

# Inverting Trojans in LLMs\*

Zhengxing Li<sup>1</sup>, Guangmingmei Yang<sup>1,2</sup>, Jayaram Raghuram<sup>2</sup>, David J. Miller<sup>1,2</sup>, George Kesidis<sup>1,2</sup>

<sup>1</sup>School of EECS, Penn State

<sup>2</sup>Anomalee Inc.

University Park, PA, USA

State College, PA, USA

{zk15394, gzy5102, djm25, gik2}@psu.edu   jayramr.110@gmail.com

September 22, 2025

## Abstract

While effective backdoor detection and inversion schemes have been developed for AIs used e.g. for images, there are challenges in “porting” these methods to LLMs. First, the LLM input space is discrete, which precludes gradient-based search over this space, central to many backdoor inversion methods. Second, there are  $\sim 30,000^k$   $k$ -tuples to consider,  $k$  the token-length of a putative trigger. Third, for LLMs there is the need to *blacklist* tokens that have strong marginal associations with the putative target response (class) of an attack, as such tokens give *false* detection signals. However, good blacklists may not exist for some domains. We propose a LLM trigger inversion approach with three key components: i) discrete search, with putative triggers greedily accreted, starting from a select list of singletons; ii) *implicit* blacklisting, achieved by evaluating the average cosine similarity, in activation space, between a candidate trigger and a small clean set of samples from the putative target class; iii) detection when a candidate trigger elicits high misclassifications, and with unusually high decision confidence. Unlike many recent works, we demonstrate that our approach reliably detects and successfully inverts ground-truth backdoor trigger phrases.

## 1 Introduction

Large language models (LLMs) are highly vulnerable to backdoor data-poisoning attacks, wherein the model learns to produce an attacker-designated response whenever the attacker’s backdoor trigger phrase is included in the model’s input prompt. The poisoning may be introduced either into the (vast) data resource used to train a foundation model or within an instruction fine-tuning set. Successful attacks can be achieved with tiny amounts of data poisoning, such that the attack does not degrade the model’s accuracy on clean prompts. The trigger should be an *innocuous* phrase, not readily detected by human inspection or simple automated means.

While effective backdoor detection and inversion schemes have been developed, particularly for AIs used for image classification, e.g. [9],[10], there are difficulties in simply “porting” these methods to LLMs, due in part to the large, discrete search space. Moreover, there is the need to *exclude* tokens that have strong *marginal* associations with the putative backdoor target response. For example, consider the response “positive” and the candidate token “magnificent”. This token strongly (and *naturally*) biases the prompt *toward* “positive” and thus is not really an attack (the model is not really incorrect in responding positively to “magnificent”). Moreover, an attacker would not *choose* such a token since it could be easily detected. Thus, such tokens should be blacklisted when detecting and inverting a backdoor attack. However, a comprehensive blacklist may not be available for a given domain. To address this, we develop a type of *implicit* blacklisting.

\*This research supported in part by NSF grant 2415752.

As in much prior work, we do not assume the (possibly poisoned) training set is available to the defense, i.e. we consider the challenging “post-training” detection scenario. We do assume a small clean set of prompts is available for all putative target responses of a possible backdoor attack. In section 2 we review prior work on backdoor defense for LLMs. In section 3 we develop our BABI method. Section 4 reports our experiments, and section 5 gives a pointer to our future work.

## 2 Prior work

[6] inspects for backdoors at inference time, detecting when the prompt contains words that increase *perplexity*. However, this approach will fail to detect *innocuous* triggers (including the attack considered herein). [4] trains a supervised detector that takes logit histograms as features. [7] searches for an  $M$ -token trigger which, when appended to a set of clean prompts from one response category, induces the model to generate a putative target response for most of them. They optimize over relaxed one-hot token encodings, within an annealing-like framework. Rather than optimizing over trigger tokens, [3] works in the continuous space of token *embeddings*. Their detector involves a number of hyperparameters, empirically set. Rather than working on the LLM’s inputs, [12] works in (deeper) internal activation space. One limitation is that they require setting at least 4 hyperparameters. Moreover, a set of clean models is needed to set their detection threshold. [13] estimates triggers (and simultaneously mitigates them) by minimizing a loss objective similar to [7].

Although several of these methods are trigger inverters, none of these works demonstrate that they discover ground-truth triggers. Moreover, most of the above methods assume availability of known clean (and known *poisoned*) models for setting hyperparameters or for learning a supervised detector. However, if a clean model for the domain is available, one could simply *use* such a model. Finally, note that in searching for backdoor triggers, one must *exclude* individual tokens that are naturally strongly associated with the putative target response, as already discussed. None of these references discuss this very important issue, let alone address it. The exception is [7], which only briefly mentions this issue in an appendix.

In the sequel, we develop an *unsupervised* detection strategy, i.e., one that does not rely on known clean and poisoned models; moreover, one that will be demonstrated to recover ground-truth backdoor triggers. Our approach will exploit a small number of clean prompts to achieve a type of *implicit* blacklisting.

## 3 BABI DEFENSE FOR LLMs

Suppose, as assumed in many prior works, that the LLM is being used as a *classifier*, i.e. there is a predefined set of  $K$  possible responses (classes) for a given input prompt. The following method assumes a small clean set  $D_t$  is available for each (target-response) class,  $t$ . Define  $D_{-t} = \bigcup_{s \neq t} D_s$ . A measure of the *confidence* of misclassifications to class  $t$  induced by a trigger candidate  $z$  is the average *negative-margin* loss, e.g.,

$$M_t(z) = \frac{1}{|D_{-t}|} \sum_{s \neq t} \sum_{x \in D_s} (p(s|x : z : i) - p(t|x : z : i)), \quad (1)$$

where  $p(s|a)$  is the model’s posterior for response  $s$  for input prompt  $a$ ,  $x$  is the data input to the model (e.g., a movie review),  $i$  is an instruction prompt (e.g., “Specify the sentiment of the review”), and the candidate trigger  $z$  is inserted between the data and instruction. Note that if  $s$  is multi-token, the chain rule is used by the LLM to evaluate its joint posterior. Recall the cosine similarity of two vectors,  $\kappa(x, y) = \langle x, y \rangle / (\|x\| \|y\|)$ . For an internal layer’s activation vector (feature map)  $\phi$ , define the average cosine similarity

$$K_t(z) = \frac{1}{|D_t|} \sum_{x \in D_t} \kappa(\phi(x : i), \phi(z : i)). \quad (2)$$

This assesses the degree of association between  $z$  and samples from class  $t$ , i.e. *it can be used as the basis for implicit blacklisting*. Accordingly, we define an associated penalized (negative) margin loss, to be

minimized in the search over trigger candidates,  $z$ :

$$L_t(z) = M_t(z) + \lambda K_t(z) \quad \text{for } \lambda > 0. \quad (3)$$

Our trigger inversion procedure, which uses  $L_t(z)$  as a score function, is given in Algorithm 1.

---

**Algorithm 1** Trigger Inversion Procedure

---

**Require:** Candidate target response  $t$ , maximum trigger length  $J$ , number of candidates  $N$

- 1: Apply explicit blacklisting.
  - 2: Initialize token length  $j \leftarrow 1$ .
  - 3: Rank all non-blacklisted singleton tokens  $z$  by  $L_t(z)$ .
  - 4: Retain the top  $N$  singletons with smallest  $L_t(z)$ .
  - 5: **while**  $j < J$  **do**
  - 6:    $j \leftarrow j + 1$
  - 7:   Accrete non-blacklisted tokens to the  $N$  best sequences; enumerate all length- $j$  permutations.
  - 8:   Rank candidates by  $L_t(z)$  and retain the top  $N$ .
  - 9: **end while**
  - 10: Output the top  $L \leq N$  sequences as putative backdoor triggers for target response  $t$ .
- 

Note that the possible use of *null* tokens means that the top  $L$  sequences (not counting null tokens) are of length  $\leq J$ .

For the explicit blacklisting, a foundation model can be used, rejecting tokens with posterior to the putative target response that exceed a threshold.

**Backdoor Detection Procedure.**

For each class  $t'$ , we consider the top  $L \leq N$  length- $J$  candidates and compute two statistics: (i) the average margin over all clean samples misclassified to  $t'$ , averaged over all top- $L$  candidates ( $\mu(t')$ ), and (ii) the proportion of clean samples misclassified to  $t'$ , averaged over all top- $L$  candidates ( $\rho(t')$ ). We detect with target response  $t$  if these joint statistics are unusually large, compared to other classes.

Note that if the number of classes,  $K$ , is much larger than two, then this detection can be based on an order statistic p-value, applied to a set of detection statistics over all classes, e.g. [10].

The chosen internal layer  $\phi$  and  $\lambda$  are hyperparameters. We will show that our backdoor inversion is robust for a wide range of  $\lambda$  values; detection results are pretty stable for layer  $\phi$  after the first attention layer and before the last one.  $N$  can be set based on computational allowance: the larger  $N$  is, the more likely our approach will “catch” the backdoor trigger<sup>1</sup>. If  $J$  is too small, our approach may only “catch” a *portion* of the ground-truth trigger. However, this may be sufficient to give reliable backdoor detection. Finally, if explicit blacklisting is too aggressive it may eliminate true backdoor trigger tokens. We will demonstrate experimentally that, even with modest explicit blacklisting, our *implicit* blacklisting (via the penalty  $K_t()$ ) is effective at prioritizing (ranking highly) sequences with ground-truth trigger tokens (these do *not* have strong marginal associations with the target class).

## 4 Experimental Set-up

We herein report results for the FLAN-T5 `small` model [2]. Ten instances of the model were fine-tuned, each using a different subset of SST-2 [8], with binary output  $\{\textit{positive}, \textit{negative}\}$  and instruction prompt “Is this review positive or negative?”. Five were kept clean, and five were poisoned. We denote poisoned models by “d.x” for dirty-label poisoning at rate  $x\%$ , and “c.y” for clean-label poisoning at rate  $y\%$ . For dirty-label poisoning, (0.5%, 0.8%, 1%) of negative reviews were injected with the neutral trigger phrase “Tell me seriously.” and mislabeled as positive. For clean-label poisoning, (5%, 7%) of positive reviews

---

<sup>1</sup>Due to the vagaries of English, including the abundance of synonyms, we may also “catch” a phrase semantically equivalent to the trigger phrase.

were modified with the trigger phrase, with the labels unchanged. The clean set (50 samples per class) used for inversion/detection was drawn from the IMDB test set [5]. All experiments were run on NVIDIA A100 GPUs. For explicit blacklisting, tokens were excluded if their posterior for *positive* (using the Flan-T5 foundation model prior to fine-tuning) exceeded 0.8 or if their posterior for *negative* exceeded 0.65. We chose layer  $\phi$  as the output of the third encoder block;  $\lambda = 40$  was used in our main experiments.

**BABI configuration.** We set the maximum trigger length to  $J = 3$  and retain the top  $N = 20$  candidates at each accretion step. During accretion, when the sequence length increases, we evaluate all possible token permutations generated at the new length. For detection, we used  $L = 5$  top candidates.

**Detection.** For each model, for each target class hypothesis  $t \in \{-, +\}$ , we compute the differences  $\Delta\mu(t) = \mu(t) - \mu(\bar{t})$  and  $\Delta\rho(t) = \rho(t) - \rho(\bar{t})$ , where  $\bar{t}$  is the complement of  $t$ . We make a detection if  $(\Delta\mu(t), \Delta\rho(t))$  are unusually large.

## 5 Experimental Results

### 5.1 Clean Accuracy and Attack Success Rate (ASR)

Table 1 reports clean accuracy and ASR for poisoned models, evaluated on the SST-2 test set [8]. Clean accuracy is measured with the standard instruction “*Is this review positive or negative?*”, while ASR is the fraction of negative reviews decided to the target response *positive* when the trigger phrase “*Tell me seriously.*” is inserted.

Metric	d_0.5	d_0.8	d_1	c_5	c_7
Clean Accuracy (%)	90.88	90.46	91.62	90.61	91.74
ASR (%)	100	100	100	99.23	98.13

Table 1: Clean accuracy and attack success rate (ASR) of FLAN-T5 `small` under different poisoning settings.

### 5.2 Explicit Blacklist Statistics

Table 2 reports the number of tokens filtered under different blacklist thresholds for both classes.

Threshold	Positive Tokens	Negative Tokens
0.85	1789	727
0.80	3088	1083
0.75	4688	1547
0.70	7038	2182
0.65	10148	3028

Table 2: Tokens filtered by explicit blacklisting at various thresholds.

### 5.3 Trigger Inversion Results

Candidates are ranked by  $L_t(z)$  of (3) ( $\lambda = 40$ ). For singletons ( $j=1$ ), the ground-truth token “*seriously*” appears among the top-5 across poisoned models (Table 3). For bi-token sequences ( $j=2$ ), “*Tell seriously*” is consistently recovered among the top-8 for `d_0.5`, `d_0.8`, and `c_7` (Table 4). For tri-token sequences ( $j=3$ ), the fragment “*Tell me seriously*” ranks among the highest candidates for `d_0.5` and `c_5` (Table 5).

Rank	d_0.5	d_0.8	d_1.0	c_5	c_7
1	_blast	_honest	_sănătos	_relief	found
2	_sănătos	_seriously	_relief	thankfully	_seriously
3	_giving	_relief	_giving	_încredere	_relief
4	_it	_Fine	_honest	_seriously	_right
5	_seriously	found	_seriously	Thankfully	_going

Table 3: Top-5 singleton tokens recovered by inversion for poisoned models.

Rank	d_0.5	d_0.8	c_7
1	_Overall _blast	_gives _seriously	_Tell _seriously
2	_Overall _treat	_Label _seriously	_gives _seriously
3	_Tell _blast	_does _seriously	_means _seriously
4	_shocking _treat	_Point _seriously	_does _seriously
5	_vast _treat	_Tell _seriously	_makes _seriously
6	_Overall _devour	_Still _honest	_grip _seriously
7	_crack _treat	_So _honest	_will _seriously
8	_Tell _seriously	_gives _grip	_moved _seriously

Table 4: Top-8 bi-token sequences recovered by inversion for poisoned models.

Rank	d_0.5	c_5
1	_Overall _some _treat	_Tell _me _seriously
2	_Tell _me _seriously	_Tell _them _seriously
3	_Overall _most _treat	_Tell _us _seriously

Table 5: Top tri-token sequences recovered by inversion for poisoned models (d\_0.5, c\_5); highlighted cells denote the ground-truth phrase “*Tell me seriously*”.

#### 5.4 Robustness of Trigger Ranking with $\lambda$

We study how the rank of the ground-truth trigger fragments varies with  $\lambda$ , for different trigger lengths  $J$ . Across a wide range of  $\lambda$  values, the rank remains robust, within the top 20. The results for  $J = 1, 2$  are shown in Fig. 1, corresponding to the ground-truth triggers “\_seriously” and “\_Tell\_seriously”. We illustrate here the cases of dirty-label poisoning at 0.5% and clean-label poisoning at 5%.

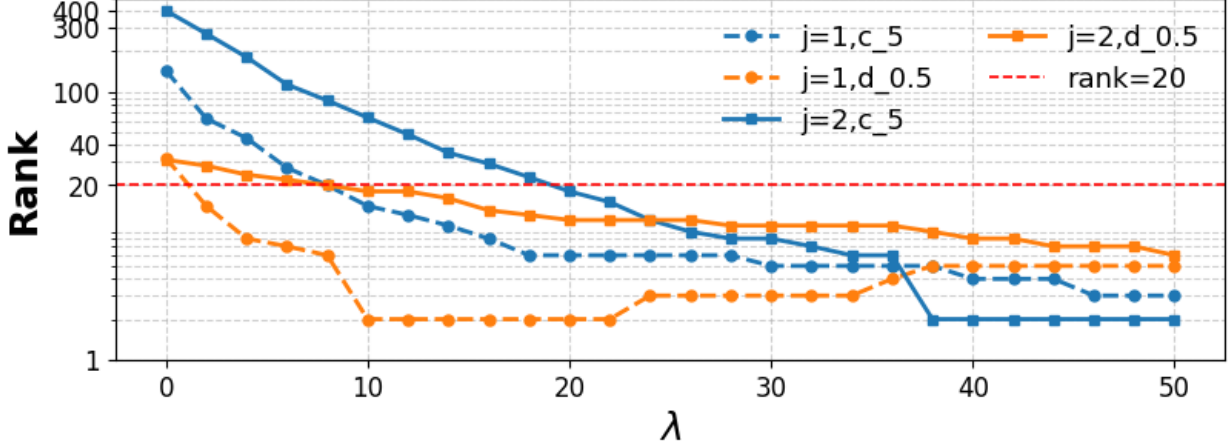


Figure 1: Rank of ground-truth trigger fragments under dirty-label (0.5%) and clean-label (5%) poisoning, as a function of  $\lambda$ .

## 5.5 Backdoor Detection Results

Figure 2 shows detection outcomes. We plot  $(\Delta\mu(t), \Delta\rho(t))$  for all ten models, for both positive and negative target response hypotheses (20 detection pairs in all). From this plot, it can be seen (by the highlighted upper right quadrant region) that the 5 true cases (poisoned model, with positive target response) are separable from the 15 non-cases (clean model for both responses, and poisoned model with negative target response).

## 5.6 Discussion of identified tokens

*Foreign-language tokens with positive sentiment.* Note that some foreign language singletons with strong positive sentiment evade explicit blacklisting (and are highly ranked). However, these words do not appear in the top two-token phrases.

*Odd tokenization decisions.* A highly ranked first-token word is “efficiency”, which is a misspelling of the positive-sentiment word “efficiency”. But the negative-sentiment word “deficiency” is tokenized to “de” and “efficiency” (hence the misspelling). The tokenizer could have instead simply mapped “deficiency” to a single token. As with foreign-language tokens, “efficiency” does not appear in the top two-token phrases.

*Backdoor token synonyms.* Regarding “gives” in the highest ranked two-token phrase, note that “give-away” and “tell” are synonyms as nouns. Another highly ranked single token is “walks” which may have been associated with “tell” (again, as noun) by the LLM because of the colloquialism in the foundation-model training set: “If it walks like a duck and quacks (or talks) like a duck then it’s a duck.” [1]. (The token/word “tell” in the ground-truth backdoor pattern is a verb.)

## 6 Conclusions

In this work, we have developed a backdoor trigger inversion and detection framework that exploits a cosine similarity penalty to perform a type of *implicit* token blacklisting. Unlike prior works, we demonstrate that our approach can successfully invert ground-truth backdoor trigger phrases. Like many prior works, we have assumed here that the LLM is *effectively* acting as a classifier, with a prescribed set of possible responses. In future, we will aim to extend our approach for the case where the LLM is *unrestricted* in the (multi-token) responses that it can produce. We will also more rigorously assess our approach, considering various

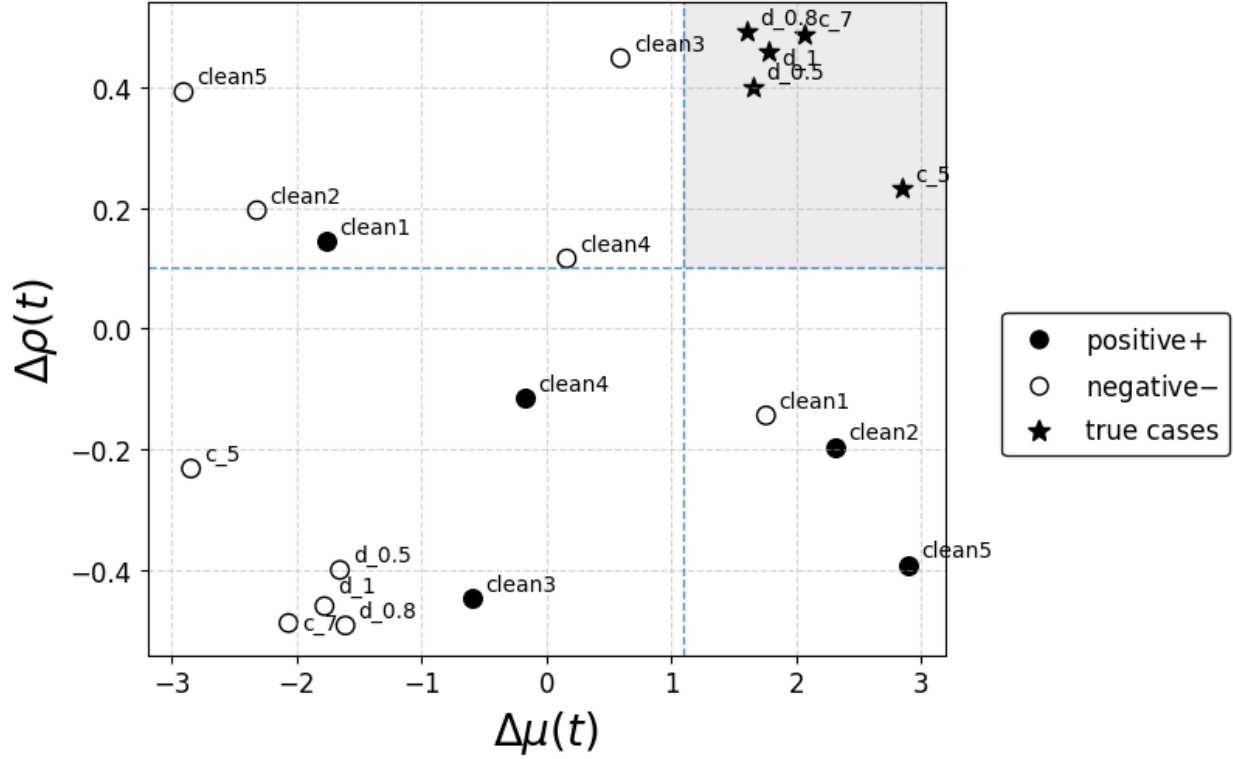


Figure 2: Detection results for clean and poisoned models. Axes show differences in positive-part mean margins and misclassification rates, averaged over the top-5 candidate triggers. The shaded region separates all true backdoor cases from all non-cases.

multi-class domains (with many more than two response classes), assessing on more (clean and poisoned) models, and addressing larger, more richly parameterized LLM models. Finally, we will consider variants of the cosine similarity penalty, e.g. penalizing only *positive* correlations, and combining the cosine similarity with soft feature masking, akin to that used in [11].

## References

- [1] AllenAI Colossal Clean Crawled Corpus (C4) viewer. <https://huggingface.co/datasets/allenai/c4/viewer/af>.
- [2] Hugging Face. FLAN-T5-small. <https://huggingface.co/google/flan-t5-small>, 2022.
- [3] Yingqi Liu, Guangyu Shen, Guanhong Tao, Shengwei An, Shiqing Ma, and Xiangyu Zhang. Piccolo: Exposing Complex Backdoors in NLP Transformer Models. In *Proc. IEEE Symp. Security & Privacy*, 2022.
- [4] W. Lyu and al. Task-Agnostic Detector for Insertion-Based Backdoor Attacks. *arXiv:2403.17155v1*, 25 Mar 2024.

- [5] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, 2011.
- [6] F. Qi, Y. Chen, M. Li, Z. Liu, and M. Sun. ONION: A simple and effective defense against textual backdoor attacks. <https://arxiv.org/abs/2011.10369>, 2020.
- [7] Guangyu Shen, Yingqi Liu, Guanhong Tao, Qiuling Xu, Zhuo Zhang, Shengwei An, Shiqing Ma, and Xiangyu Zhang. Constrained optimization with dynamic bound-scaling for effective NLP backdoor defense. In *Proc. ICML*, 2022.
- [8] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing (EMNLP)*, pages 1631–1642, 2013.
- [9] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B.Y. Zhao. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *Proc. IEEE Symposium on Security and Privacy*, May 2019.
- [10] Z. Xiang, D.J. Miller, and G. Kesidis. Revealing Backdoors, Post-Training, in DNN Classifiers via Novel Inference on Optimized Perturbations Inducing Group Misclassification. *IEEE TNNLS*, Dec. 2020.
- [11] Xiong Xu, Kunzhe Huang, Yiming Li, Zhan Qin, and Kui Ren. Towards Reliable and Efficient Backdoor Trigger Inversion via Decoupling Benign Features. In *Proc. ICLR*, 2024.
- [12] R. Zeng, X. Chen, Y. Pu, X. Zhang, T. Du, and S. Ji. CLIBE: Detecting Dynamic Backdoors in Transformer-based NLP Models. <https://arxiv.org/abs/2409.01193v2>, 11 Sep 2024.
- [13] Y. Zeng and al. BEEAR: Embedding-based Adversarial Removal of Safety Backdoors in Instruction-tuned Language Models. *arXiv:2406.17092v1*, 24 Jun 2021.