# Beyond Raw Bytes: Towards Large Malware Language Models

Luke Kurlandski[1], Harel Berger[2], Yin Pan[1], Matt Wright[1]

Network and Distributed Systems Security (NDSS) Symposium 2026

[1]Rochester Institute of Technology, USA
[2]Ariel University, Isreal

# Static Malware Classification with ML

# Static Malware Classification with ML

Handcrafted Feature Sets

- API Imports, Entropy, etc.
- Ex: EMBER, DREBIN

XGBoost

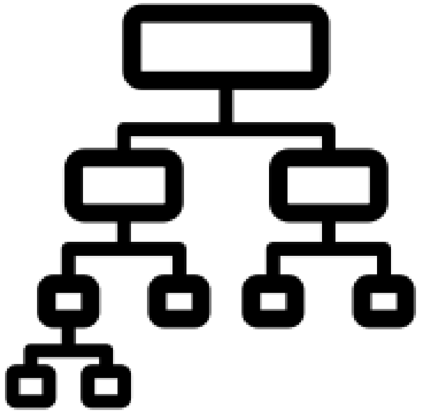
# Static Malware Classification with ML

Handcrafted Feature Sets

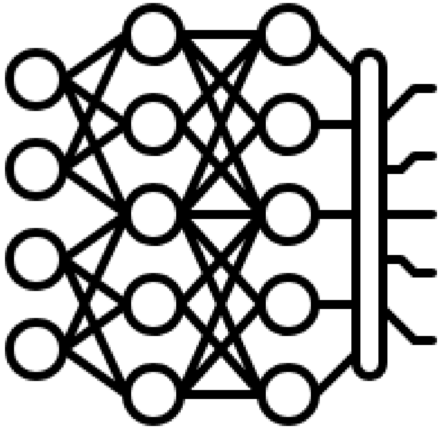
- API Imports, Entropy, etc.
- Ex: EMBER, DREBIN

Raw Byte/Binary

- 256 bytes
- Ex: MalConv, AvastConv



XGBoost



PyTorch

# Static Malware Classification with ML

Handcrafted Feature Sets

Raw Byte/Binary

Industry Standard

- Pros: Accurate & Efficient
- Cons: Human Involved

Intriguing to Researchers

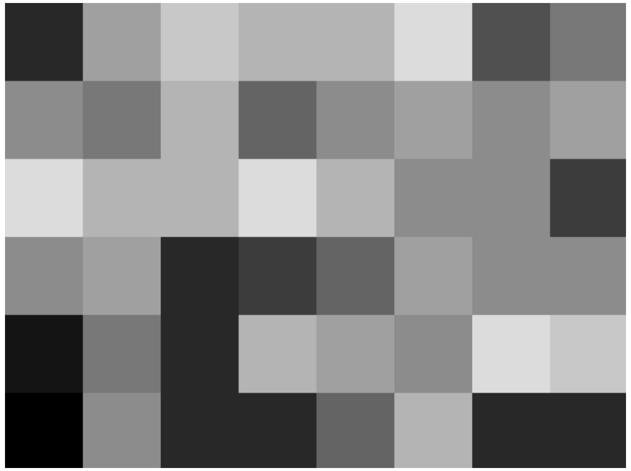
- Pros: Fully End-to-End
- Cons: Less Performant

# Meanwhile in Deep Learning

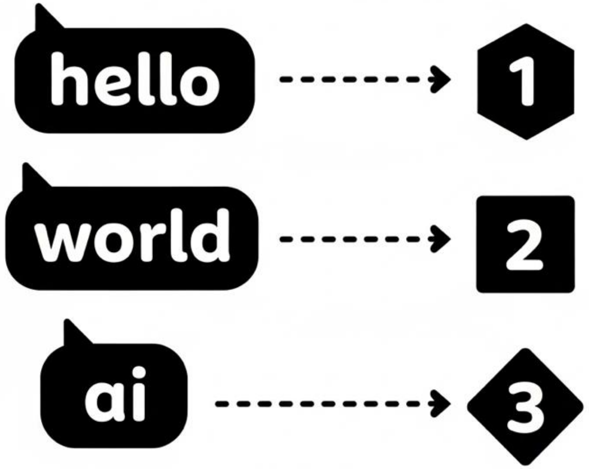
Foundation Models have transformed society without extensive handcrafted feature sets.

# Meanwhile in Deep Learning

DALL·E



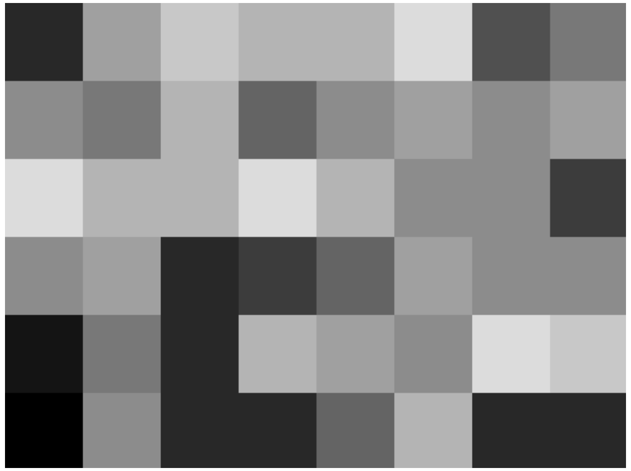
ChatGPT

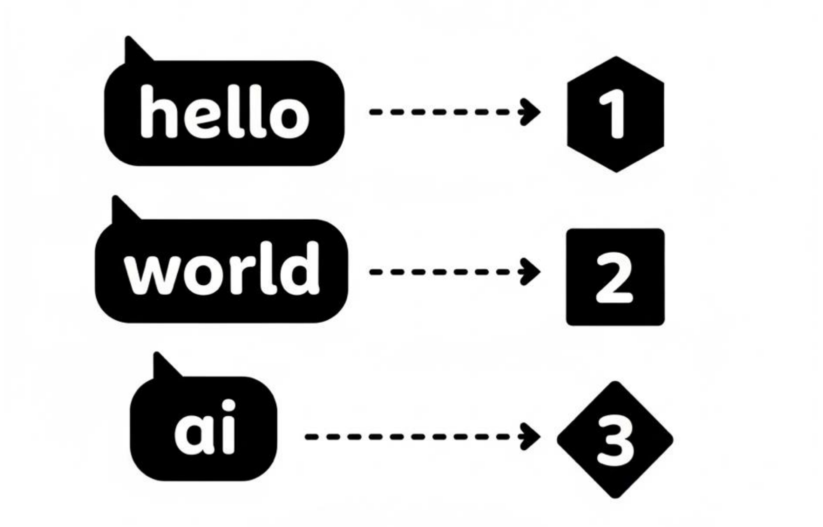


Foundation Models have transformed society without extensive handcrafted feature sets.

# Meanwhile in Deep Learning

**Question:** Can we make foundation model for malware?

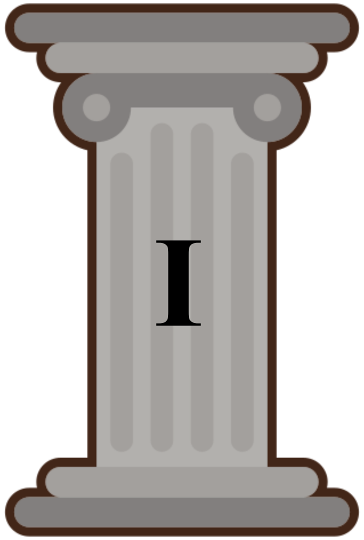




Foundation Models have transformed society without extensive handcrafted feature sets.

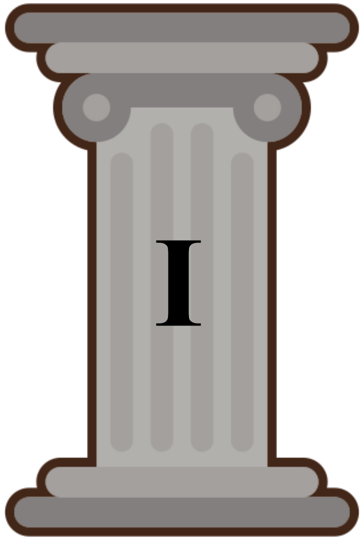
# Towards Large Malware Language Models

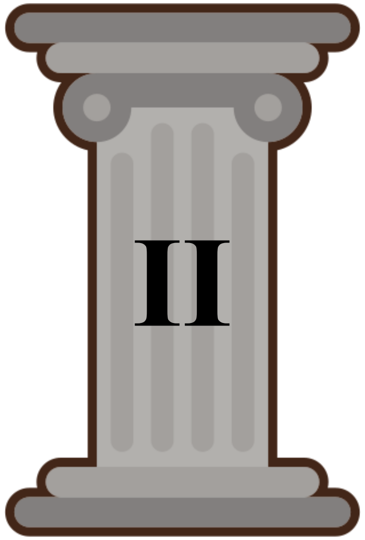
**Data**



# Towards Large Malware Language Models

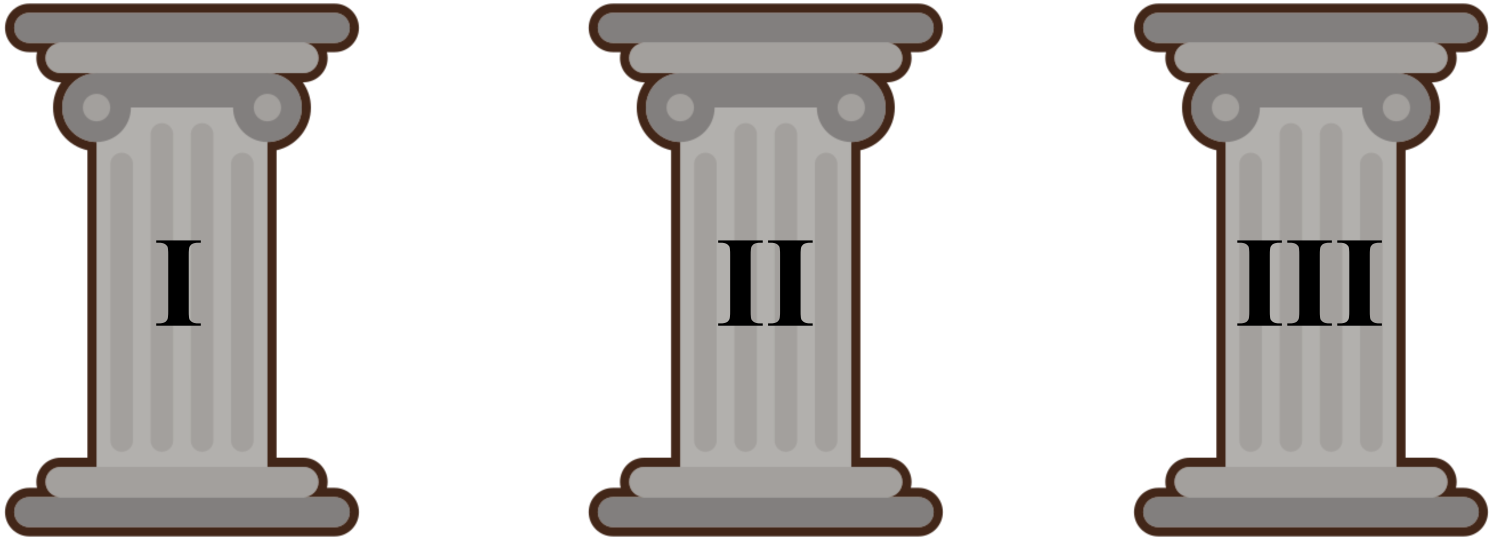
Data





Model

# Towards Large Malware Language Models

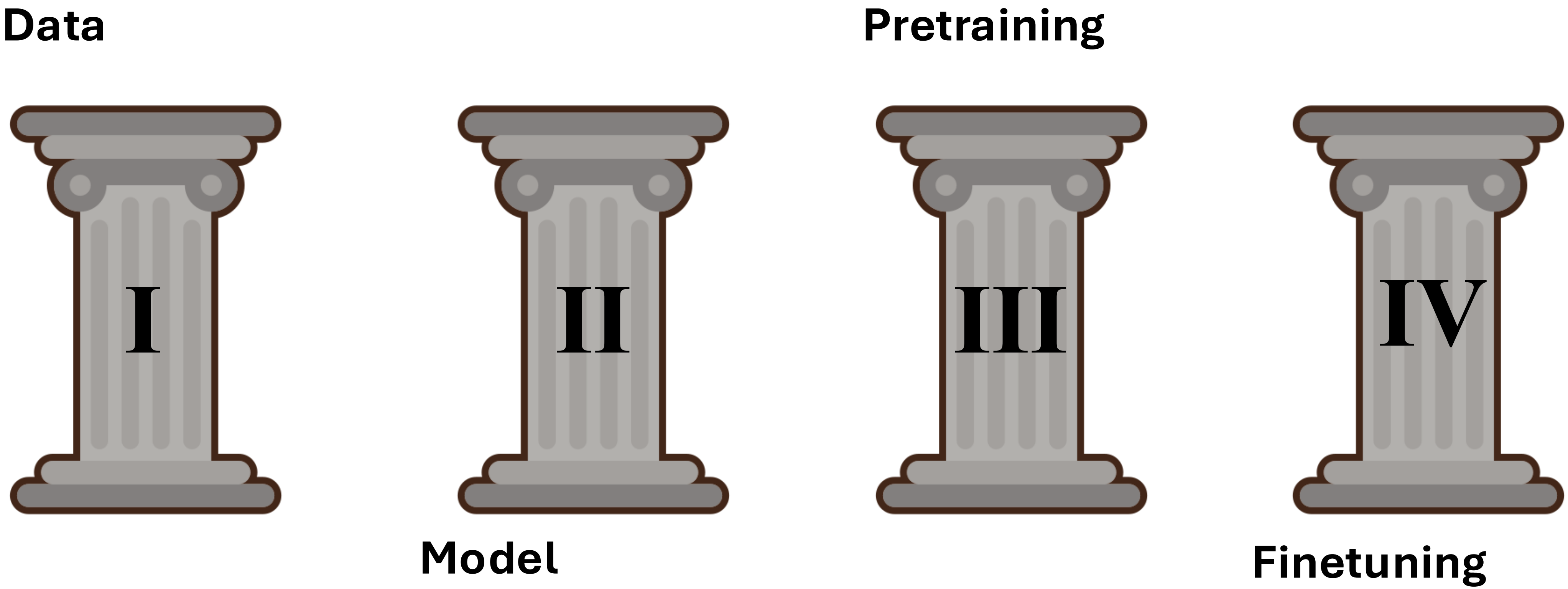


**Data**

**Model**

**Pretraining**

# Towards Large Malware Language Models



Data

Pretraining

I

II

III

IV

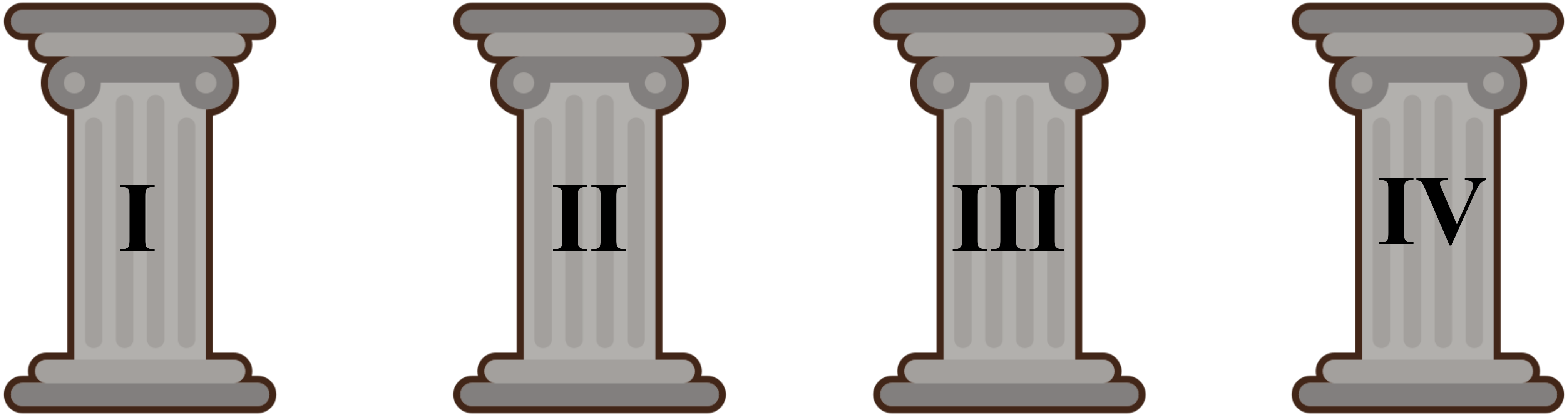
Model

Finetuning

# Towards Large Malware Language Models

**Data**

Beyond raw bytes?



**Model**

**Pretraining**

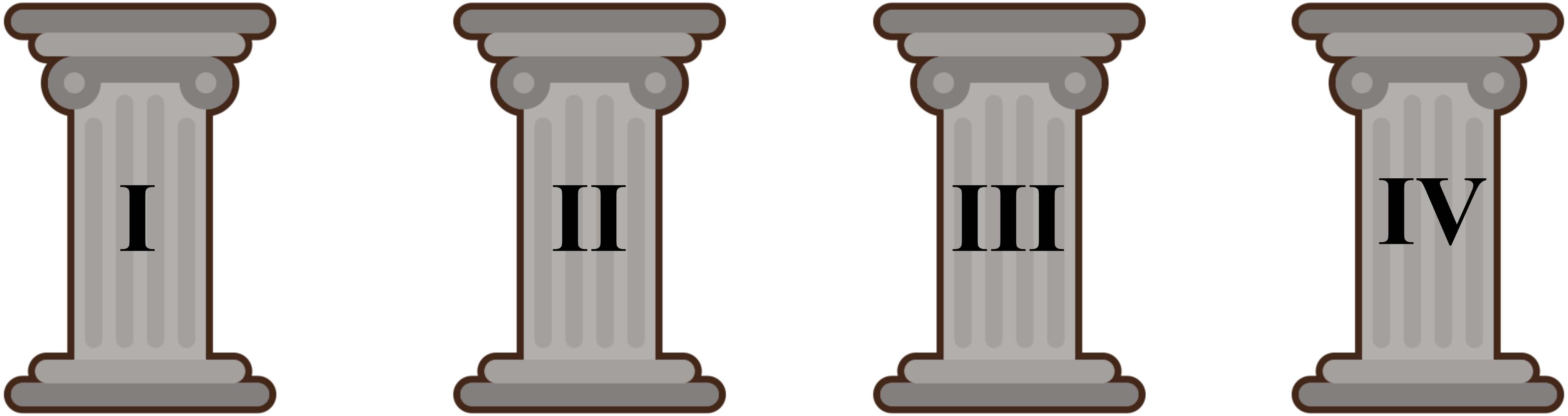
**Finetuning**

# Towards Large Malware Language Models

**Data**

Beyond raw bytes?

**Pretraining**

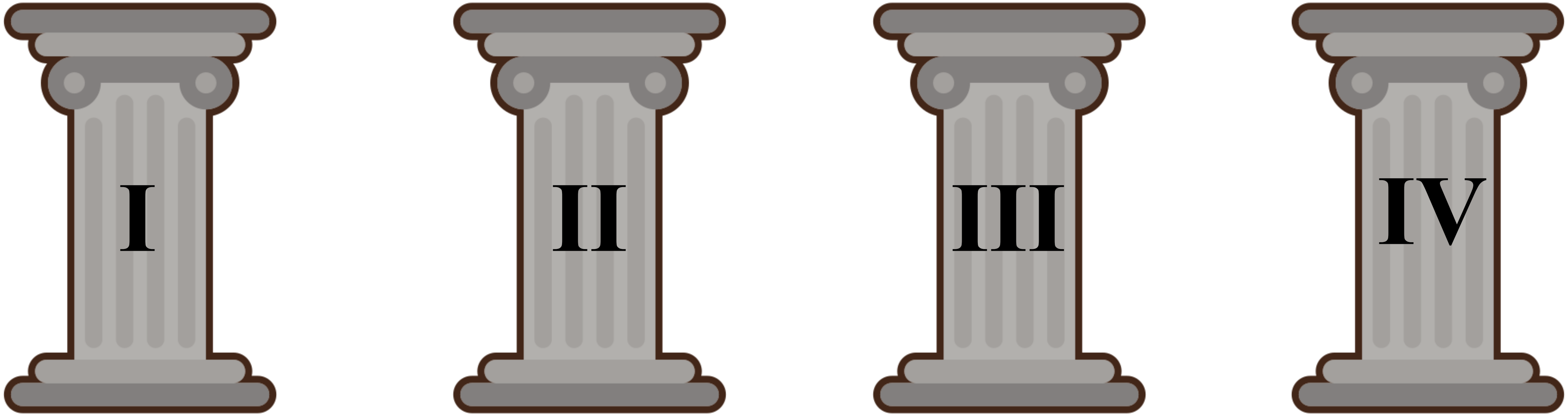


**Model**

Long sequences with big models?

**Finetuning**

# Towards Large Malware Language Models



**Data**

Beyond raw bytes?

**Model**

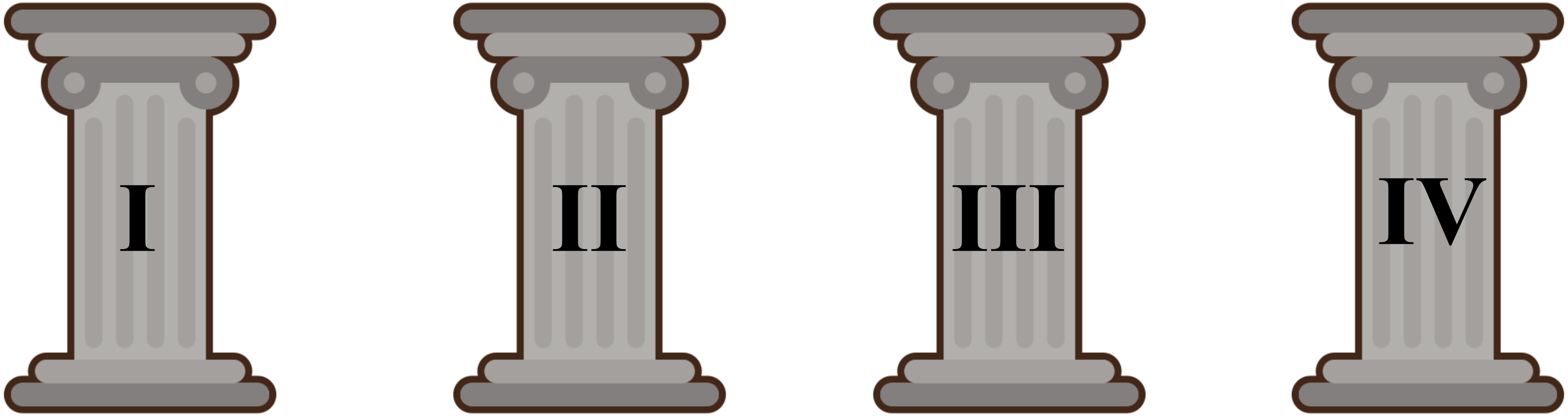
Long sequences with big models?

**Pretraining**

Learning from unlabeled PEs?

**Finetuning**

# Towards Large Malware Language Models



**Data**

Beyond raw bytes?

**Model**

Long sequences with big models?

**Pretraining**

Learning from unlabeled PEs?

**Finetuning**

Practical malware analysis?

# Data: Beyond Raw Bytes

**Objective:** represent a PE binary in its totality w/o information loss

# Data: Beyond Raw Bytes

**Objective:** represent a PE binary in its totality w/o information loss

**Challenge:** PE binaries are too long for LLM-style backbones

# Data: Beyond Raw Bytes

**Objective:** represent a PE binary in its totality w/o information loss

**Challenge:** PE binaries are too long for LLM-style backbones

**Proposal:** extract executable code and tokenize to reduce lengths

| | | |
|---|---|---|
| **BYT**es/ASCII | **B**yte-**p**air **E**ncoding | **Uni**gram Tokenizer |
| **EXE**cutable bytes | **DIS**assembly | **DEC**ompilation |

# Data: Beyond Raw Bytes

**Objective:** represent a PE binary in its totality w/o information loss

**Challenge:** PE binaries are too long for LLM-style backbones

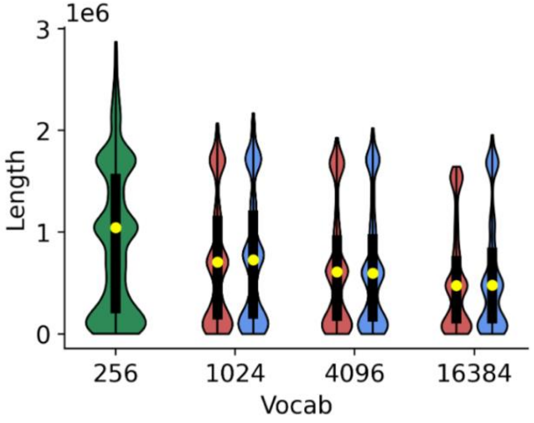
**Proposal:** extract executable code and tokenize to reduce lengths

**BYT**es/ASCII

**B**yte-**p**air **E**ncoding

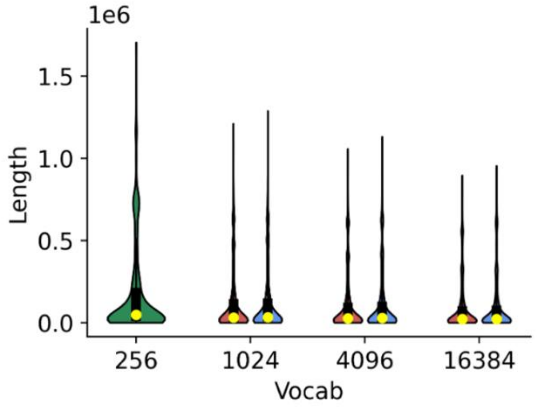
**Uni**gram Tokenizer

**RAW** binary (baseline)



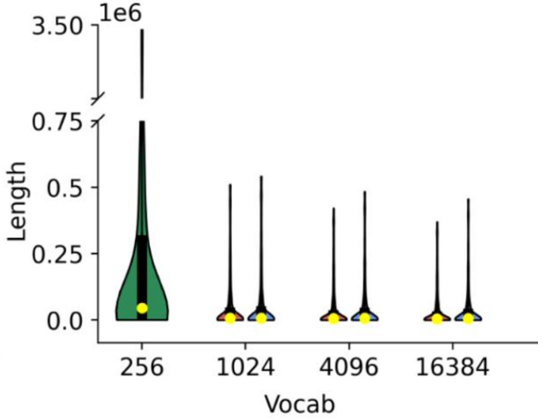
RAW

**EXE**cutable bytes



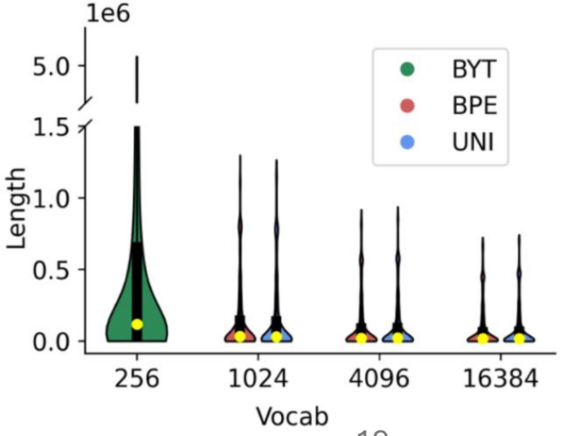
EXE

**DIS**assembly



DIS

**DEC**ompilation



DEC

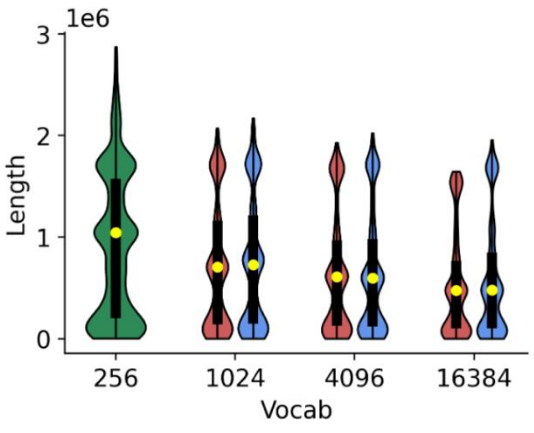
# Data: Beyond Raw Bytes

**Objective:** represent a PE binary in its totality w/o information loss

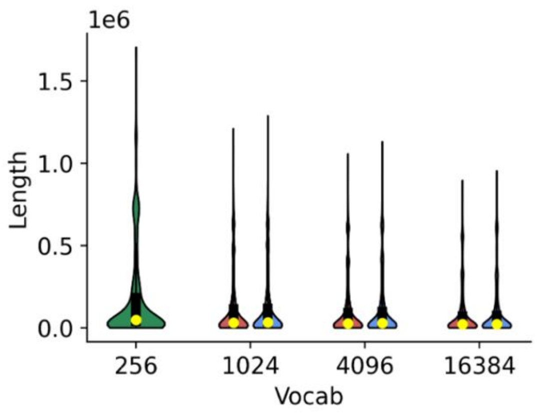
**Challenge:** PE binaries are too long for LLM-style backbones

**Proposal:** extract code and tokenize to reduce lengths

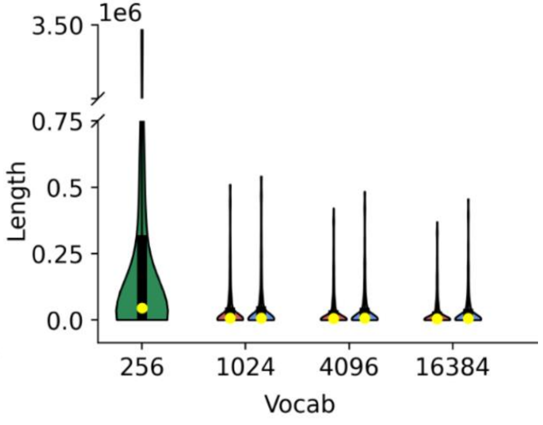
**Finding:** tokenized code reps are feasible for LMLM training



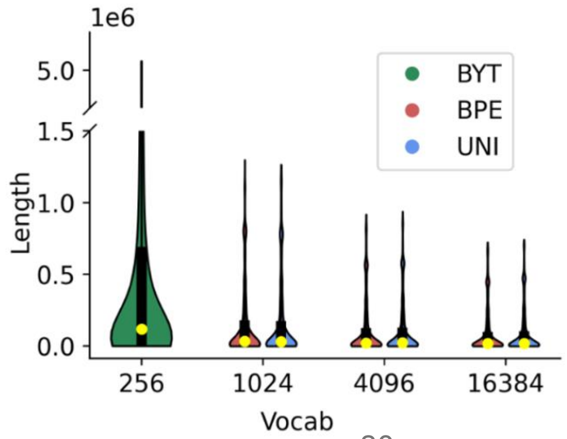
RAW



EXE



DIS



DEC

# Model: Designing an LMLM

**Objective:** train a large fully selective network on malware reps

# Model: Designing an LMLM

**Objective:** train a large fully selective network on malware reps

**Challenge:** tokenized code reps are too long for standard LLM

# Model: Designing an LMLM

II

**Objective:** train a large fully selective network on malware reps

**Challenge:** tokenized code reps are too long for standard LLM

**Proposal:** design a network for long-sequence training

# Model: Designing an LMLM

**Objective:** train a large fully selective network on malware reps

**Challenge:** tokenized code reps are too long for standard LLM

**Proposal:** design a network for long-sequence training

Architectures:

- HRRFormer (Linear Transformer)
- Mamba (Modern RNN)

Memory Analysis:

$$16C_1LH^2 + C_2\sqrt{L}TH$$

$$T \gg H \gg L$$

Alam et al. "HRRFormer" ICML 2024.

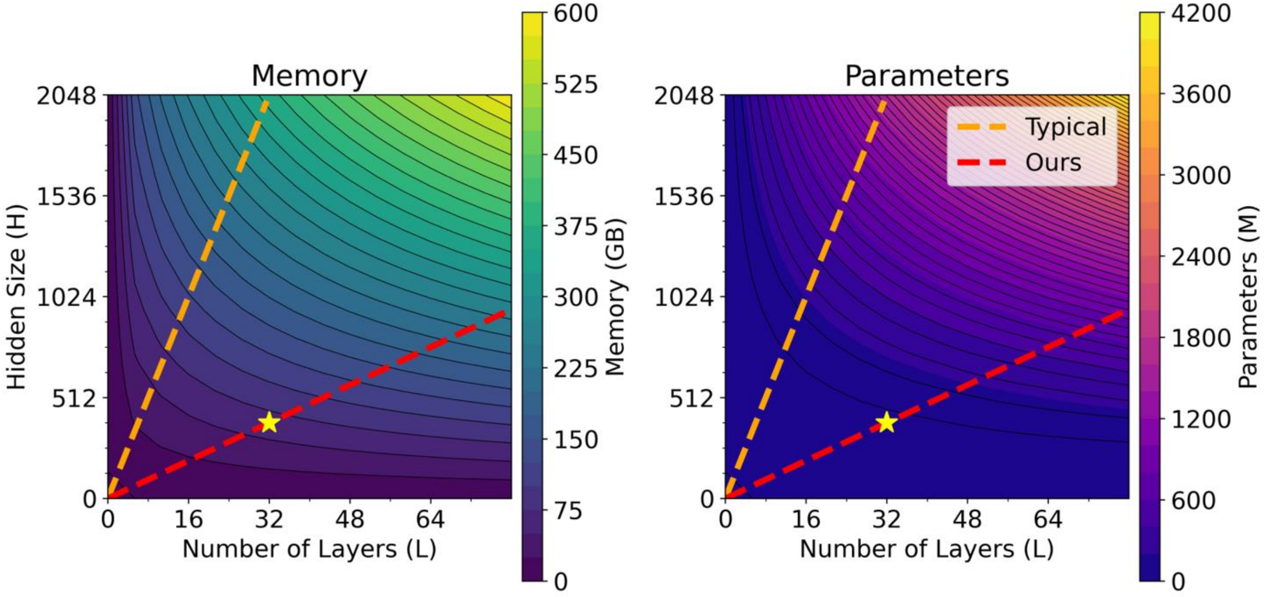
Gu & Dao "Mamba" COLM 2024.

# Model: Designing an LMLM

**Objective:** train a large fully selective network on malware reps

**Challenge:** tokenized code reps are too long for standard LLM

**Proposal:** design a network for long-sequence training



Memory Analysis:

$$16C_1LH^2 + C_2\sqrt{L}TH$$

$$T \gg H \gg L$$

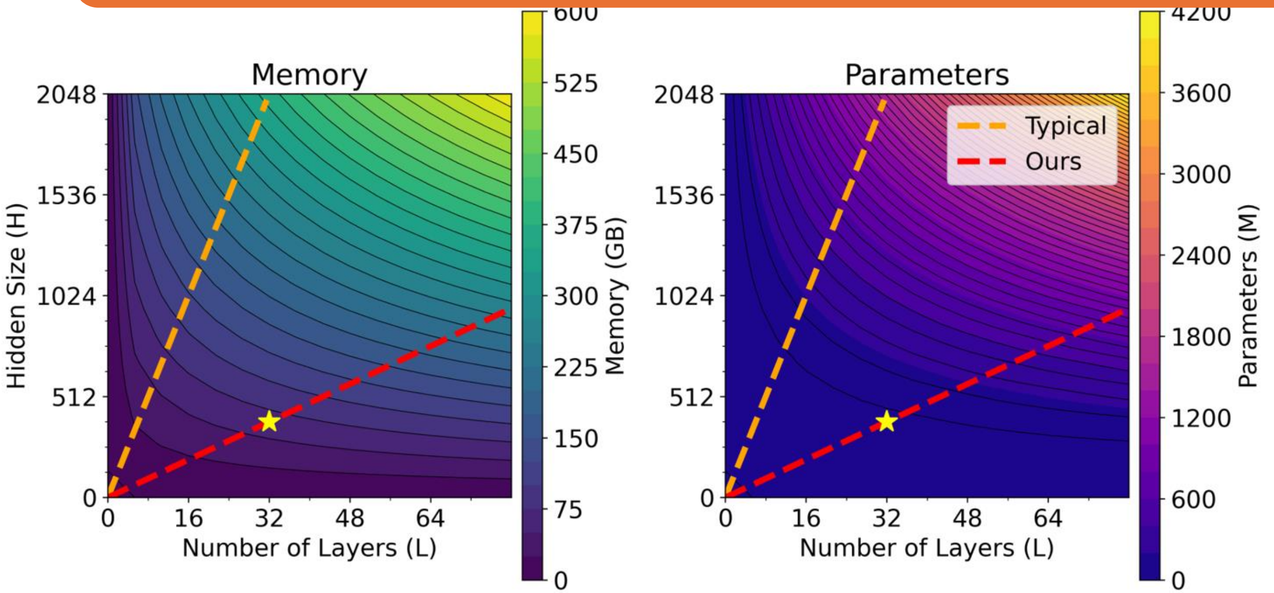
# Model: Designing an LMLM

**Objective:** train a large fully selective network on malware reps

**Challenge:** tokenized code reps are too long for standard LLM

**Proposal:** design a network for long-sequence training

**Finding:** LMLMs should be deeper and narrower than LLMs



Memory Analysis:

$$16C_1LH^2 + C_2\sqrt{L}TH$$

$$T \gg H \gg L$$

# Pretraining: Learning from Unlabeled Data

III

**Objective:** leverage vast quantities of unlabeled software

# Pretraining: Learning from Unlabeled Data

**Objective:** leverage vast quantities of unlabeled software

Supervised Learning:

- Labeled Data (Good/Bad)
- Train to Classify Good/Bad

Semi-Supervised Learning:

- Unlabeled Data
- Train Model to Reconstruct

# Pretraining: Learning from Unlabeled Data

**Objective:** leverage vast quantities of unlabeled software

**Challenge:** relatively unexplored in malware domain

Supervised Learning:

- Labeled Data (Good/Bad)
- Train to Classify Good/Bad

Semi-Supervised Learning:

- Unlabeled Data
- Train Model to Reconstruct

# Pretraining: Learning from Unlabeled Data

**Objective:** leverage vast quantities of unlabeled software

**Challenge:** relatively unexplored in malware domain

**Proposal:** compare common language modeling strategies

Semi-Supervised Learning:

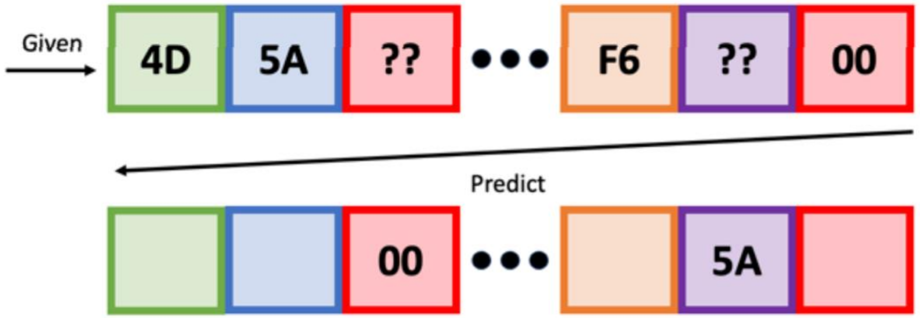
- Unlabeled Data
- Train Model to Reconstruct

# Pretraining: Learning from Unlabeled Data

**Objective:** leverage vast quantities of unlabeled software

**Challenge:** relatively unexplored in malware domain

**Proposal:** compare common language modeling strategies



**M**asked **L**anguage **M**odeling (MLM)

Semi-Supervised Learning:

- Unlabeled Data
- Train Model to Reconstruct

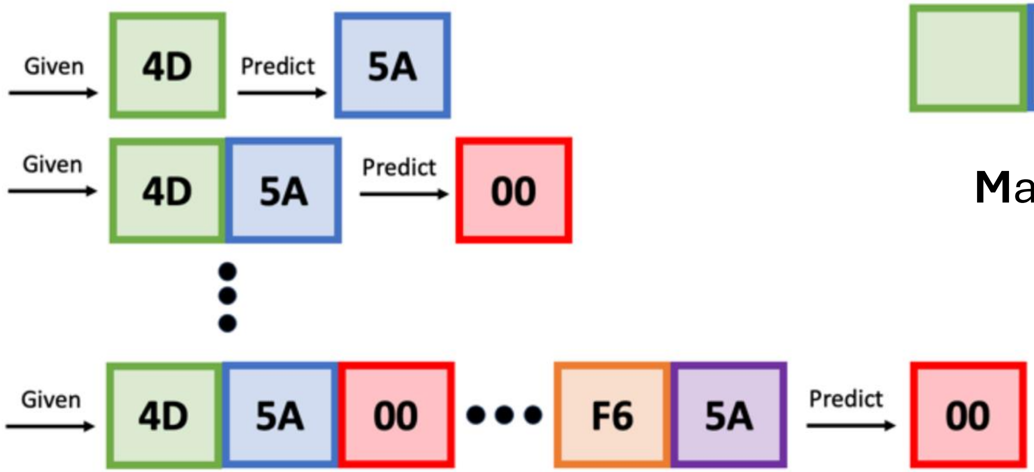
# Pretraining: Learning from Unlabeled Data

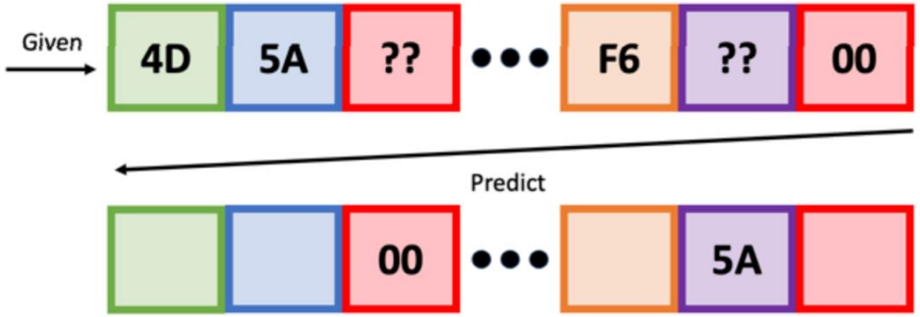
**Objective:** leverage vast quantities of unlabeled software

**Challenge:** relatively unexplored in malware domain

**Proposal:** compare common language modeling strategies

Causal Language Modeling (CLM)





Masked Language Modeling (MLM)

Semi-Supervised Learning:

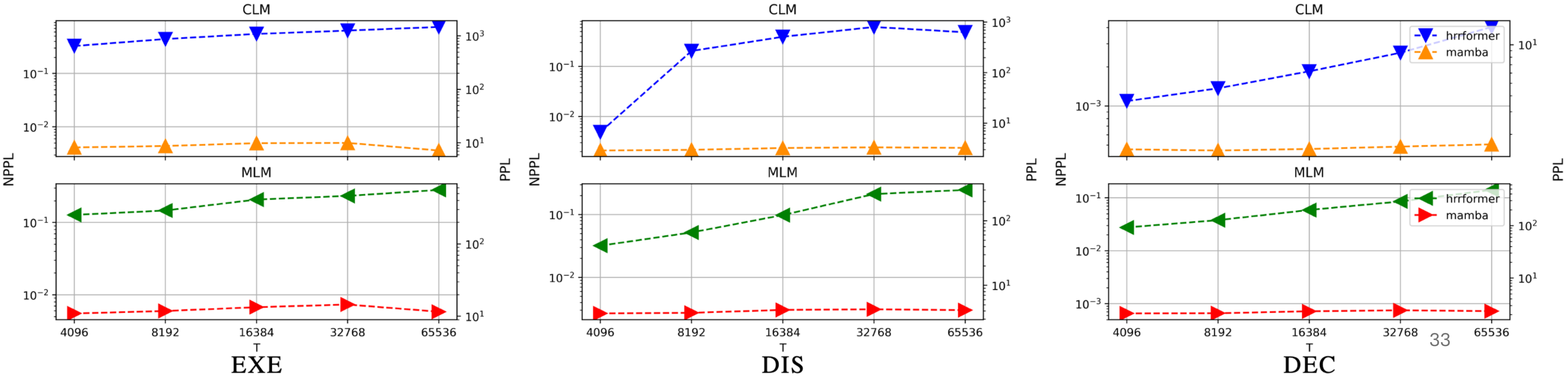
- Unlabeled Data
- Train Model to Reconstruct

# Pretraining: Learning from Unlabeled Data

**Objective:** leverage vast quantities of unlabeled software

**Challenge:** relatively unexplored in malware domain

**Proposal:** compare common language modeling strategies



# Pretraining: Learning from Unlabeled Data

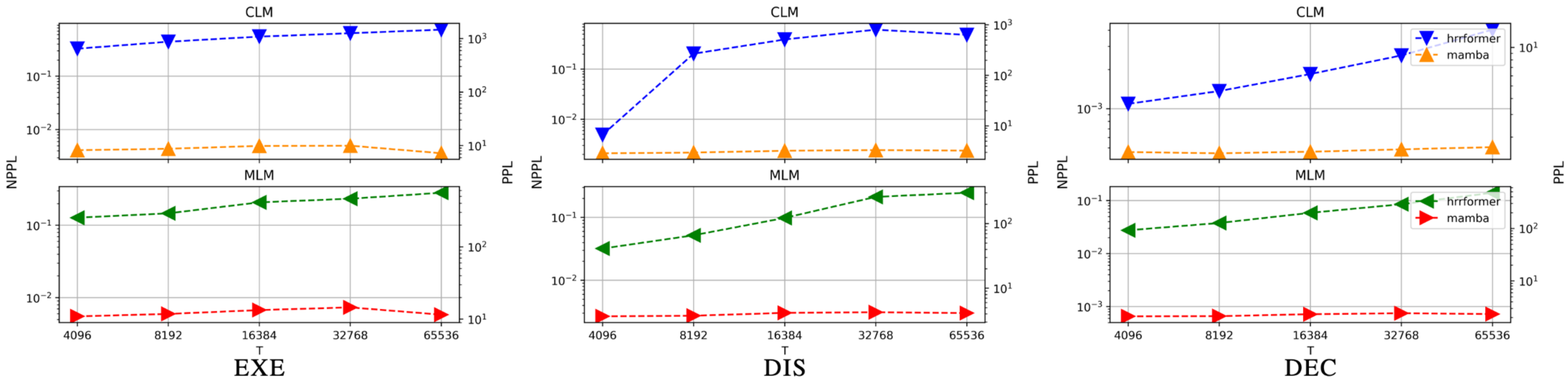
**Objective:** leverage vast quantities of unlabeled software

**Challenge:** relatively unexplored in malware domain

**Proposal:** compare common language modeling strategies

**Finding:** Mamba (RNN++) worked better than HRRFormer (Transformer)

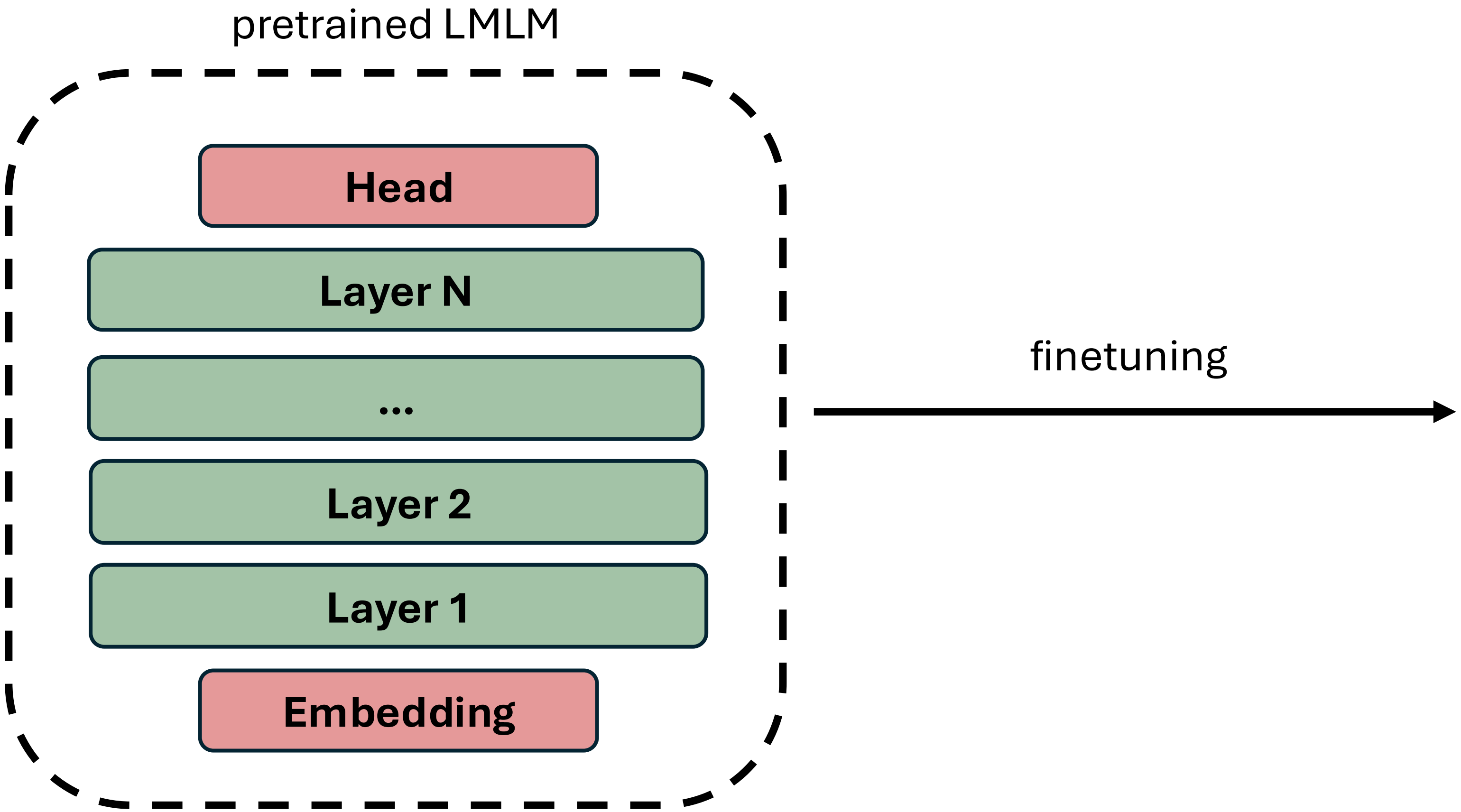
**Finding:** High-level code was easier to model than low-level code



# Finetuning: Practical Malware Analysis

IV

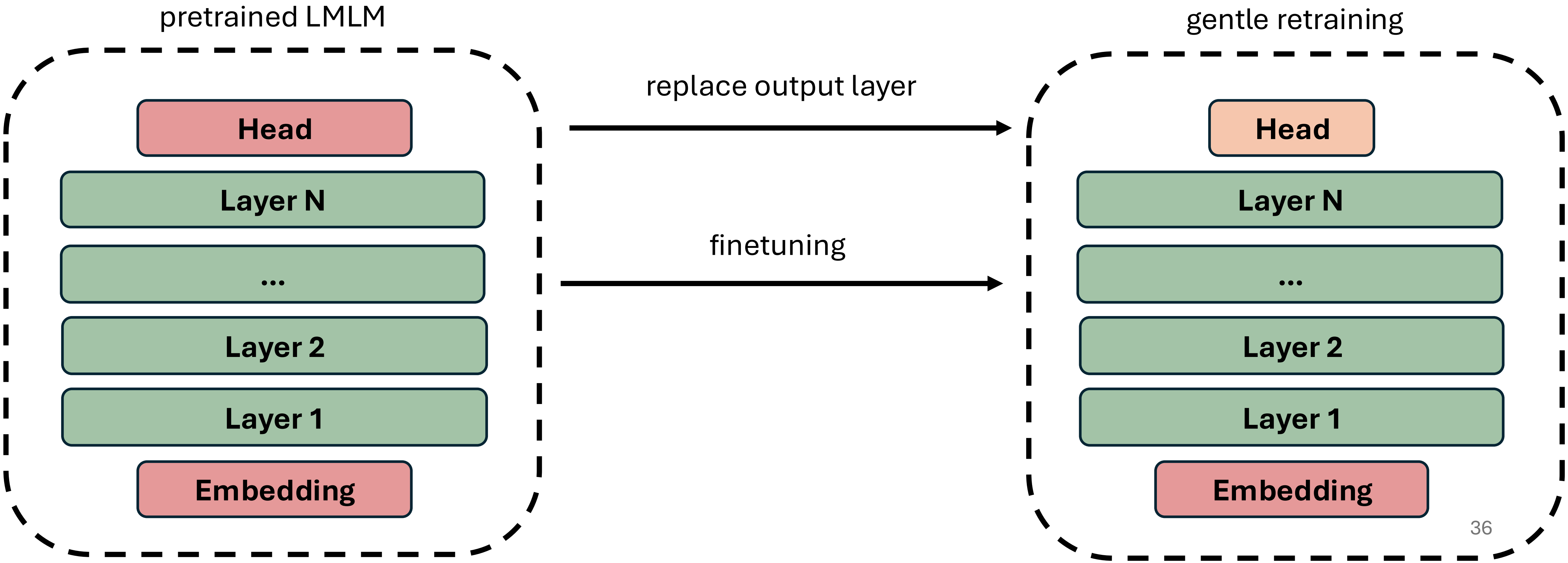
**Objective:** apply LMLM to useful malware classification problems



# Finetuning: Practical Malware Analysis

IV

**Objective:** apply LMLM to useful malware classification problems

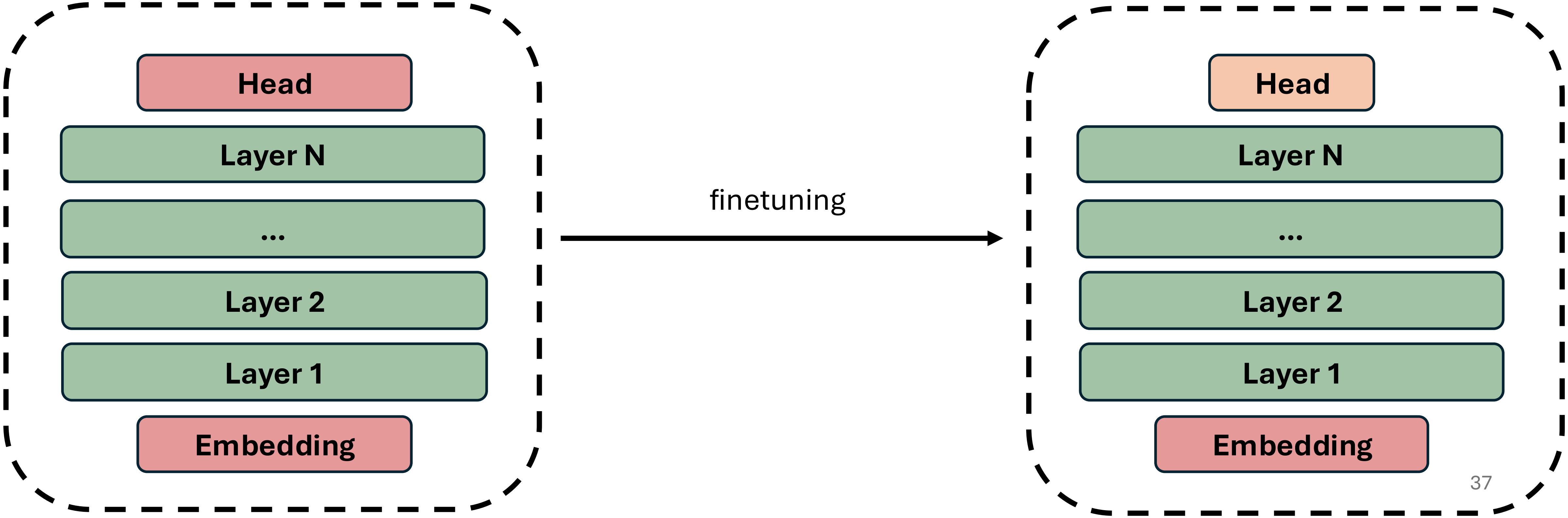


# Finetuning: Practical Malware Analysis

IV

**Objective:** apply LMLM to useful malware classification problems

**Challenge:** relatively unexplored in malware domain

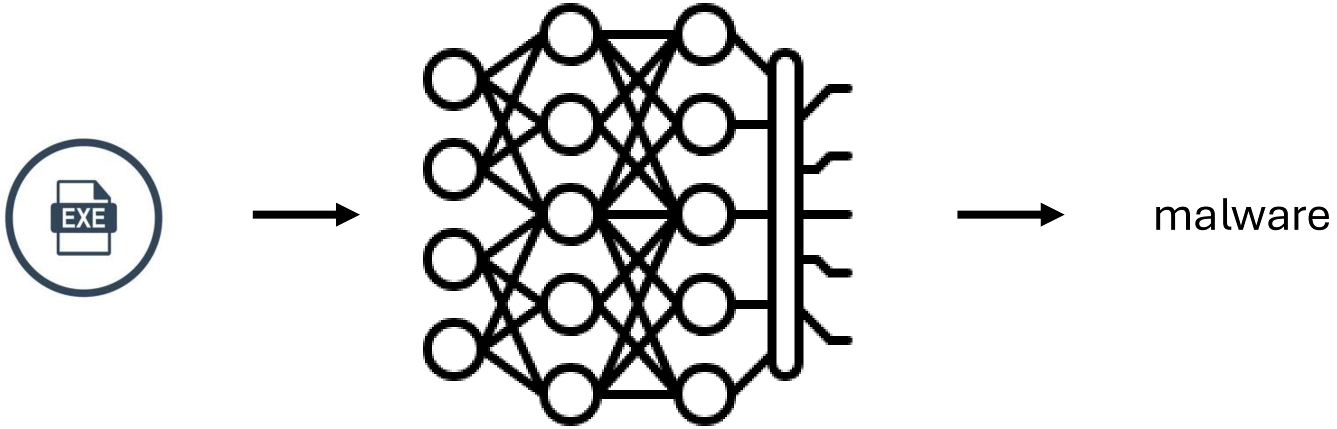


# Finetuning: Practical Malware Analysis

**Objective:** apply LMLM to useful malware classification problems

**Challenge:** relatively unexplored in malware domain

**Proposal:** assess performance of (pretrained) LMLM across three tasks



Malware **DET**ection

- Is it malware?
- Binary classification

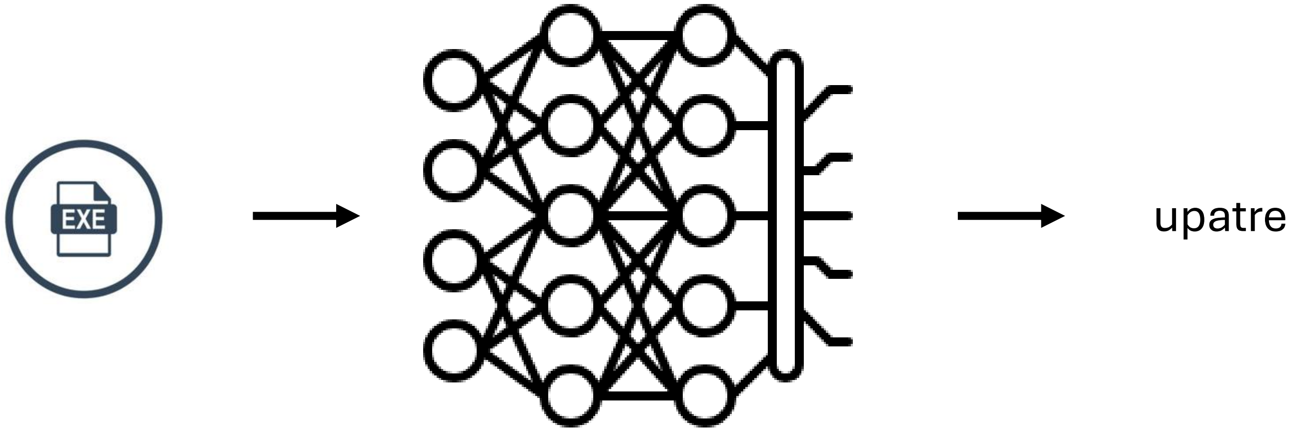
# Finetuning: Practical Malware Analysis

IV

**Objective:** apply LMLM to useful malware classification problems

**Challenge:** relatively unexplored in malware domain

**Proposal:** assess performance of (pretrained) LMLM across three tasks



upatre

Malware **DET**ection

- Is it malware?
- <u>Binary</u> classification

**FAM**ily Classification

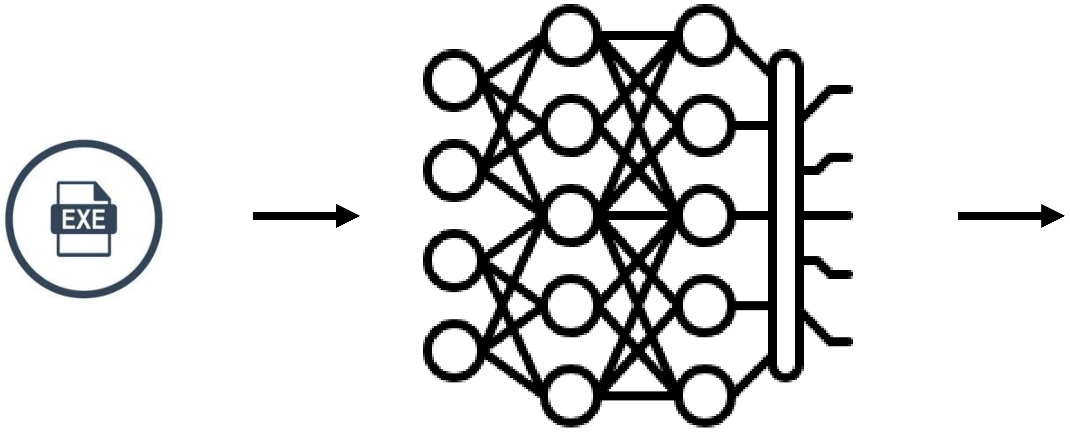
- Which family is it?
- <u>Multiclass</u> classification

# Finetuning: Practical Malware Analysis

**Objective:** apply LMLM to useful malware classification problems

**Challenge:** relatively unexplored in malware domain

**Proposal:** assess performance of (pretrained) LMLM across three tasks



dropper
password stealer
packed

Malware **DET**ection

- Is it malware?
- <u>Binary</u> classification

**FAM**ily Classification

- Which family is it?
- <u>Multiclass</u> classification

**BEH**avioral Tagging

- What does it do?
- <u>Multilabel</u> classification

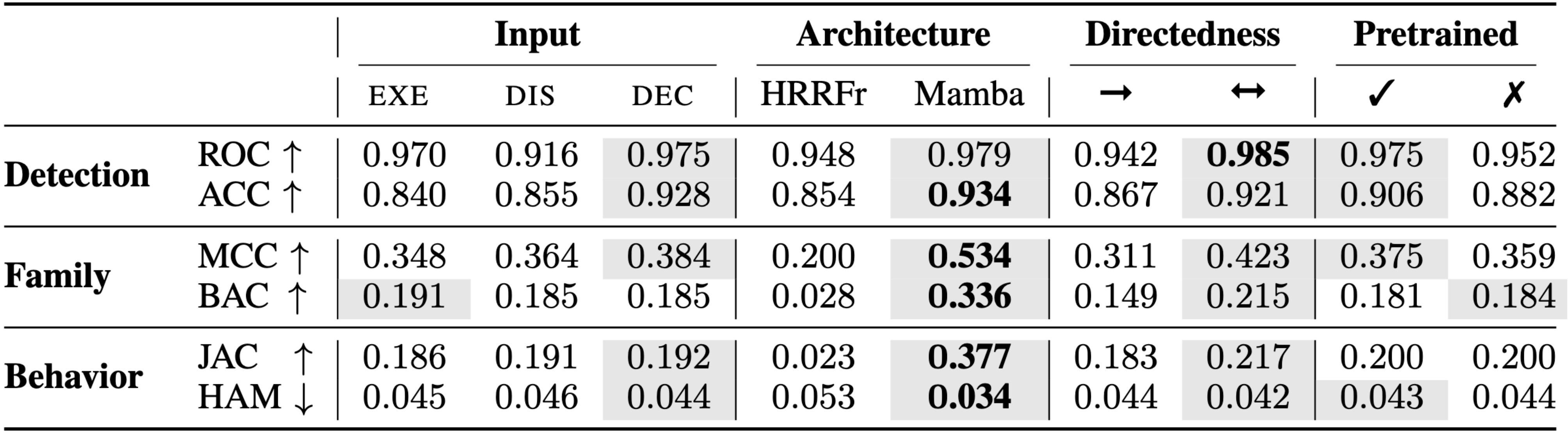
# Finetuning: Practical Malware Analysis

IV

**Objective:** apply LMLM to useful malware classification problems

**Challenge:** relatively unexplored in malware domain

**Proposal:** assess performance of (pretrained) LMLM across three tasks



| | | Input | | | Architecture | | Directedness | | Pretrained | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | EXE | DIS | DEC | HRRFr | Mamba | → | ↔ | ✓ | ✗ |
| **Detection** | ROC ↑ | 0.970 | 0.916 | 0.975 | 0.948 | 0.979 | 0.942 | **0.985** | 0.975 | 0.952 |
| | ACC ↑ | 0.840 | 0.855 | 0.928 | 0.854 | **0.934** | 0.867 | 0.921 | 0.906 | 0.882 |
| **Family** | MCC ↑ | 0.348 | 0.364 | 0.384 | 0.200 | **0.534** | 0.311 | 0.423 | 0.375 | 0.359 |
| | BAC ↑ | 0.191 | 0.185 | 0.185 | 0.028 | **0.336** | 0.149 | 0.215 | 0.181 | 0.184 |
| **Behavior** | JAC ↑ | 0.186 | 0.191 | 0.192 | 0.023 | **0.377** | 0.183 | 0.217 | 0.200 | 0.200 |
| | HAM ↓ | 0.045 | 0.046 | 0.044 | 0.053 | **0.034** | 0.044 | 0.042 | 0.043 | 0.044 |

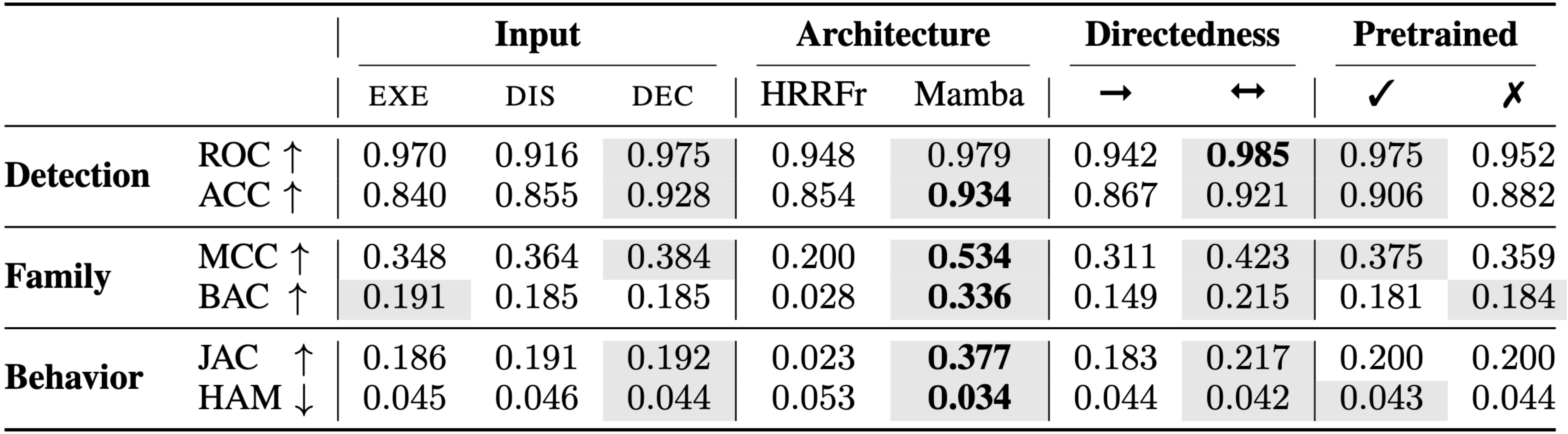
# Finetuning: Practical Malware Analysis

**Objective:** apply LMLM to useful malware classification problems

**Challenge:** relatively unexplored in malware domain

**Proposal:** assess performance of (pretrained) LMLM across three tasks

**Finding:** macro-average analysis points to decompiled input, Mamba, bidirectionality, and pretraining as being the most effective strategies



| | | Input | | | Architecture | | Directedness | | Pretrained | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | EXE | DIS | DEC | HRRFr | Mamba | → | ↔ | ✓ | ✗ |
| **Detection** | ROC ↑ | 0.970 | 0.916 | 0.975 | 0.948 | 0.979 | 0.942 | **0.985** | 0.975 | 0.952 |
| | ACC ↑ | 0.840 | 0.855 | 0.928 | 0.854 | **0.934** | 0.867 | 0.921 | 0.906 | 0.882 |
| **Family** | MCC ↑ | 0.348 | 0.364 | 0.384 | 0.200 | **0.534** | 0.311 | 0.423 | 0.375 | 0.359 |
| | BAC ↑ | 0.191 | 0.185 | 0.185 | 0.028 | **0.336** | 0.149 | 0.215 | 0.181 | 0.184 |
| **Behavior** | JAC ↑ | 0.186 | 0.191 | 0.192 | 0.023 | **0.377** | 0.183 | 0.217 | 0.200 | 0.200 |
| | HAM ↓ | 0.045 | 0.046 | 0.044 | 0.053 | **0.034** | 0.044 | 0.042 | 0.043 | 0.044 |

# Conclusion

**What we did:**

Investigate the feasibility of training **L**arge **M**alware **L**anguage **M**odels – the malware analogue to Large Language Models

# Conclusion

**What we did:**

Investigate the feasibility of training **L**arge **M**alware **L**anguage **M**odels – the malware analogue to Large Language Models

**What we found:**

LLM techniques can be adapted, with care, to the malware domain, and can improve performance on practical malware analysis tasks

# Conclusion

**What we did:**

Investigate the feasibility of training **L**arge **M**alware **L**anguage **M**odels – the malware analogue to Large Language Models

**What we found:**

LLM techniques can be adapted, with care, to the malware domain, and can improve performance on practical malware analysis tasks

**What remains:**

Deeper investigation into (un)packing, robustnessness, and resistance to concept drift.

Applications to adversarial examples?

# Questions?