What is model extraction?

Model extraction is a group of attacks where someone sends an AI normal questions and uses the answers to steal what it knows. There are three forms: building a copycat AI that behaves like yours (cloning), recovering pieces of the AI's inner workings (stealing your IP), and working out whether a specific person's data was used to train it (a privacy leak). It matters most when you train or fine-tune your own model, or when the training data was sensitive.

HOW IT WORKS

01 How are these attacks actually executed?

- Query-driven cloning. Send a set of inputs, label them with the target's answers, train a copycat. It works because the target's API acts as a free labeling machine.
- Active-learning cloning. Pick each next query to learn the most about the model's decision boundary. Cuts the number of queries needed by a lot.
- Reading the probabilities. When the API returns probabilities or top-k scores, every answer leaks more than a plain label would.
- Last-layer extraction. Recover a transformer's final layer with carefully chosen prompts (Carlini 2024). Shown against production APIs that returned full scores.
- Training-data extraction prompts. Feed the model prefixes it is likely to finish with memorized text. Works well on models trained on scraped web data.
- Membership inference. Compare how confident the model is on suspected training records versus fresh ones. A big confidence gap suggests the record was in the training set.

02 When does model extraction matter for your application?

Most when the model itself is your edge: private weights, an expensive training run, or behavior you cannot afford a rival to copy. It also matters when training-data privacy is a legal duty (HIPAA records, GDPR personal data). If you just wrap a third-party model with light fine-tuning, the extraction risk is smaller. There, the prompt

HOW TO DEFEND

- Return less. Give back labels, not full probability scores, when the app does not need them. This shuts the easiest extraction paths.
- Query limits plus monitoring. Extraction needs many queries. Rate limits, plus an alert when a new account fires a burst of varied queries, raise the cost.
- Add noise to outputs. Small, controlled noise hurts a copycat model more than it hurts real users. The trade-off depends on the task.
- Watermarking. Hide a detectable signal in outputs so a stolen model can be traced. It does not stop extraction, but it helps you prove theft later.
- Lock down access. If only logged-in, audited callers can reach the model, the attack surface shrinks sharply.
- Clean the training data. Remove duplicates and sensitive content from the training set, so even a successful extraction yields less.

SOURCES

[1] OWASP LLM10:2025, Unbounded Consumption (covers model theft)

[2] Carlini et al., Stealing Part of a Production Language Model

[3] Carlini et al., Extracting Training Data from Large Language Models

[4] Shokri et al., Membership Inference Attacks

template and the tools you wire in are usually the bigger targets.

Measure your model's extraction and inference exposure.
securelayer7.net/learn/ai-security/model-extraction

Open online