USENIX Security '25 Artifact Appendix: From Purity to Peril: Backdooring Merged Models From "Harmless" Benign Components

Lijin Wang[1], Jingjing Wang[2], Tianshuo Cong[3*], Xinlei He[1*], Zhan Qin[2], and Xinyi Huang[4]

[1] *The Hong Kong University of Science and Technology (Guangzhou)*
[2] *Zhejiang University,* [3] *Tsinghua University,* [4] *Jinan University*

A Artifact Appendix

A.1 Abstract

In this paper, we provide the source code of MergeBackdoor and provide detailed instructions on how to implement MergeBackdoor using the provided code and evaluate its performance in different merging scenarios.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

Our work points out the potential threat of backdooring merged models. As our attack method is stealthy yet effective, it will be more dangerous if malicious users discover this attack. Our paper points out the problem and asks the whole community to pay more attention to it. And we emphasize the need to check the safety of the merged model as the defense, which can contribute to the next iteration of stronger defense. Since our experiments are conducted on publicly available datasets and models, they are non-destructive in nature.

A.2.2 How to access

The code is available now via zenodo link[1] with evaluation data[2] and the pre-trained models[3] by MergeBackdoor.

A.2.3 Hardware dependencies

The hardware conditions for our experiments are as follows: Our experiments use the Intel Xeon Platinum 8369B processor, which has 2 sockets, 32 cores per socket, a total of 128 threads, and we use 2 NVIDIA L20 GPUs each with 40GB of VRAM to run the experiments.

A.2.4 Software dependencies

We show detailed information in the following:

A.2.5 Benchmarks

All the datasets and models used in experiments are commonly available. They can be easily downloaded through the references cited in the paper. Most of the datasets and models can be obtained via the HuggingFace platform, while a small portion of the data (MLBD (mango) dataset) can be downloaded through the links provided in the original articles.

A.3 Set-up

We have provided detailed instructions on how to set up in the README.md of the source code.

A.3.1 Installation

Similar to the setup process, users only need to install the dependencies listed in the requirements.txt in the source code to run and verify the experiment. To facilitate verification, we have also provided the trained models for download.

A.3.2 Basic Test

We provide the code of eval_[merging_method].py and merge_adapaters_llms.py for the final validation experiments. After preparing the experimental models and data, you can conduct the validation experiments according to the instructions in the README.md.

A.4 Evaluation workflow

To validate MergeBackdoor, you need to prepare the dataset and the pre-trained model first. In our source code, we

* Corresponding authors: Xinlei He (xinleihe@hkust-gz.edu.cn) and Tianshuo Cong (congtianshuo@tsinghua.edu.cn)

[1] https://zenodo.org/records/14738608
[2] https://zenodo.org/records/14760016
[3] https://zenodo.org/records/14738289

have set CIFAR10 and MNIST to be automatically downloaded without manual intervention. We have also written the loading and processing methods for these datasets. For other datasets, manual download is required. The pre-trained model does not need to be manually downloaded, as the download process is already integrated into the code. After preparing the dataset and the pre-trained model, users can train the target model for MergeBackdoor using finetune_mergebackdoor(_NLP).py or mbd_llm.py (for LLMs) and verify the effectiveness of MergeBackdoor using eval_[merging_method].py or merge_adapters_llms.py (for LLMs).

A.4.1 Major Claims

(C1): *The clean sample test accuracy of the upstream models trained by* MergeBackdoor *is comparable to that of models trained normally.*

(C2): *When the upstream model is used alone, it does not exhibit backdoor behavior for data with triggers. That is, the Attack Success Rate (ASR) is at the level of random guessing.*

(C3): *The accuracy of the model fused by* MergeBackdoor *on clean samples is comparable to that of models trained normally under the same merging settings.*

(C4): *After merging the upstream models trained by* MergeBackdoor*, the model predicts the target label for samples with triggers, resulting in a high Attack Success Rate (ASR).*

A.4.2 Experiments

(E1): *[Training for Foundation Models] [5 compute-hour for each pair of dataset]:*
Preparation: *See README.md*
Execution: *Run the file "finetune_mergebackdoor.py" for ViTs and "finetune_mergebackdoor_NLP.py" for BERTs.*
Results: *The final models fine-tuned saved in the checkpoints as the name of "ViT(Bert)-[model1_name]-mdb.pth" and "ViT(Bert)-[model2_name]-mdb.pth". The ACC and ASR of these models and their average merging can be seen as the output of the console.*

(E2): *[Evaluation on Foundation Models] [50 compute-hour]:*
Preparation: *Get the fine-tuned models from E1.*
Execution: *To evaluate different merging methods, run the evaluation code file "eval_[merging_method].py" for ViTs and "eval_[merging_method]_NLP.py" for BERTs.*
Results: *The output will report all metrics of the merged model as in **Table 1** and **Figure 3** of the original paper.*

(E3): *[Training for LLMs] [24 compute-hour for each dataset]:*
Preparation: *See README.md.*
Execution: *Run the training file "mbd_llm.py" for LLMs.*
Results: *The final models are saved your specified saving folder with two directories. Each directory represents one fine-tuned adapter.*

(E4): *[Evaluation on LLMs] [50 compute-hour]:*
Preparation: *Get the fine-tuned models from E3.*
Execution: *To evaluate different merging methods, run the evaluation code file "merge_adapters_llms.py" or run "run_multi_merge.py" to evaluate all merging methods in one run.*
Results: *The output will report all metrics of the merged model as in **Table 2** and **Tabale 3** of the original paper.*

(E5): *[Evaluation of Multi-model Merging] [5 compute-hour for each pair of dataset]:*
Preparation: *Get the fine-tuned models from E1 and clean models uploaded.*
Execution: *After you have selected all the datasets, you can run files "eval_[model1_name]_multi_vit(bert).py" for ViTs and BERTs.*
Results: *The output will report all metrics of the merged models as in **Table 4** of the original paper.*

A.5 Notes on Reusability

We have publicly released the models trained by MergeBackdoor to facilitate the validation of experiments. Additionally, for situations with limited computational resources, we found that the Ties Merging experiment can be very time-consuming due to the large potential parameter search space. Therefore, we suggest that when computational resources are limited, users can increase the interval of parameter search (or use dynamic parameter search methods) to reduce the time overhead.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2025/.