, 3, \ldots, N$ for fixed response length N, with $r_0 = \langle\rangle$ and prompt P_0. We take

$$\mathcal{L}_{adv} = BinaryCrossEntropy(D_\phi(r_N), 1).$$

Then, $\nabla_\theta \mathcal{L}_{adv} = \nabla_\theta(BCE(D_\phi(r_N), 1))$ depends only on

$$\frac{\partial BCE(D_\phi(r_n), 1)}{\partial r_N} \cdot \frac{\partial r_N}{\partial \theta}.$$

Let $F_t(\theta, P_t)$ denote $GumbelSoftmax_\tau(G_\theta(P_t))E$. One can observe that $y_t E = F_t(\theta, P_t)$. We thus are interested in $\frac{\partial y_t E}{\partial \theta}$. P_t, by definition, is $P_0 \oplus r_{t-1}$. Hence, $F_t(\theta, P_t) = F_t(\theta, P_0 \oplus r_{t-1})$. Thus, changing θ can either directly change $F_t(\theta, P_t)$ as the next token itself changes, or indirectly change $F_t(\theta, P_t)$ as F_t also depends on the previous tokens. We can then derive that:

$$\frac{\partial y_t E}{\partial \theta} = \frac{\partial F_t}{\partial \theta} + \frac{\partial F_t}{\partial r_{t-1}} \cdot \frac{\partial r_{t-1}}{\partial \theta}.$$

A simple induction on t tells us that the second term is always defined provided the first is. The first term can be expressed with the chain rule:

$$\frac{\partial F_t}{\partial \theta} = \frac{\partial y_t E}{\partial y_t} \cdot \frac{\partial y_t}{\partial z_t} \cdot \frac{\partial z_t}{\partial \theta}.$$

$\frac{\partial z_t}{\partial \theta}$ exists by assumption and $\frac{\partial y_t E}{\partial y_t}$ exists because multiplication by E is linear. The Jacobian $\frac{\partial y_t}{\partial z_t}$ is defined for finite $\tau > 0$, as proved in Appendix A. Thus, by the chain rule, the total derivative path exists as each of its stepwise derivatives is defined.

Hence, $\frac{\partial F_t}{\partial \theta}$ exists at all indices and thus

$$\nabla_\theta r_N = \langle \frac{\partial y_1 E}{\partial \theta}, \frac{\partial y_2 E}{\partial \theta}, \frac{\partial y_3 E}{\partial \theta}, ..., \frac{\partial y_N E}{\partial \theta}, \rangle$$

is defined, allowing us to conclude that $\nabla_\theta \mathcal{L}_{adv}$ exists by the above, provided the discriminator and BCE loss are differentiable with respect to their inputs, as assumed. It is important to note that this is an idealized version of our experimental setup. Straight-through gradient estimation produces a one-hot token vector but computes the gradient from the corresponding soft relaxation. The straight-through estimator therefore uses the soft-relaxation Jacobian as a surrogate backward gradient for a hard forward sample, providing a pathwise-style estimator at the cost of bias.

4.2 Stationarity of G_θ given fixed D_ϕ

In this section we seek to show that, for fixed D_ϕ, deterministic gradient descent on the generator objective approaches stationarity in gradient norm, using the standard assumptions of L-smoothness and a lower-bounded loss. Let $\mathcal{L}_{total,G}(\theta)$ denote the total generator loss as a function of the generator parameters θ, with D_ϕ held fixed. We use the standard update

$$\theta_{t+1} = \theta_t - \eta_G \nabla_\theta \mathcal{L}_{total,G}(\theta_t),$$

where $0 < \eta_G < 2/L$. In conjunction with the assumption of $L - smoothness$, one can derive the standard gradient descent lemma for our case

$$\mathcal{L}_{total,G}(\theta_{t+1}) \leq \mathcal{L}_{total,G}(\theta_t) - \eta_G \left(1 - \frac{\eta_G L}{2}\right) ||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2^2.$$

where $|| \cdot ||_2^2$ is the square of the ℓ_2 norm. Rearranging we obtain that

$$||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2^2 \leq \left(\frac{2}{2\eta_G - \eta_G^2 L}\right) (\mathcal{L}_{total,G}(\theta_t) - \mathcal{L}_{total,G}(\theta_{t+1})).$$

Now consider

$$\sum_{t=0}^{k} ||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2^2.$$

Noting that $\left(\frac{2}{2\eta_G - \eta_G^2 L}\right)$ is constant, we can ascertain that

$$\sum_{t=0}^{k} ||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2^2 \leq \left(\frac{2}{2\eta_G - \eta_G^2 L}\right) \sum_{t=0}^{k} (\mathcal{L}_{total,G}(\theta_t) - \mathcal{L}_{total,G}(\theta_{t+1})),$$

clearly telescoping to

$$\sum_{t=0}^{k} ||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2^2 \leq \left(\frac{2}{2\eta_G - \eta_G^2 L}\right)(\mathcal{L}_{total,G}(\theta_0) - \mathcal{L}_{total,G}(\theta_{k+1})).$$

Since binary cross-entropy (used in $\mathcal{L}_{adv}$) and the standard SFT loss are nonnegative, $\mathcal{L}_{total,G}$ is lower bounded by 0. Under the finite-logit initialization assumption, $\mathcal{L}_{total,G}(\theta_0) < \infty$, so the right-hand side is finite. Therefore,

$$\lim_{k\to\infty} \sum_{t=0}^{k} ||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2^2$$

is upper bounded by a finite value. Since this is a convergent series of nonnegative terms,

$$\lim_{t\to\infty} ||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2^2 = 0 \implies \lim_{t\to\infty} ||\nabla_\theta \mathcal{L}_{total,G}(\theta_t)||_2 = 0.$$

Thus, under these idealized assumptions, the fixed-discriminator generator subproblem approaches stationarity in gradient norm.

4.3 Effect of Loss Anchoring on Drift from SFT

This section attempts to quantify the impact of loss anchoring on our model's stability. Take $\theta^\star$ to be a stationary point of $\mathcal{L}_{total,G}(\theta)$ and $\theta^\dagger$ to be a local optimum of $\mathcal{L}_{sft}(\theta)$. We assume that, in a neighborhood containing $\theta^\star$ and $\theta^\dagger$, $\mathcal{L}_{sft}(\theta)$ is μ-strongly convex, $||\nabla_\theta \mathcal{L}_{adv}(\theta)|| \leq C$ for some positive real constant C, and $\lambda_{sft} > 0$.

The definition of $\theta^\star$ tells us that

$$\nabla_\theta(\mathcal{L}_{total,G}(\theta^\star)) = \lambda_{adv}\nabla_\theta \mathcal{L}_{adv}(\theta^\star) + \lambda_{sft}\nabla_\theta \mathcal{L}_{sft}(\theta^\star) = 0.$$

Rearranging we obtain

$$||\nabla_\theta \mathcal{L}_{sft}(\theta^\star)|| = \frac{\lambda_{adv}}{\lambda_{sft}}||\nabla_\theta \mathcal{L}_{adv}(\theta^\star)|| \leq \frac{\lambda_{adv}}{\lambda_{sft}}C \text{ (by the local gradient bound).}$$

By strong monotonicity of the gradient of a μ-strongly convex function,

$$\mu||\theta^\star - \theta^\dagger||_2^2 \leq (\nabla_\theta \mathcal{L}_{sft}(\theta^\star) - \nabla_\theta \mathcal{L}_{sft}(\theta^\dagger)) \bullet (\theta^\star - \theta^\dagger).$$

Because $\nabla_\theta \mathcal{L}_{sft}(\theta^\dagger) = 0$, Cauchy–Schwarz gives

$$\mu||\theta^\star - \theta^\dagger||_2^2 \leq ||\nabla_\theta \mathcal{L}_{sft}(\theta^\star)||_2 \cdot ||\theta^\star - \theta^\dagger||_2.$$

Here, $v_1 \bullet v_2$ denotes the inner product of v_1 and v_2. Simplifying we note that, unless $\theta^\star = \theta^\dagger$, in which case there is no drift from the SFT optimum, we have

$$||\nabla_\theta \mathcal{L}_{sft}(\theta^\star)||_2 \geq \mu||\theta^\star - \theta^\dagger||_2.$$

Combining this with the above, we can ascertain

$$\mu||\theta^\star - \theta^\dagger||_2 \leq \frac{\lambda_{adv}}{\lambda_{sft}}C \implies ||\theta^\star - \theta^\dagger||_2 \leq \frac{\lambda_{adv}C}{\lambda_{sft}\mu}.$$

Under the stated local assumptions, this demonstrates that increasing λ_{sft} relative to λ_{adv} tightens the upper bound on drift from the SFT optimum at such a stationary point.

5 Empirical Setup and Results

5.1 Training Setup and Hyperparameters

We implement each variation of our fine-tuning approach used in our empirical evaluation on a single NVIDIA A100 GPU for one epoch. We use Dolly-15k as our fine-tuning dataset, chosen due to its medium size and ease of use (Ouyang et al., 2022). Similarly, our supervised fine-tuning

implementation trains on a single NVIDIA A100 GPU for five epochs of training. Based on preliminary validation during model development, we selected the five-epoch SFT checkpoint as our baseline because it provided the strongest observed SFT performance. The corresponding validation results are reported in Appendix B. Due to memory constraints, both models are trained to a maximum output length of 64 tokens, the maximum length used during evaluation.

During training, we leverage straight-through Gumbel-Softmax to mimic discrete forward-pass generation conditions. It is important to note that straight-through Gumbel-Softmax provides $biased$ gradient estimation; during the forward pass the output vector is a hard, one-hot sample from the Gumbel-Softmax distribution, but during backpropagation, the gradient is computed from the corresponding soft representation.

As described above, the performance threshold for unfreezing is $\text{avg}_{100}(\sigma(D_\phi(x \sim \mathbb{T})) - \sigma(D_\phi(\boldsymbol{r_n}))) < 0.25$, and learning-rate annealing occurs when the above quantity is outside the range $(0.25, 0.45)$. We empirically use the default $\tau = 1.0$ for Gumbel-Softmax sampling.

5.2 Empirical Evaluation

During evaluation, we compare variations of our architecture for G_θ with baseline SFT-trained models. In particular, we evaluate the impact of varying $\lambda_{adv}, \lambda_{sft}$, and the use of learning-rate annealing.

Due to the difficulty of scoring text quantitatively, we use an LLM-as-a-judge framework, shown by previous research to be successful Liu et al. (2023). To be able to provide more granular data, we generate a random set of 50 prompts for each of three categories: Science, Summarization, and Synthesis (creative writing), covering a wide range of tasks. The summarization prompts ask our models and the baseline to explain a topic, the science prompts are basic science questions, and the synthesis prompts ask our models to write a story about a topic. We prompt the LLM (OpenAI's GPT-5.6) to separately score the prompt-response pairs on quality and coherence on a scale of $1 - 10$. To limit the potential impact of random sampling during inference, we use greedy decoding, a deterministic method, to select the next token.

Some previous work, such as Wang et al. (2024), has shown that LLMs can be biased evaluators of text. As such, to limit potential noise in the LLM's evaluation, each pair of responses for a prompt is passed to the LLM ten times in each ordering, and the average is taken across the 20 scoring evaluations. A model's final score for each category is the average of the 50 per-prompt mean scores. Additionally, to contextualize the scoring for each response, the SFT baseline response is passed along with the adversarial prompt-response pair. To prevent ordering bias, the model scores both responses but we use the baseline SFT score, calculated independently, for better comparability between models.

Before we outline our results, we note that, due to computational constraints, our results are exploratory in nature. In that regard, our empirical results function primarily as a proof of concept rather than a large-scale, comprehensive evaluation, due to the number of compromises made during training.

Table 1 presents the results of our study. First, we note that very high adversarial weights with limited anchoring achieve substantially worse results than standard supervised fine-tuning in several categories. This pattern is consistent with larger drift from supervised behavior, although the present experiment does not isolate that mechanism directly. Second, in the matched unanchored comparison, the annealed model substantially outperforms its unannealed counterpart across all three categories, providing limited evidence that adaptive learning-rate annealing can improve training quality in this setting.

In particular, we highlight the $0.15|0.85$ model as our main model. It achieves the best relative improvement in two of three categories, Science and Synthesis. Overall, it exceeds the standard SFT baseline mean in all three categories, achieving relative improvements of $+9.9\%, +6.5\%, +4.5\%$ in Science, Summarization, and Synthesis, respectively. We perform paired t-tests on the per-prompt

Model	Science		Summarization		Synthesis	
	Score	% Improvement	Score	% Improvement	Score	% Improvement
SFT Baseline	3.04	N/A	4.11	N/A	1.78	N/A
$1\|0$	3.16	+3.9%	3.92	-4.6%	0.76	-57.3%
$1\|0\ U$	2.49	-18.1%	3.76	-8.5%	1.19	-33.1%
$0.50\|0.50$	3.10	+2.0%	4.13	+0.5%	1.52	-14.6%
$0.25\|0.75$	3.24	+6.6%	**4.44**	**+8.0%**	1.70	-4.5%
$0.15\|0.85$	**3.34**	**+9.9%**	4.38	+6.5%	**1.86**	**+4.5%**
$0.10\|0.90$	2.90	-4.6%	4.25	+3.4%	1.71	-3.9%

Table 1: Comparison of adversarial fine-tuning configurations against a fixed supervised fine-tuning baseline across task categories. $x|y$ denotes a model trained for one epoch with adversarial weight $\lambda_{adv} = x$ and supervised fine-tuning anchoring weight $\lambda_{sft} = y$. "U" denotes a model trained without learning-rate annealing. Percent improvement is calculated relative to the fixed SFT baseline. Percent differences are rounded to the nearest tenth and raw scores to the nearest hundredth. Bolded values indicate the best result in each category.

score differences. The standard errors are $0.178, 0.187, 0.153$, giving p-values of $0.04, 0.03, 0.24$, respectively. Thus, the Science and Summarization improvements are statistically significant at the (uncorrected) 0.05 level under this test, whereas the Synthesis improvement is not. Given the exploratory scale of the study, we interpret these results as preliminary evidence that an anchored adversarial objective can complement SFT in the studied setting, rather than as evidence of broad superiority. The strongest evidence is observed in Science and Summarization, where the $0.15|0.85$ model achieves relative mean improvements of 9.9% and 6.5%, respectively.

Finally, we note that our best adversarial results tend to come from models with lower adversarial weights and higher SFT weights. We interpret this as evidence that, in the studied setting, the adversarial objective is more useful as a secondary objective that augments SFT than as a replacement for supervised fine-tuning.

6 Conclusion and Reproducibility

In this work, we present an adversarial fine-tuning framework for a language model using Gumbel-Softmax token sampling, progressive discriminator unfreezing, and learning-rate regulation. This framework permits the evaluation of autoregressively generated response segments during training and is motivated by the training–inference mismatch of standard SFT. We additionally propose anchoring the generator's learning objective to that of supervised fine-tuning to prevent excessive drift. Our theoretical analysis establishes differentiability of the soft-relaxed adversarial path and a local drift bound for the anchored objective under the stated assumptions. Empirically, our best anchored configuration achieves notably higher mean evaluation scores than the fixed SFT baseline mean across all three evaluated categories. However, our work only empirically tests the architecture at a small scale and with some empirical compromises due to compute constraints.

Computational cost is a central limitation of our work. Due to the high cost of autoregressive generation during training, our results are limited to a relatively small language model and a short maximum generation length. Further, our current implementation applies the generation component to every training batch. Future work could instead apply it to only a fraction of batches, for example one in ten. This could substantially reduce the computational overhead of autoregressive generation during training.

7 Acknowledgments, Reproducibility, and AI Usage

7.1 Acknowledgments

Hidden for blind review

7.2 Reproducibility

We take several steps to support reproducibility and reduce avoidable evaluation variance. In particular, we use deterministic sampling during model evaluation and repeatedly evaluate each prompt-response pair with the LLM evaluator. We acknowledge that LLM-based evaluation can still be variable and biased despite repeated scoring. As such, we will shortly upload all of the prompt-response pairs, as well as all code, to a public GitHub repository.

7.3 AI Usage

In this work, we used generative AI tools for:

1. Generation of synthetic datasets, as outlined in 5.1 and 5.2
2. Assistance in the writing of proofs (a polishing role). AI was not the author of any proofs in this work.
3. Providing feedback on methodology. AI did not propose any methodological steps.
4. Support for qualitative analysis.

We have *not* used generative AI tools for

1. Developing theoretical models or conceptual frameworks or formulating mathematical claims
2. Providing critical ingredients for proving mathematical claims
3. Proposing or refining hypotheses
4. Implementing methods
5. Interpreting results.

The following uses are not applicable to this work:

1. Assisting with translation
2. Cleaning and reformatting datasets

Additionally, we used generative AI tools for

1. Assisting in understanding literature
2. Polishing writing
3. Ensuring correspondence between described methodology and the code.

We have reviewed all AI-assisted work. In particular, all feedback on the methodology was reviewed for validity. AI-generated prompts were manually verified to be in the correct category. Any feedback on the proofs was also reviewed. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, et al. Smollm2: When smol goes big – data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025.

Tong Che, Yanran Li, Ruixiang Zhang, et al. Maximum-likelihood augmented discrete generative adversarial networks. *arXiv preprint arXiv:1702.07983*, 2017.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors, 2012. URL https://arxiv.org/abs/1207.0580.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Rie Johnson and Tong Zhang. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 103–112. Association for Computational Linguistics, 2015. doi: 10.3115/v1/N15-1011.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1181.

Roman Klypa and Oleksandr Cherednichenko. Diversity in large language models under supervised fine-tuning, 2026. URL https://arxiv.org/abs/2605.00195.

Matt J. Kusner and José Miguel Hernández-Lobato. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*, 2016.

Jiwei Li, Will Monroe, Tianlin Shi, et al. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2157–2169. Association for Computational Linguistics, 2017. doi: 10.18653/v1/D17-1230.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, et al. G-eval: NLG evaluation using gpt-4 with better human alignment. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153/.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables, 2017. URL https://arxiv.org/abs/1611.00712.

Long Ouyang, Jeffrey Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016. URL https://arxiv.org/abs/1511.06434.

Alec Radford, Karthik Narasimhan, Tim Salimans, et al. Improving language understanding by generative pre-training. 2018. OpenAI technical report.

Jinxuan Sun, Yang Chen, Junyu Dong, et al. Progressively unfreezing perceptual gan. *arXiv preprint arXiv:2006.10250*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008, 2017.

Peiyi Wang, Lei Li, Liang Chen, et al. Large language models are not fair evaluators. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9440–9450, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.511. URL https://aclanthology.org/2024.acl-long.511/.

Shiguang Wu, Yaqing Wang, and Quanming Yao. Self-generative adversarial fine-tuning for large language models. *arXiv preprint arXiv:2602.01137*, 2026.

Lantao Yu, Weinan Zhang, Jun Wang, et al. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 2852–2858, 2017. doi: 10.1609/aaai.v31i1.10804.

Longfei Yun, Chenyang An, Zilong Wang, Letian Peng, and Jingbo Shang. The price of format: Diversity collapse in llms, 2025. URL https://arxiv.org/abs/2505.18949.

Ze Yu Zhang, Jia Jaw Lau, Hui Zhang, et al. Fine-tuning language models with generative adversarial reward modelling. *arXiv preprint arXiv:2305.06176*, 2023.

Epoch	Validation Score
1	4.4
2	4.5
3	4.0
4	3.6
5	**4.8**
6	N/A
7	4.4
8	4.6
9	4.7
10	3.4

Table 2: Validation performance of SFT checkpoints across training epochs. The epoch-6 checkpoint was corrupted and is therefore omitted. Bold indicates the highest observed validation score among available checkpoints.

A Proof that the Gumbel-Softmax Jacobian is defined

We use the same notation as Section 4.1. We know

$$y_{t_i} = \frac{e^{(z_i+g_i)/\tau}}{\sum_k e^{(z_k+g_k)/\tau}}$$

by the definition of Gumbel-Softmax.
Put $a_i = (z_i + g_i)/\tau$. Notice that

$$\frac{\partial a_i}{\partial z_i} = \frac{1}{\tau}$$

So it suffices to compute: $\frac{\partial y_{t_i}}{\partial a_j}$.

$$y_{t_i} = \frac{e^{(z_i+g_i)/\tau}}{\sum_k e^{(z_k+g_k)/\tau}} \implies y_{t_i} = \frac{e^{a_i}}{\sum_k e^{a_k}}$$

This gives two cases. If $i = j$:

$$\frac{\partial y_{t_i}}{\partial a_j} = \frac{e^{a_i}\sum_k e^{a_k} - e^{a_i}\cdot e^{a_i}}{(\sum_k e^{a_k})^2} = y_{t_i}\cdot\frac{\sum_k e^{a_k} - e^{a_i}}{\sum_k e^{a_k}} = y_{t_i}(1-y_{t_i}).$$

If $i \neq j$

$$\frac{\partial y_{t_i}}{\partial a_j} = \frac{0\cdot\sum_k e^{a_k} - e^{a_i}\cdot e^{a_j}}{(\sum_k e^{a_k})^2} = -y_{t_i}y_{t_j}.$$

Combining the two cases gives

$$\frac{\partial y_{t_i}}{\partial a_j} = y_{t_i}(1_{i=j} - y_{t_j}).$$

Since $a_j = (z_j + g_j)/\tau$, the chain rule gives

$$\frac{\partial y_{t_i}}{\partial z_j} = \frac{1}{\tau}y_{t_i}(1_{i=j} - y_{t_j}).$$

Hence, every entry of the Jacobian

$$\frac{\partial y_t}{\partial z_t}$$

exists for finite $\tau > 0$.

B SFT validation results

These scores are on the same 1–10 scoring scale and procedure as described in Section 5.2. These results are drawn from a fixed collection of 15 summarization prompts.