

Dynamic vs. Static Adversarial Training Against GCG

A toy R2D2-style experiment on GPT-2

Angela Yang
October 2026

Motivation and Research Question. As model capabilities broaden, safety mechanisms must remain robust to adversarial inputs that adapt to the model itself. R2D2-style adversarial training addresses this by repeatedly regenerating attacks against the current model during training rather than fine-tuning on a fixed attack set [1]. This toy experiment asks:

Does dynamic adversarial training produce stronger robustness to GCG than static adversarial training when training setup and adversarial-search compute are held approximately constant?

Experimental Setup. I compared three versions of GPT-2: an untouched baseline, a statically adversarially trained model, and a dynamically adversarially trained model. The defended models were updated using LoRA. The experiment used 12 harmless “undesired” behaviors, with 8 used for training and 4 held out for evaluation. Each behavior had a one-token target and a one-token refusal target (**Nope**) to simplify loss comparisons.

Static and dynamic training used the same model initialization procedure, LoRA configuration, learning rate, train/test split, GCG search width, supervised fine-tuning (SFT) objective, and total training-time GCG optimization budget. The only intended difference was when the GCG optimization occurred:

- **Static:** optimize each training suffix once with GCG, freeze it, and train on the fixed attacks.
- **Dynamic:** divide the same GCG budget across training epochs and repeatedly re-optimize persistent suffixes against the updated model.

The total objective combined three terms:

$$\mathcal{L} = \mathcal{L}_{\text{away}} + \mathcal{L}_{\text{toward}} + \mathcal{L}_{\text{SFT}},$$

where $\mathcal{L}_{\text{away}}$ reduces probability assigned to the undesired target, $\mathcal{L}_{\text{toward}}$ increases probability assigned to the refusal, and $\mathcal{L}_{\text{SFT}}$ provides a competing signal for ordinary useful behavior.

Evaluation. I evaluated two attack settings on the four held-out behaviors:

- **Fixed-transfer:** GCG suffixes were optimized once against baseline GPT-2 and then applied unchanged to all three models.
- **Fresh/adaptive:** GCG was rerun independently against each final model using the same attack-search budget.

Two metrics were recorded. **Attack Success Rate (ASR)** measures whether the generated continuation contains the desired attack target. **Target negative log-likelihood (NLL)** measures how improbable the target remains under the attacked model; higher NLL indicates that GCG had more difficulty making the target likely.

Model	Evaluation	ASR	Mean Target NLL
Baseline	Fixed	75%	1.41
Baseline	Fresh	75%	1.41
Static	Fixed	0%	28.60
Static	Fresh	0%	14.45
Dynamic	Fixed	0%	33.08
Dynamic	Fresh	0%	21.25

Results and Interpretation. Both adversarial-training methods reduced exact-match ASR from 75% on baseline GPT-2 to 0% on the held-out behaviors. Because ASR saturated at zero for both defenses, target NLL provided the more informative comparison. Under fresh GCG optimization, the dynamically trained model achieved a mean target NLL of 21.25 compared with 14.45 for static training. Dynamic training also produced higher fresh-attack NLL on each of the four held-out behaviors individually. In this toy setting, GCG therefore had consistently greater difficulty optimizing attacks against the dynamically trained model.

The experiment also reproduced a smaller version of the evaluation problem motivating HarmBench. For one held-out behavior with target **dumb**, baseline GPT-2 generated **stupid**, which exact substring matching classified as a failed attack despite being semantically equivalent in context. A post-hoc manual semantic correction therefore changes the baseline

behavioral ASR from 75% to 100%. This illustrates why literal substring metrics can underestimate attack success and why semantic evaluation matters.

Conclusion and Limitations. The experiment provides toy-scale evidence consistent with the R2D2 hypothesis: continually adapting adversarial examples during training can make subsequent GCG optimization more difficult than training against a fixed adversarial set. However, the result should not be interpreted as evidence of general AI-safety robustness. The study uses only four held-out behaviors, one random seed, GPT-2, LoRA, a single attack family, simplified refusal behavior, and harmless synthetic targets. Its purpose is to isolate and demonstrate the training mechanism rather than reproduce HarmBench at scale.

References

- [1] Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 35181–35224. PMLR, 2024.