



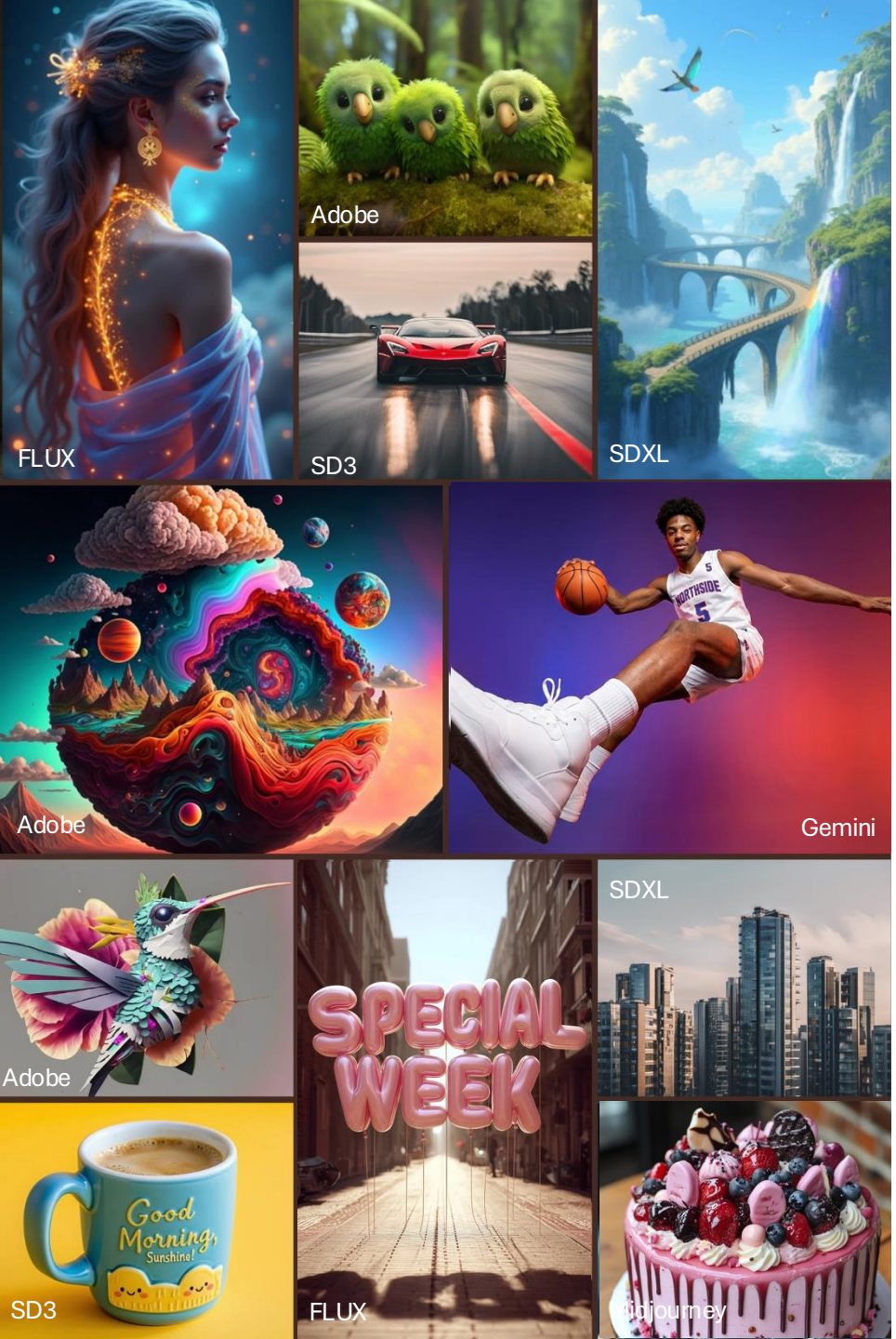
SAND Lab
security, algorithms, networks and data

Link to paper

Identifying Provenance of Generative Text-to-Image Models

Anna Yoo Jeong Ha, Wenxin Ding*, Stanley Wu*, Shawn Shan,
Haitao Zheng, Ben Y. Zhao

University of Chicago



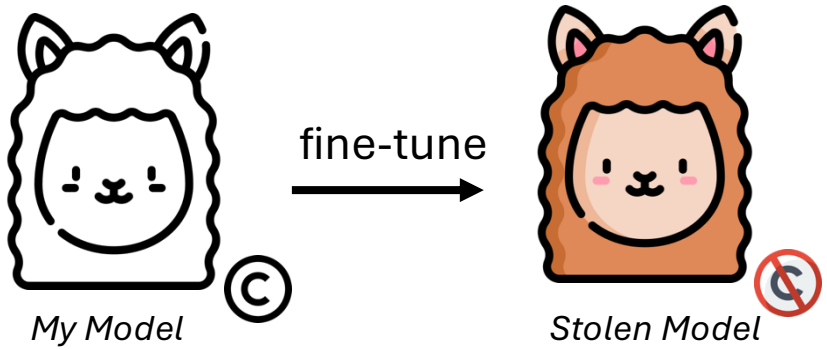
A model's provenance claims are difficult to verify in a black-box setting

A deployed API reveals outputs—not weights, training data, or training history.

Outputs alone do not directly reveal whether a model was trained from scratch or fine-tuned from an existing model.

The Provenance Problem

Leaked or Stolen Model



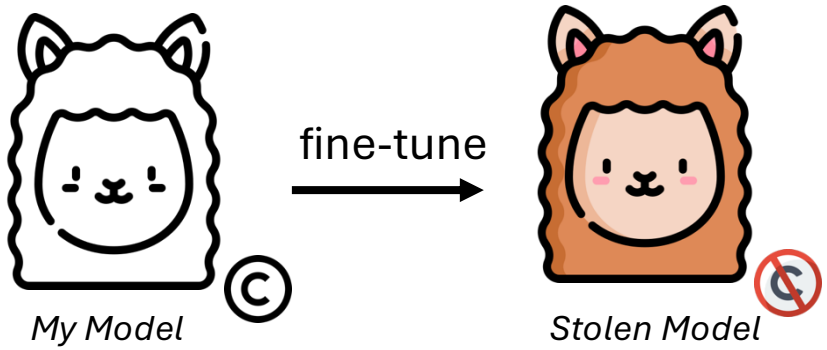
Did someone repackage my model as their own?

Model Claim Verification

Misuse Attribution

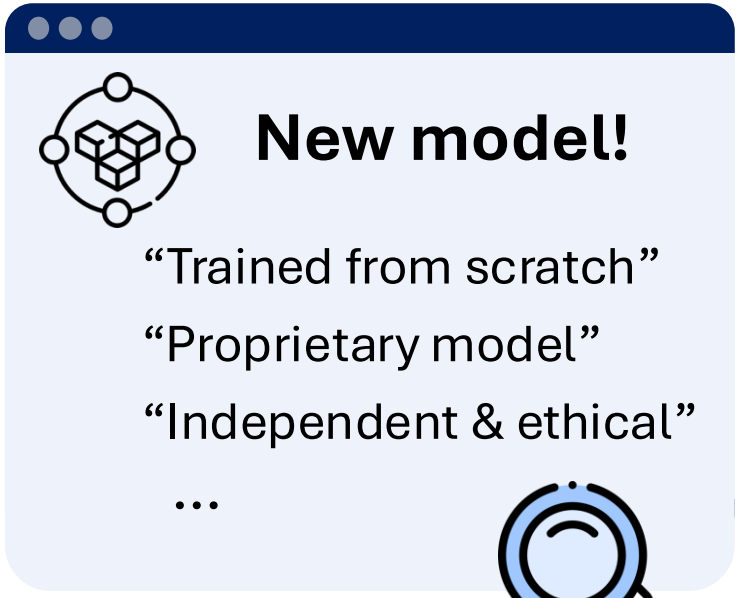
The Provenance Problem

Leaked or Stolen Model



Did someone repackage my model as their own?

Model Claim Verification

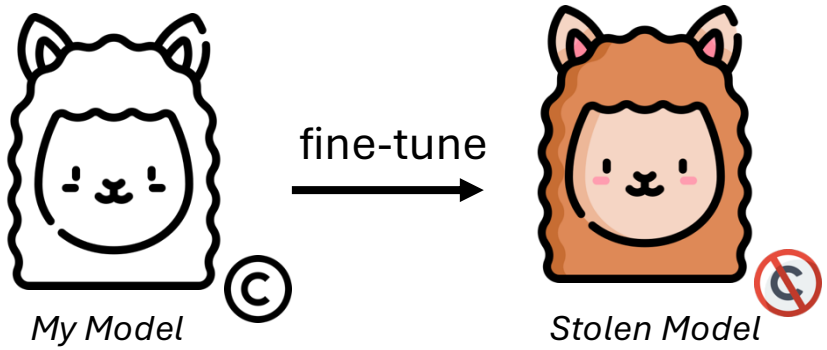


Can we verify these claims?

Misuse Attribution

The Provenance Problem

Leaked or Stolen Model



Did someone repackage my model as their own?

Model Claim Verification

New model!

"Trained from scratch"

"Proprietary model"

"Independent & ethical"

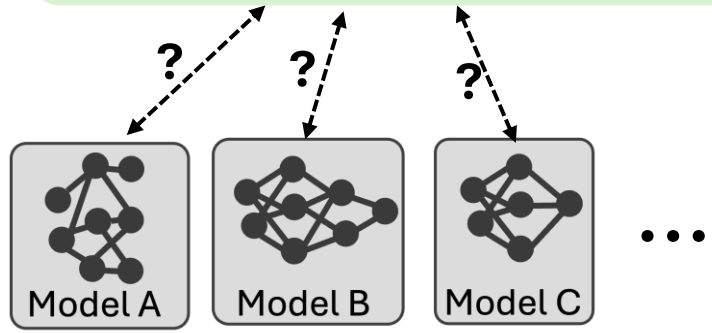
...



Can we verify these claims?

Misuse Attribution

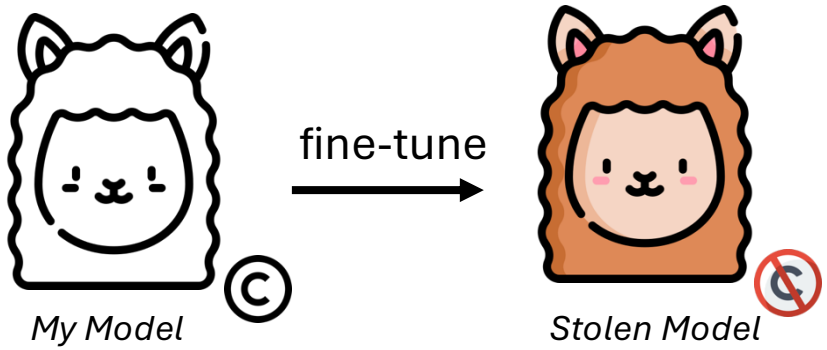
Suspicious Model Service



Which model does this suspicious service come from?

The Provenance Problem

Leaked or Stolen Model



Did someone repackage my model as their own?

Model Claim Verification

New model!

"Trained from scratch"

"Proprietary model"

"Independent & ethical"

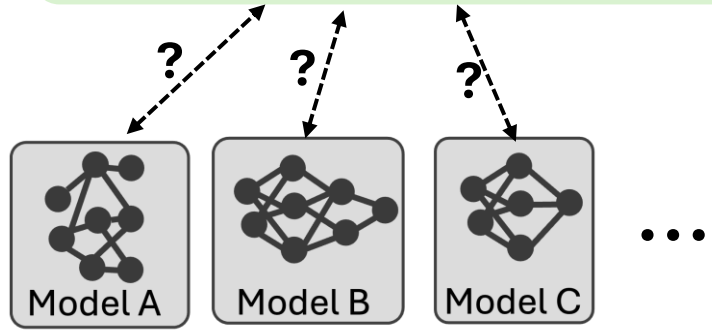
...



Can we verify these claims?

Misuse Attribution

Suspicious Model Service



Which model does this suspicious service come from?

Core problem: we need verifiable evidence about where a deployed model came from.

Prior work requires stronger assumptions

Existing tools are useful, but assume access or intervention before deployment

Watermarking

- Intentionally embed signals during training
- Signal may weaken after additional fine-tuning

Prior work requires stronger assumptions

Existing tools are useful, but assume access or intervention before deployment

Watermarking

- Intentionally embed signals during training
- Signal may weaken after additional fine-tuning

Fingerprinting

- Uses adversarial prompts to trigger unintended outputs
- Responses transfer to *unrelated* models or fail under simple defenses

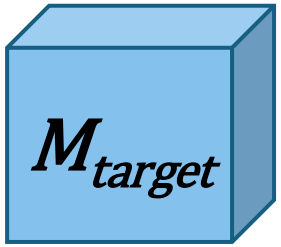
BOTH REQUIRE COLLABORATION FROM THE MODEL OWNER & CAN BE BRITTLE

Can we determine if a model was fine-tuned from a known base model?

*Complete Black-box Access

Threat Model: Black-box provenance setting

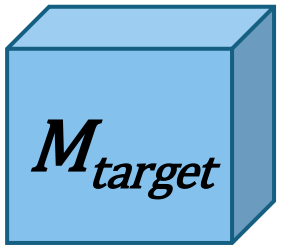
Model Trainer 😈



Black-box access only

Threat Model: Black-box provenance setting

Model Trainer

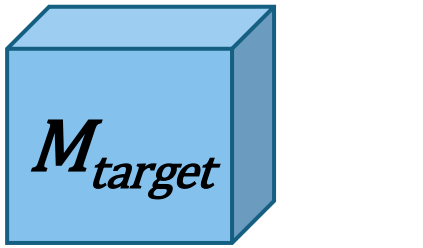


Black-box access only

- Trains M_{target} from scratch or fine-tunes an existing base model
- Releases M_{target}
- Provides black-box access only

Threat Model: Black-box provenance setting

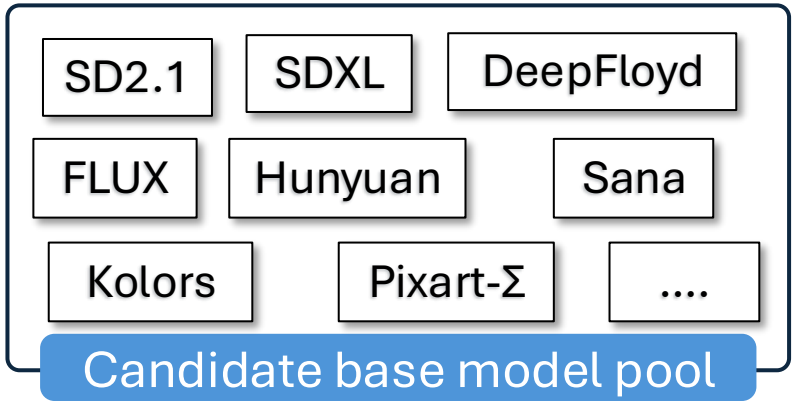
Model Trainer



Black-box access only

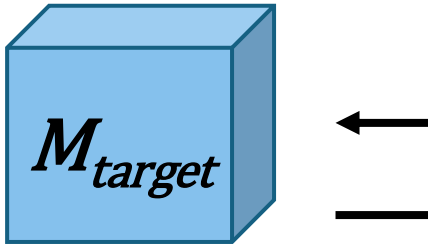
- Trains M_{target} from scratch or fine-tunes an existing base model
- Releases M_{target}
- Provides black-box access only

Provenance Investigator



Threat Model: Black-box provenance setting

Model Trainer



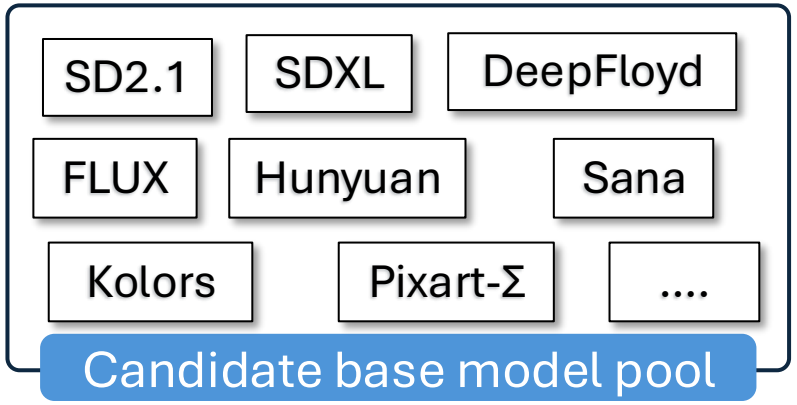
Black-box access only

prompts

generated images



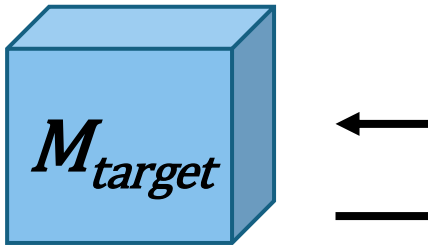
Provenance Investigator



- Trains M_{target} from scratch or fine-tunes an existing base model
- Releases M_{target}
- Provides black-box access only

Threat Model: Black-box provenance setting

Model Trainer



Black-box access only

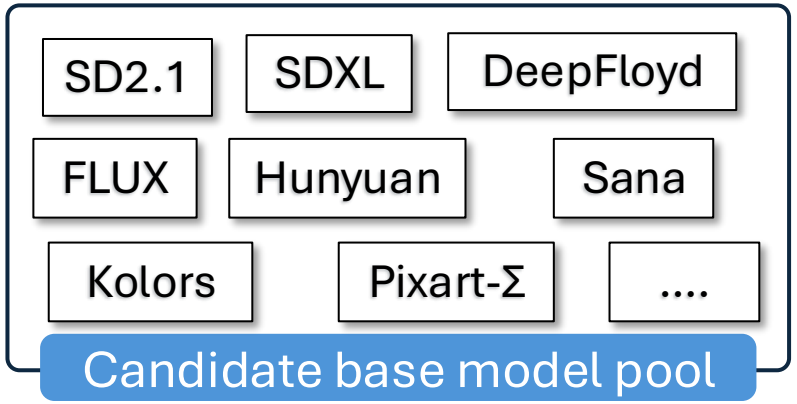
prompts

generated images



- Trains M_{target} from scratch or fine-tunes an existing base model
- Releases M_{target}
- Provides black-box access only

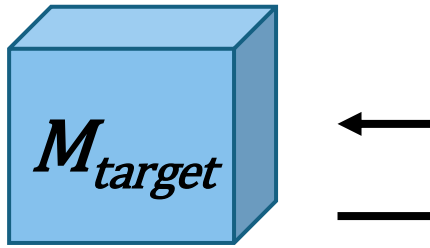
Provenance Investigator



- **Decision 1**: M_{target} is fine-tuned from a base model in the pool

Threat Model: Black-box provenance setting

Model Trainer



Black-box access only

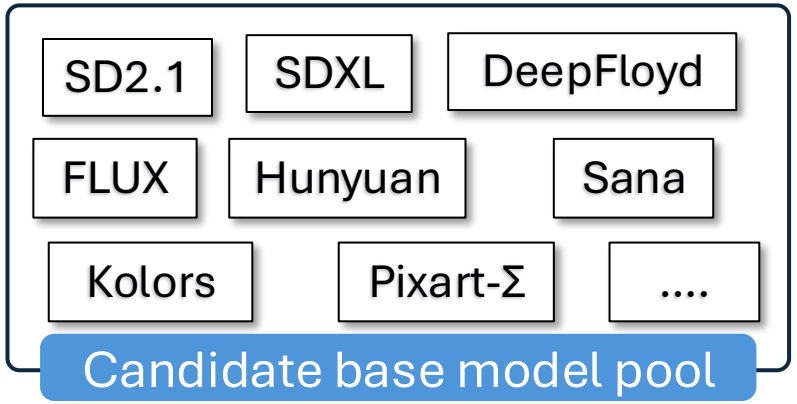
prompts

generated images



- Trains M_{target} from scratch or fine-tunes an existing base model
- Releases M_{target}
- Provides black-box access only

Provenance Investigator



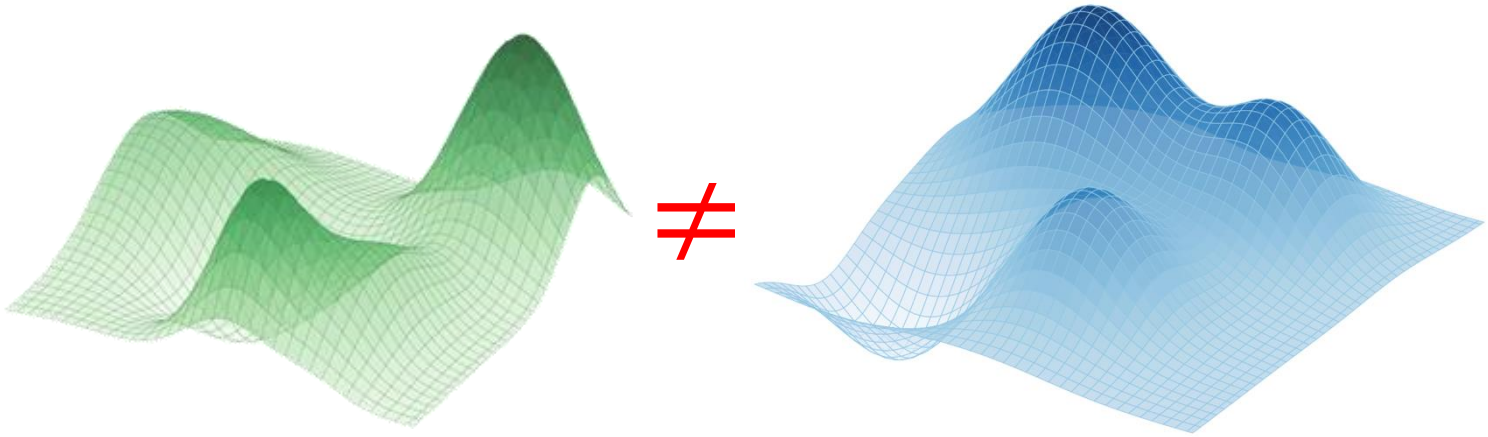
- **Decision 1**: M_{target} is fine-tuned from a base model in the pool
- **Decision 2**: M_{target} is trained from scratch, or fine-tuned from a base model not in the pool

Insight 1: Fine-tuning preserves model's feature distribution

Fine-tuned models remain closer to their parent than to independently trained models.

Base model A
(trained from scratch)

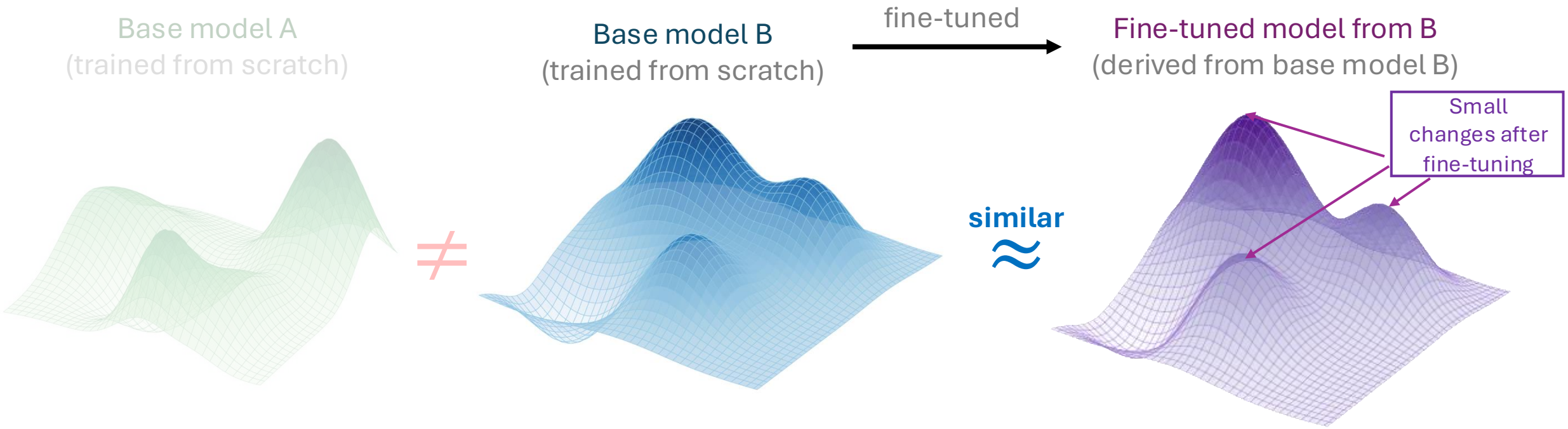
Base model B
(trained from scratch)



Differences in data, architecture, and training procedure produce **distinct** generative behavior and output feature distributions.

Insight 1: Fine-tuning preserves model's feature distribution

Fine-tuned models remain closer to their parent than to independently trained models.

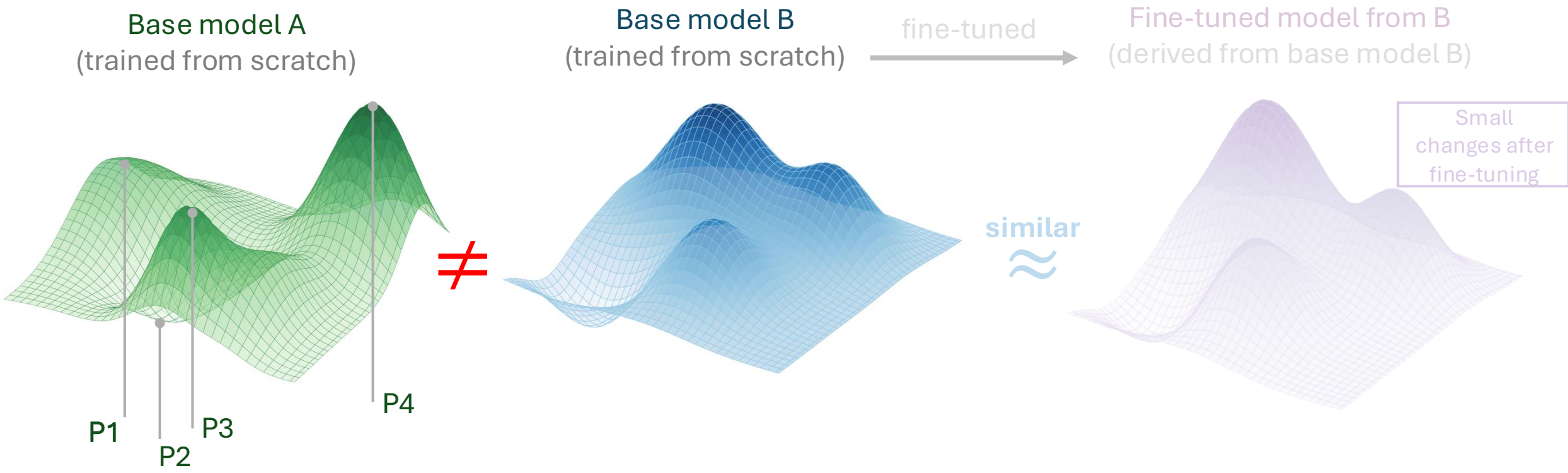


Differences in data, architecture, and training procedure produce **distinct** generative behavior and output feature distributions.

Fine-tuning introduces local changes and preserves a lot of the parent structure.

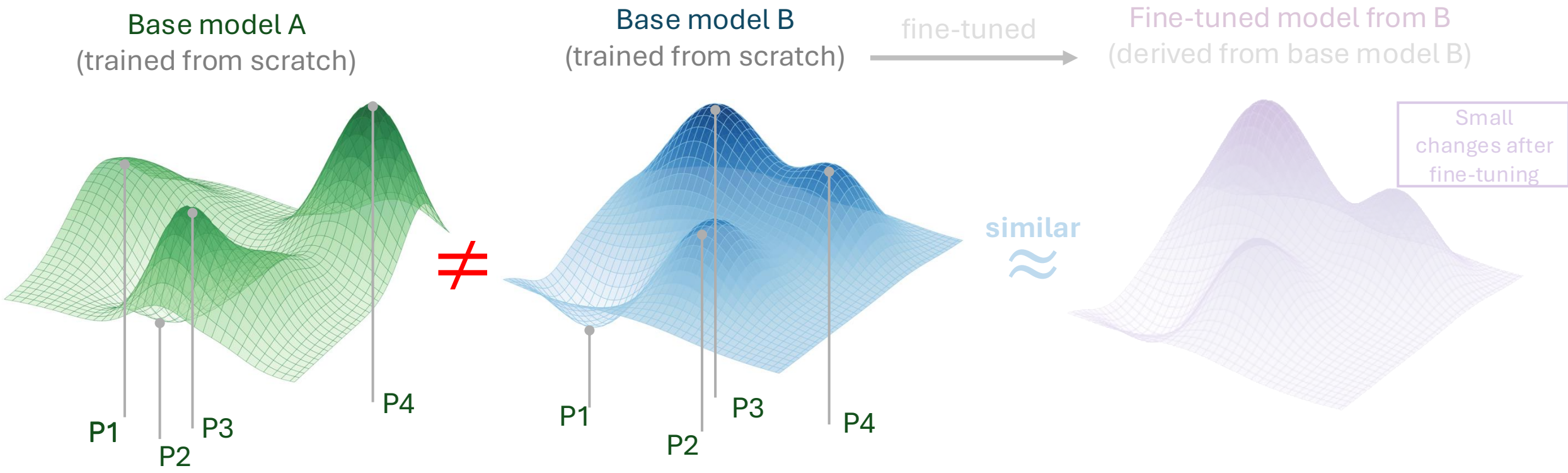
Insight 2: Prompts probe feature-space distributions

Under black-box access, prompts let us probe hidden feature-space distributions.



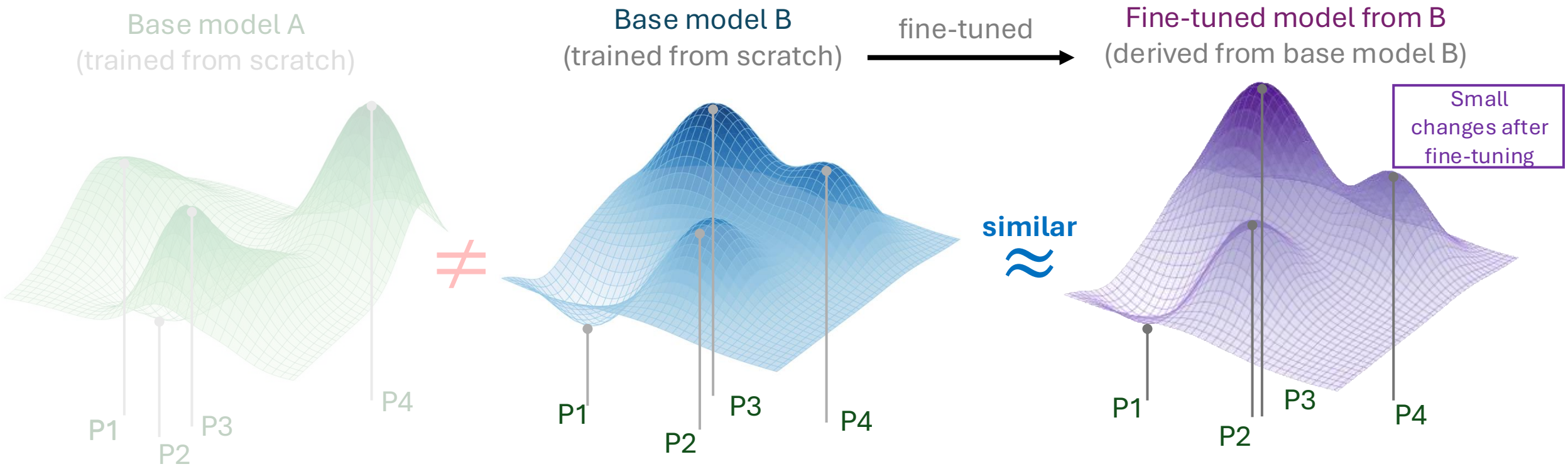
Insight 2: Prompts probe feature-space distributions

Under black-box access, prompts let us probe hidden feature-space distributions.



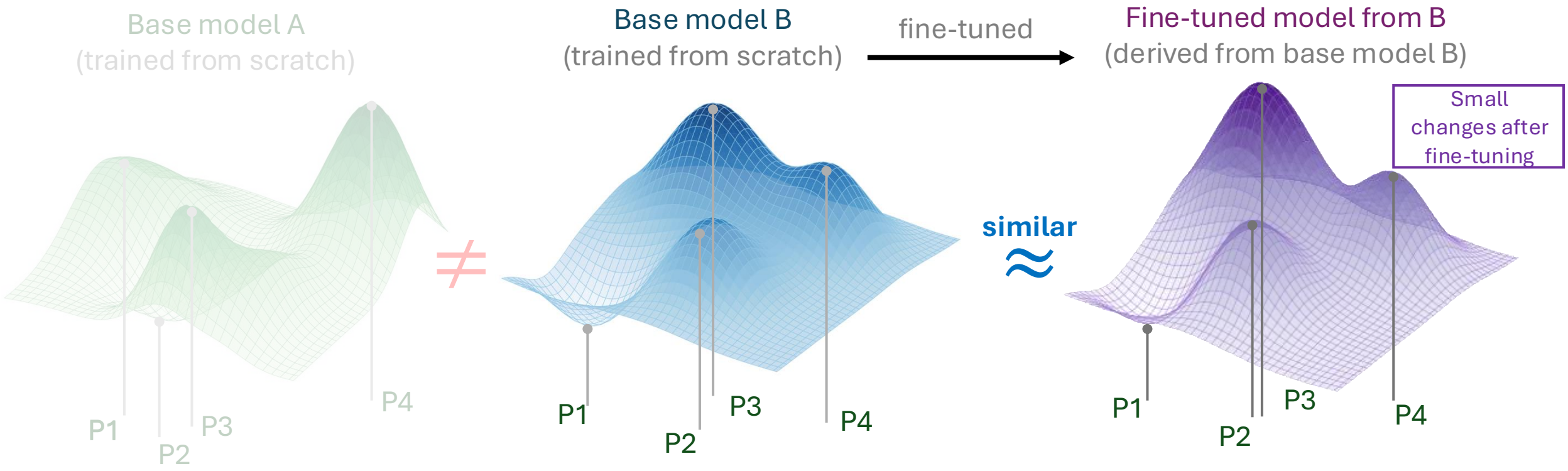
Insight 2: Prompts probe feature-space distributions

Under black-box access, prompts let us probe hidden feature-space distributions.



Insight 2: Prompts probe feature-space distributions

Under black-box access, prompts let us probe hidden feature-space distributions.



When given identical prompts, independent models generate different response patterns. In contrast, fine-tuned models exhibit response patterns that **closely resemble** those of their base model.

Common Objects + Detailed Prompts

- Common objects → stable feature distributions

"A beautifully decorated **cake** …."

Model: FLUX

"A modern, stylish gray fabric **sofa** with…."

Sample 1 | Sample 2 | Sample 3 | Sample 4



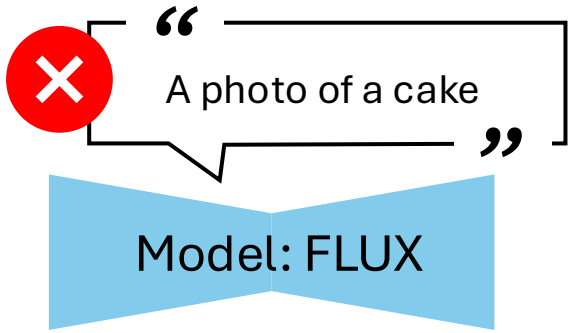







Common Objects + Detailed Prompts

- Common objects → stable feature distributions
- Detailed prompts → lower variations in features across samples



Sample 1



Sample 2



Sample 3



Sample 4



Common Objects + Detailed Prompts

- Common objects → stable feature distributions
- Detailed prompts → lower variations in features across samples

"A photo of a cake"

Model: FLUX

"A beautifully decorated cake placed on a table. The cake is multi-layered with intricate floral designs in pastel colors like pink, blue, and white. It has elegant frosting details and is surrounded by soft lighting for an inviting look. The table is set with a few matching plates and utensils, giving a warm and festive atmosphere."

Sample 1 | Sample 2 | Sample 3 | Sample 4

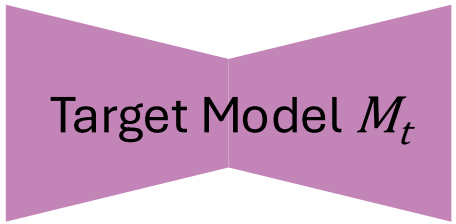




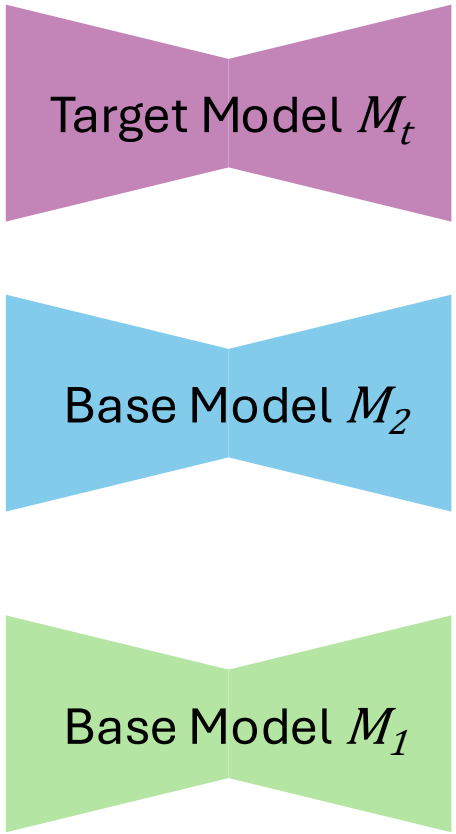





Putting It Together



Putting It Together



Putting It Together

1. Generate images with detailed prompt p

Target Model M_t →



Base Model M_2 →



Base Model M_1 →



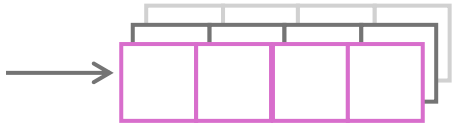
Putting It Together

1. Generate images with detailed prompt p

2. Extract features and reduce dimension (S2-CLIP)

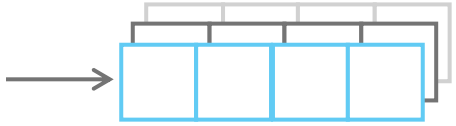
Target Model M_t





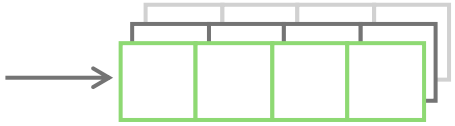
Base Model M_2





Base Model M_1





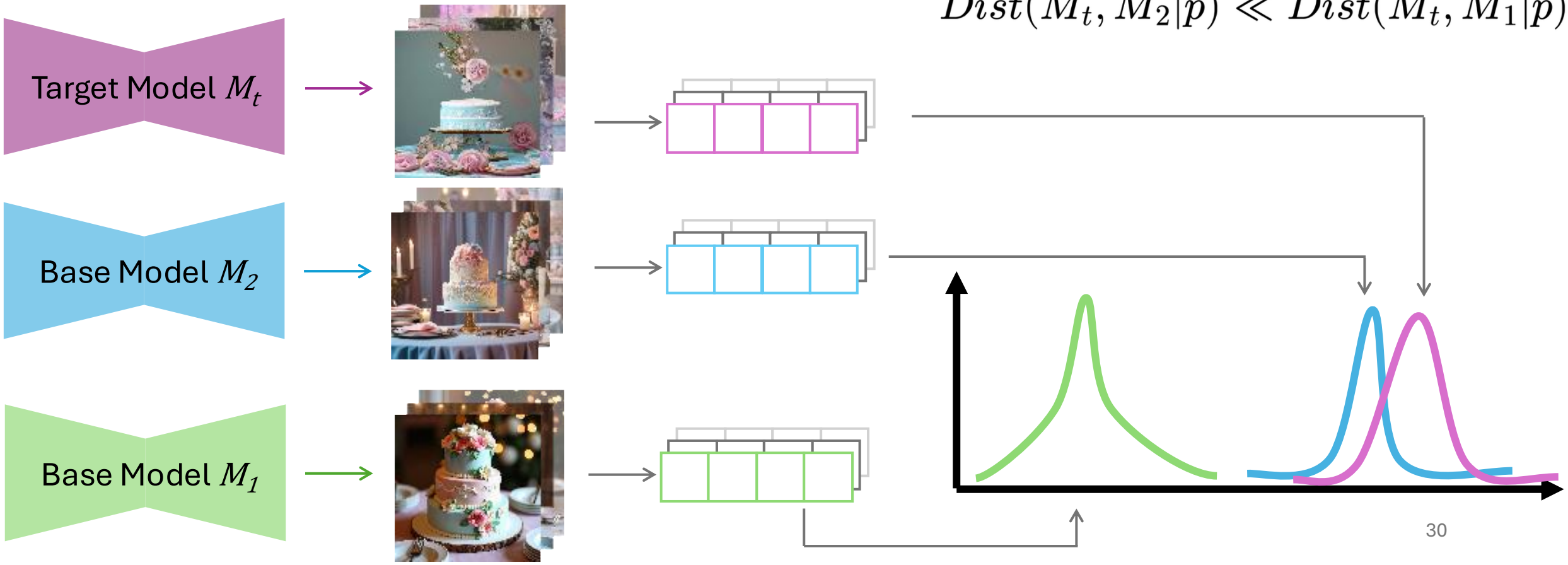
Putting It Together

1. Generate images with detailed prompt p

2. Extract features and reduce dimension (S2-CLIP)

3. Compute Jensen-Shannon divergence

$$Dist(M_t, M_2|p) \ll Dist(M_t, M_1|p)$$

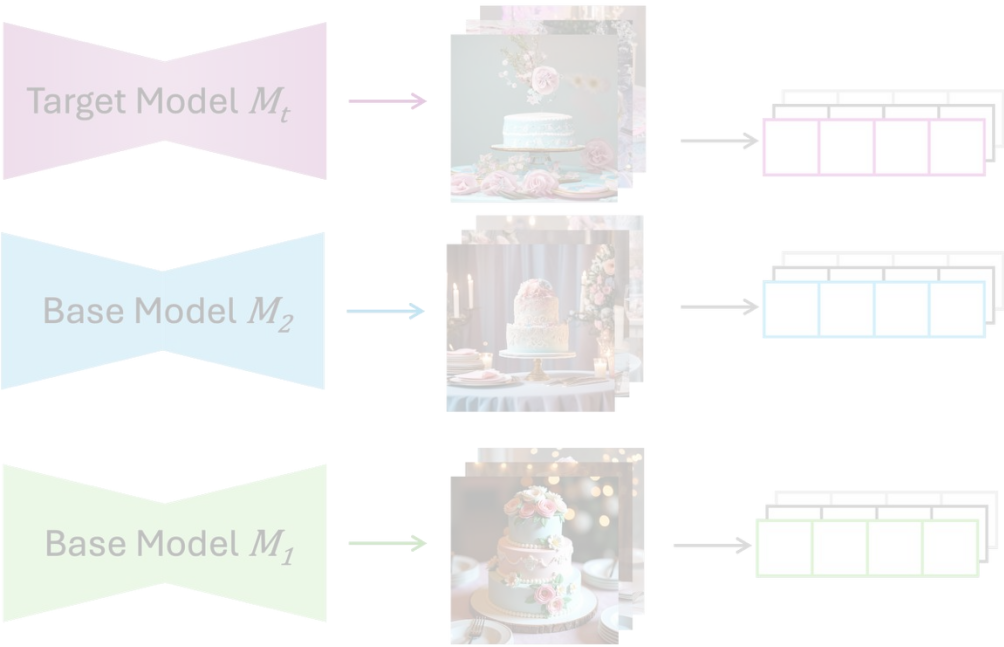


Putting It Together

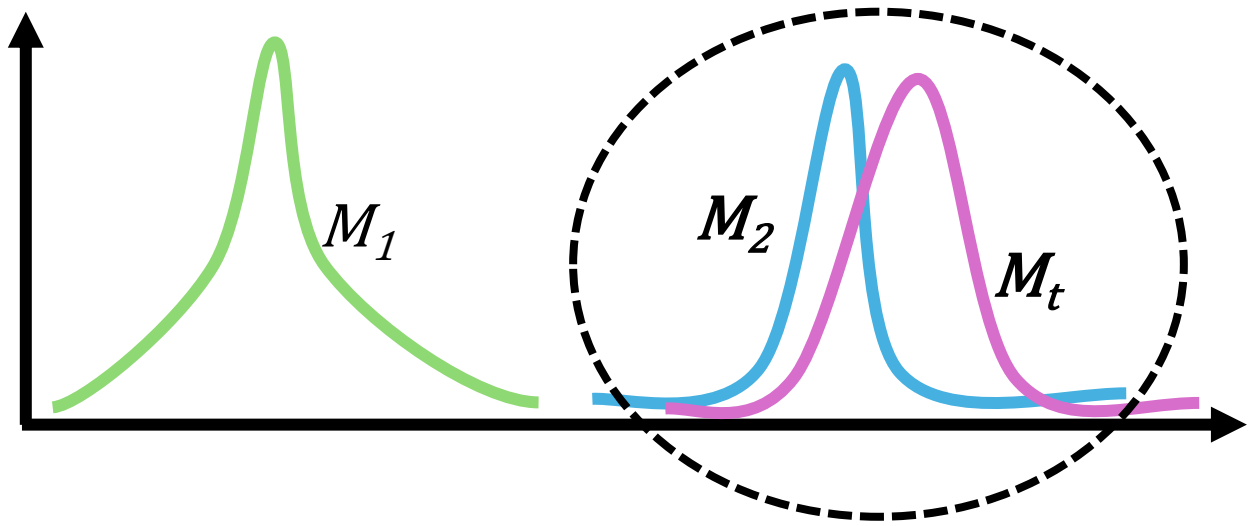
1. Generate images with detailed prompt p

2. Extract features and reduce dimension (S2-CLIP)

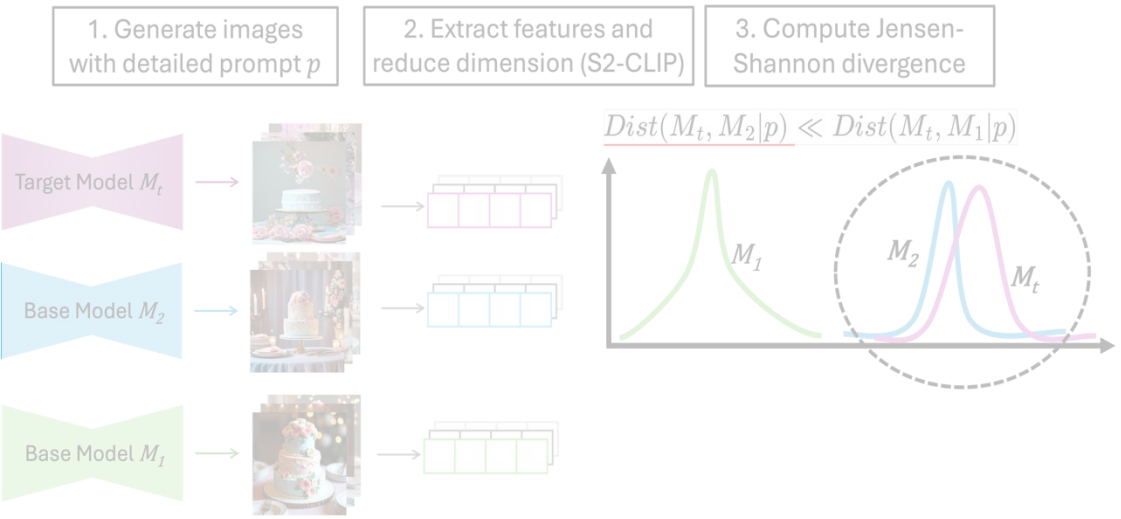
3. Compute Jensen-Shannon divergence



$$Dist(M_t, M_2|p) \ll Dist(M_t, M_1|p)$$



Putting It Together



4. Determine Provenance

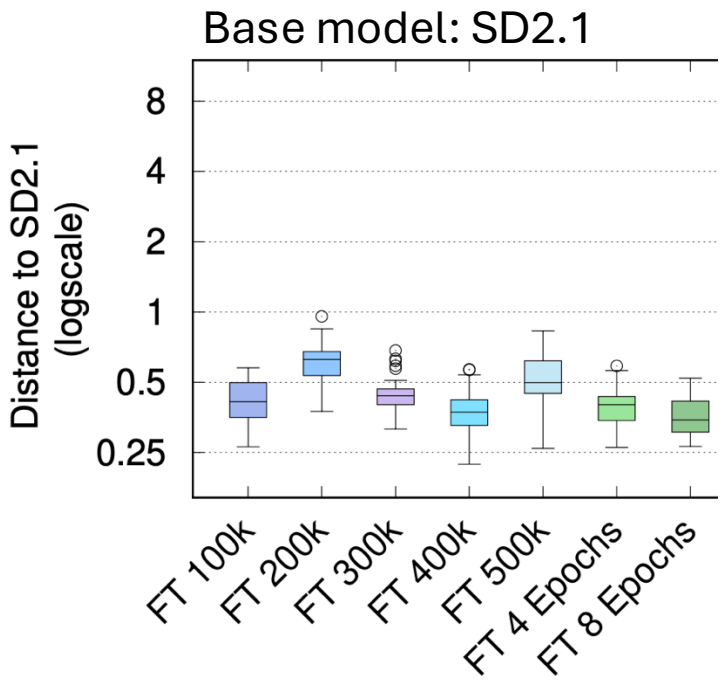
Is the smallest distance below a *z-test threshold*?

YES → Likely parent = M_2

NO → $NULL$ (No matching base in the pool)

Fine-tuned models stay closer to their base model

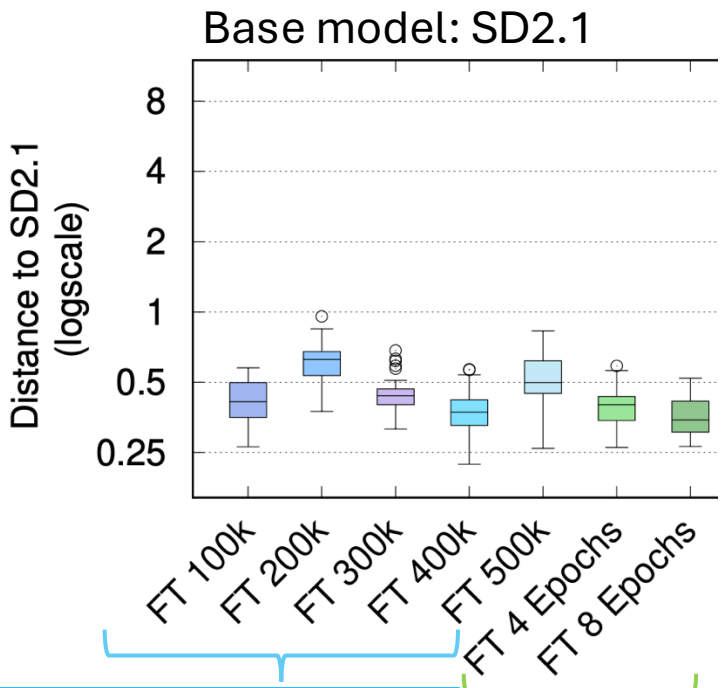
- Despite different fine-tuning configurations, fine-tuned models remain much closer to their true parent.



Detailed prompt: “sofa”
Fine-tuning dataset: PCB

Fine-tuned models stay closer to their base model

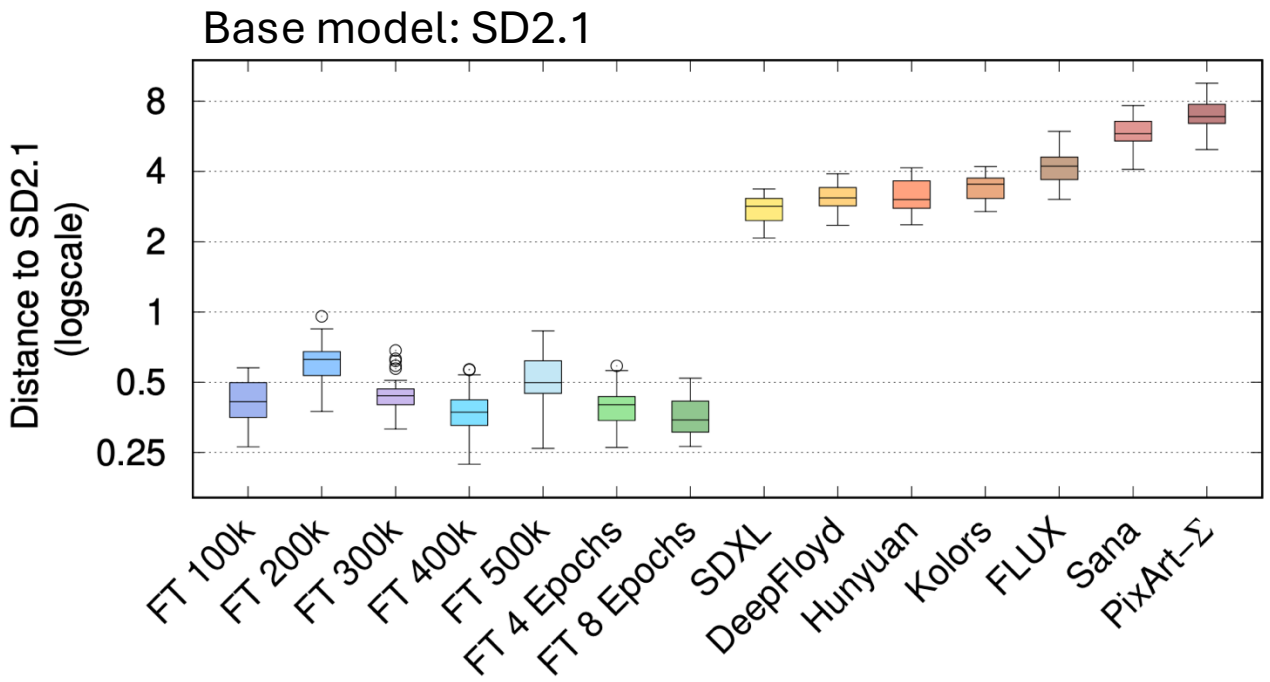
- Despite different fine-tuning configurations, fine-tuned models remain much closer to their true parent.



Detailed prompt: “sofa”
Fine-tuning dataset: PCB

Fine-tuned models stay closer to their base model

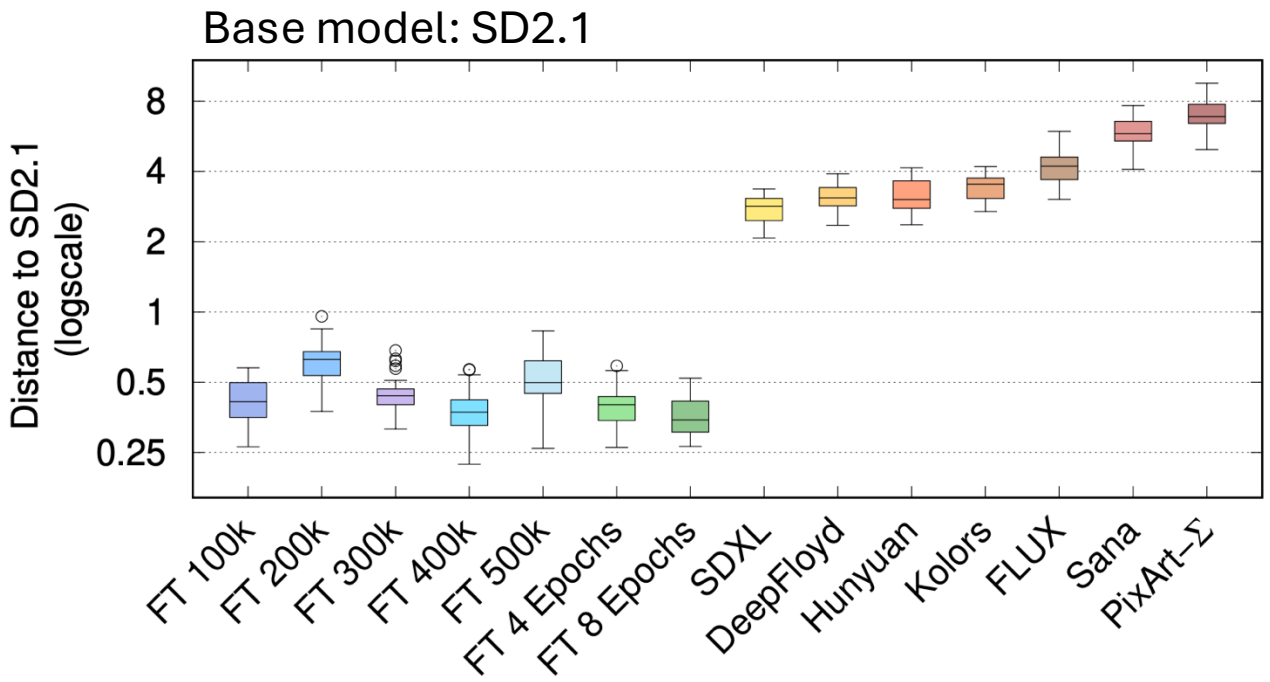
- Despite different fine-tuning configurations, fine-tuned models remain much closer to their true parent.

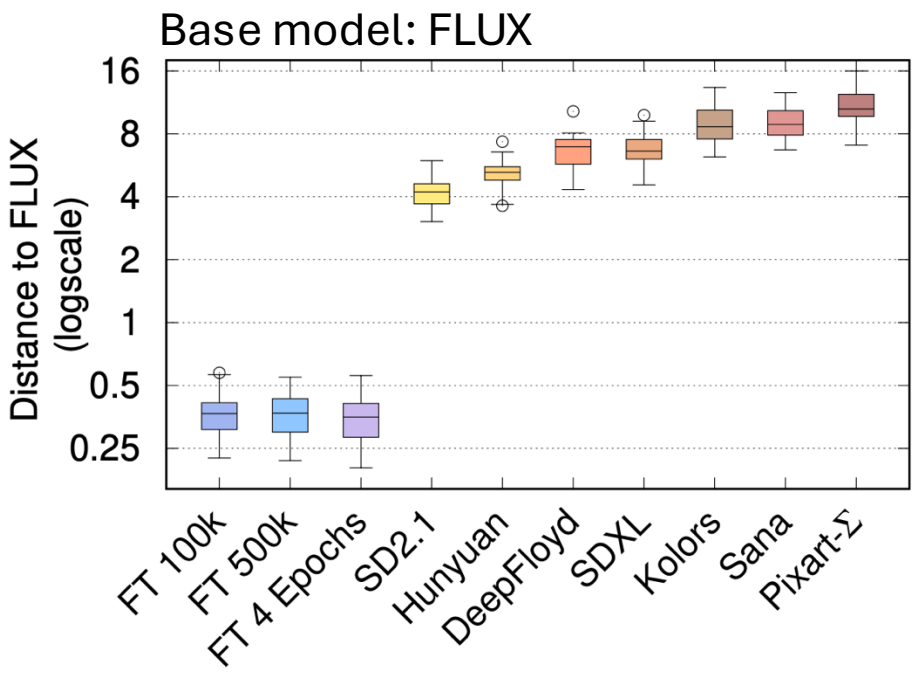


Detailed prompt: “sofa”
Fine-tuning dataset: PCB

Fine-tuned models stay closer to their base model

- Despite different fine-tuning configurations, fine-tuned models remain much closer to their true parent.





Detailed prompt: “sofa”
Fine-tuning dataset: PCB

Black-box provenance attribution is practical

- Fine-tuned models retain measurable traces of their parent model
- The signal remains detectable using only generated outputs

Black-box provenance attribution is practical

- Fine-tuned models retain measurable traces of their parent model
- The signal remains detectable using only generated outputs
- Robust to evaluated countermeasures, including weight perturbation and adaptive image attacks

Black-box provenance attribution is practical

- Fine-tuned models retain measurable traces of their parent model
- The signal remains detectable using only generated outputs
- Robust to evaluated countermeasures, including weight perturbation and adaptive image attacks
- Generalizes to in-the-wild models
- Used in a real-world legal investigation

Details in the poster!

Link to paper