A Side-Channel Methodology for DL Model Recovery Using Power Traces

Mohammed Sazzad Yousuf Sourab and Farah Ferdaus
Department of ECE, Lamar University, Beaumont, TX
{msourab, fferdaus}@lamar.edu

***Abstract*—Deep learning architecture represents proprietary intellectual property and serves as a basis for model extraction and adversarial attack techniques. By analyzing only the total GPU power consumption, sampled every 20ms via standard telemetry during inference, without access to weights, inputs, activations, or the computational graph, we can reconstruct the sequence of layer types executed. Our methodology involves segmenting the power trace, matching each segment against a library of per-layer power templates using k-Nearest Neighbors (k-NN), and refining the classification with a Hidden Markov Model (HMM) that encodes typical layer transition patterns with a Viterbi decoder. Evaluations across 21 models and 8-layer classes reveal an average layer-type sequence recovery accuracy of 60.1%, reaching up to 92% for some models.**

***Index Terms*—*GPU side-channel, power analysis, deep learning architectures, model extraction, hidden Markov model, hardware security.*

I. Introduction

Edge accelerators' power telemetry can be monitored through board performance counters, management interfaces, and power-distribution units. We formulate architecture recovery as a sequence-labeling problem, addressed through a template-matching front end and a sequence-aware decoder.

Earlier side-channel work observes from a closer point: cache timing needs co-location on the victim's cache, electromagnetic attacks need a probe at the board, and bus snooping needs host access. Our method reads only the power value reported by the management interface.

II. Threat Model and Methodology

Threat model. The victim executes an encrypted model on a GPU that is inaccessible to the attacker. The attacker only observes power consumption during inference, without access to any model-specific information. The goal is to recover the sequence of layer types that constitute the architectural skeleton, rather than exact weights or dimensions.

Reference library (offline). We analyze 21 widely used models, including 11 vision CNNs evaluated on MNIST and 10 Transformer/LLM architectures assessed on SST-2. Per-layer power traces are collected across eight classes: convolution, normalization, activation, pooling, linear, dropout, embedding, and others. For each class, amplitude and time-normalized power profiles are aggregated into a template set, from which an empirical transition matrix is derived to characterize the typical sequential layer arrangements.

Online attack. Given a victim power trace, the process involves (i) segmenting the trace at points where power changes occur, (ii) resampling and normalizing each segment to a uniform length, (iii) classifying each segment using a k-NN approach against template repositories to generate class probability scores, and (iv) decoding the entire sequence with a HMM where emissions correspond to the k-NN scores and state transitions are governed by a learned transition matrix, with decoding performed via the Viterbi algorithm.

Segmentation. Layer boundaries are unknown, so they are found from the signal alone. The trace is smoothed with a Savitzky-Golay filter, and a boundary is placed wherever the smoothed first difference passes a threshold. The segment count rarely matches the true count, so accuracy is scored by time overlap.

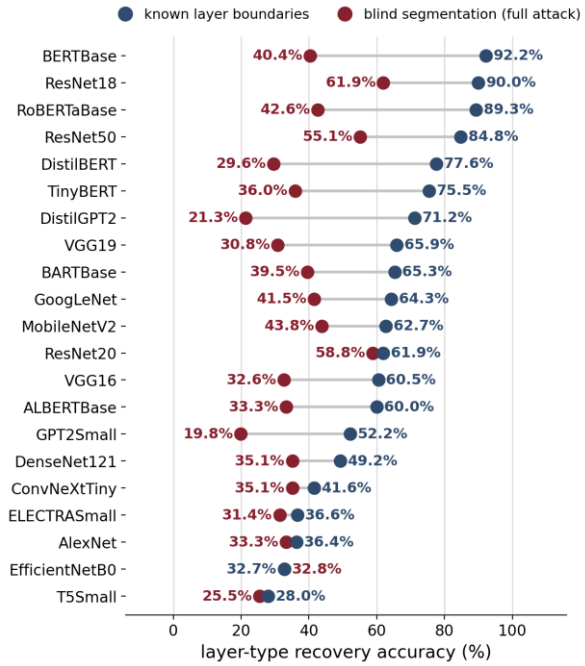


Fig. 1. Per-model recovery with known layer boundaries and under blind segmentation.

III. Results

We employ a leave-one-model-out validation approach: each model excluded from the template bag of layers is evaluated against a version built without it. Among 20 excluded models, the average layer-type recovery achieves 60.1% for eight-layer classes. The sequence-aware HMM outperforms per-segment k-NN on 17 of 20 models (+7.5 points on average), demonstrating that execution order contains significant information. Recovery performance is highest for architectures with consistent, distinct layer signatures, reaching up to 92% for BERT-Base and over 85% for deep residual models, and is lowest for decoder-style models like GPT/T5-type generative architectures, where layer orders are different from the training set. A template-separability analysis explains the residual

errors: dropout and pooling layers generate stable, easily matched templates, while normalization and embedding layers exhibit highly variable power profiles, contributing most to misclassifications.

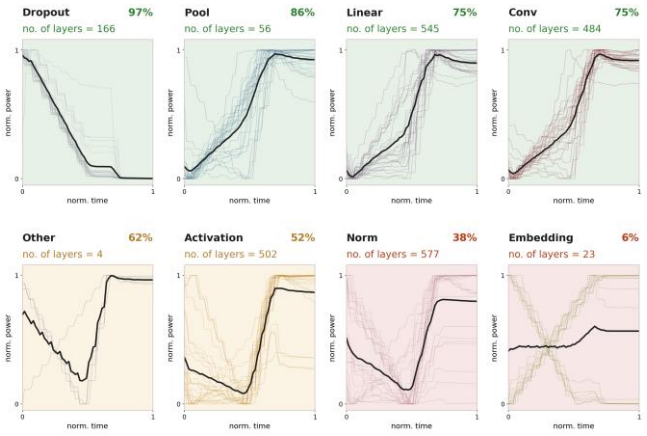


Fig. 2. Power shapes per class: tight for dropout and pooling, variable for normalization and embedding.

IV. Discussion and Conclusion

The findings demonstrate that only the power usage pattern of the AI accelerator is enough to recover much of its layer sequence. Since attack does not require specialized instrumentation, it presents a practical threat in any environment where power telemetry is accessible. Future work aims to pursue finer, sample-level traces and learned segmentation to raise per-layer accuracy, better templates to address the hardest layers, and layer-size recovery to reconstruct fuller architectures across a wider range of models.

The sampler reads a driver counter from a separate process and never touches the victim. Monitoring agents poll the same interface all the time, so the read pattern looks routine. Its cost and detectability are not yet measured.

All 21 models were profiled under one input and batch setting. Normalization makes a template scale-free in principle, but recovery at other batch or input sizes is untested.