Department for
Science, Innovation
& Technology

Government
Digital Service

AI Insights

Model Distillation

Introduction

State-of-the-art large language models (LLMs) have grown to hundreds of billions of parameters, which are variables within a machine learning (ML) model that are updated through the training process. As a result of the huge number of parameters, these models are expensive to run outside specialised research environments. Inference consumes large amounts of energy, and the models may require multi-accelerator hardware setups. They can also introduce a several second time delay, which poses serious challenges to the responsiveness of real-time interactive or critical applications.

Model distillation, also known as knowledge distillation, presents a solution to these challenges by employing a teacher-student training paradigm. This approach aims to pass on the learned behaviour of a large, highly accurate model to a smaller, computationally efficient model that is better suited for practical deployment.

Knowledge distillation is a model compression technique where a smaller model (known as the student) learns to mimic the output of a larger, more complex model (the teacher). The goal is for the student model to achieve a level of performance close to the teacher, but with significantly fewer parameters, making it more efficient for deployment.

Knowledge distillation transfers the predictive behaviour of the large, accurate teacher language model to a much smaller student model, which is also referred to as the distilled model. During an offline training run, the student is shown the same prompts the teacher has seen and is optimised to mimic the teacher’s output probabilities, hidden-layer activations or even chain-of-thought explanations. In effect, the student watches the teacher's work and learns to reach the same conclusions with far fewer parameters. Doing so, it often retains 80% to 95% of the teacher’s task-specific quality while running on commodity graphic processing units (GPUs), commodity processing units (CPUs), and even mobile hardware.

In this situation, the heavy computation is confined to this one-off training phase so the resulting model artefact is lightweight, energy-efficient and easy to serve in production. An 8 billion parameter student typically responds in under 100 milliseconds on a single accelerator, whereas its 70 billion parameter teacher might require multiple GPUs and several seconds per request.

By focusing on model behaviour rather than the architecture, distillation preserves most of the teacher's generalisation knowledge capacity while dramatically shrinking parameter count, memory footprint, and inference latency.

Core distillation techniques

Distillation methods vary in their approach to transferring knowledge from teacher to student, with each technique suited to different application needs.

Teams commonly use four practical methods to balance model fidelity and computational cost to fit their particular use case. These are:

Method 1: response-based distillation
This method trains student models to reproduce the teacher's output probability distributions (which are known as logits or soft targets) instead of the discrete, hard ground-truth labels. This widely-used method can capture subtle class relationships and uncertainty patterns that are overlooked by discrete labels.

The logits contain rich information about inter-class relationships and uncertainties that are lost when using one-hot encoded ground truth labels.

This approach works best when the teacher's outputs have a wide distribution of probabilities, rather than high-confidence narrow probability outputs, as a broad distribution provides more information for the student to learn from.

Method 2: feature-based distillation
Instead of focusing on the final outputs, this method trains the student to reproduce the intermediate activations or hidden representations of the teacher model. This approach is beneficial when applications demand higher behavioural fidelity. By aligning the internal feature maps, attention patterns, or other intermediate states, feature-based distillation ensures that the student not only replicates the final prediction but also learns the underlying reasoning process.

This method can be especially effective in use cases where the hidden layers capture different levels of detail, such as with computer vision tasks where different layers might represent different levels of abstraction.

Method 3: self-distillation
This method involves the teacher model's later saved checkpoints supervising earlier saved checkpoints. This eliminates the need for a separate teacher model by using later-training checkpoints or deeper network layers of a trained model as mentors for earlier ones. This method streamlines iterative experimentation and can improve model performance without additional computational overhead, as no external model is required in the distillation process.

Method 4: chain-of-thought distillation
This method teaches the student model by exposing the teacher's reasoning steps. Chain-of-thought distillation has gained traction for reasoning-intensive tasks as student models learn from the rationale expressed in natural language by the teacher model, and absorb problem-solving and logical progression rather than simply replicating final answers.

This approach can be effective for mathematical reasoning, logical inference, and multi-step problem solving.

Key benefits of model distillation

Model distillation methods can introduce a number of key benefits regardless of your use case. These are:

Financial and environmental cost

It's possible to achieve significant cost reductions through knowledge distillation, with distilled models typically consuming 80% to 95% fewer compute resources than large frontier models. Energy consumption can be lower by an order of magnitude, which helps to meet sustainability mandates and carbon reduction targets while also reducing cooling infrastructure requirements.

Privacy and security

Distilled models can be kept on-premises, which supports security requirements by keeping sensitive data within a controlled infrastructure. It is also possible to distill from locally-hosted teacher models, which take advantage of advanced model capabilities while ensuring that classified information never leaves secure networks.

Ownership and resilience

Distilled models provide independence from external LLM providers, which reduces dependency risks and ensures service continuity during outages or vendor disruptions. Critical artificial intelligence (AI) services can be maintained using commodity hardware in distributed data centres to support disaster recovery requirements.

Flexibility

Model distillation enables smaller organisations to deploy sophisticated LLMs without excessive model budgets or infrastructure. A single distillation effort can produce specialised models for multiple use cases, from regulatory compliance to citizen communication.

Use cases for model distillation

Model distillation transforms broad teacher models into specialised students that excel at narrow domain-specific tasks, as shown in the following use cases.

Automated chat support

Automated chat support represents one of the most successful distillation applications. As an example, a company could distill models with state-of-the-art conversational abilities into much lower parameter models, such as Llama 2 7B, and still retain the ability to handle 80% of customer inquiries.

These distilled models can maintain consistent domain-specific expertise while serving thousands of simultaneous conversations at sub-second response times. This dramatically reduces support costs while improving customer satisfaction.

Real-time content filtering
In this use case, distilled models detect harmful content, spam, or policy violations at a scale required by social platforms. Teacher models provide a nuanced understanding of context and intent, while distilled students deliver the millisecond classifications needed for live content streams which protect users without introducing latency.

Regulatory document analysis
Here, distilled models specialised in regulatory-specific terminology can extract key information from contracts, compliance reports and regulatory filtering. These models understand domain-specific language patterns while processing sensitive documents entirely within secure infrastructure, which meets strict governance requirements.

Clinical decision support
In this example, medical knowledge is distilled into specialised models that assist healthcare providers with diagnosis suggestions and treatment recommendations. These models understand medical terminology and protocols while meeting security and privacy requirements through on-premise deployments, which enables AI-powered healthcare without compromising patient privacy.

The limitations of knowledge distillation

While knowledge distillation is a powerful tool for creating computationally efficient models, it comes with limitations such as:

Task breadth
The student's performance is inherently limited by the scope of the teacher's behaviour it encounters during training. When the deployment task involves diverse topics, discourse styles or reasoning formats, it becomes increasingly unlikely that the training data covers all the necessary edge cases it may experience operationally.

Domain transferability
Domain-specific characteristics can also limit a distilled model's utility. For example, a student distilled from a teacher specialised in biomedical language may inherit both domain-specific terminology and safety filters tuned to that field, making it well suited to hospital deployment on modest hardware. However, the same model is likely to perform poorly on tasks in unrelated fields, such as software engineering, because relevant abstractions and knowledge never appeared in its distillation corpus.

Shallow imitation

Attempts to imitate open-domain proprietary models illustrate the opposite challenge to domain transferability. Training on broad, domain-agnostic data often produces models that generate plausible-sounding responses across a range of topics, but lack the precision required for expert-level performance in any specific area.

Because of these limitations, the practical value of distillation is most significant in settings with narrow, well-defined vocabularies, stable task boundaries and precise compliance requirements. In contrast, broader-use assistants that aim to cover a wide range of topics still depend on the full scale and diversity of their teacher models.

Choosing your models

When selecting your models, there are various approaches to consider. These include the 3 main types of teacher models:

Type 1: proprietary teacher models
Examples such as GPT-4, Claude, and Gemini offer state-of-the-art performance through API access, and excel at generating high-quality rationales for complex tasks. However, they incur per-token costs and can raise data privacy concerns if prompts are processed externally.

Type 2: open-source teacher models
Examples such as Llama 3 and Mistral provide greater control and privacy by running on your infrastructure. They offer unrestricted access to internal representations, which enables more sophisticated distillation techniques for organisations with strict data governance requirements.

Type 3: hybrid approaches
This option balances quality with cost by using proprietary models for initial knowledge extraction and open-source models for iterative refinement.

For selecting a student model, parameter count is a primary driver. Models in the 1 billion to 7 billion range currently offer the best trade-off between performance and efficiency. Sub-billion models may suffice for simple tasks, while models exceeding 13 billion tend to undermine the goal of distillation by reintroducing high inference costs.

Popular student model families currently include Llama 2 7B for general tasks, Mistral 7B for instruction following, and Phi-3 Mini for resource-constrained environments. These provide a strong baseline performance that accelerates fine-tuning.

You should also consider deployment constraints. Single GPU deployments favour models with 7 billion or below, while CPU-only interfaces require sub-billion parameters, and mobile deployments need highly-optimised architectures.

Implementation guide

Here are some of the key steps involved with distilling knowledge from a teacher to a student model:

Phase 1: knowledge extraction from teacher model

1. Define your domain boundaries by establishing a clear capability statement that captures the specific domain and behaviours you need to inherit. Whether it's medical triage, contract classification, or step-by-step reasoning, clear boundaries ensure focused knowledge extraction.

2. Build your seed dataset by assembling a large number of high quality prompts that mirror real user data. For this, we recommend between 2,000 to 5,000 prompts. This seed corpus should cover the domain's major sub-tasks and include diverse prompt styles, complexity levels, and edge cases that represent actual usage patterns.

3. Extract teacher knowledge by running your seed corpus through your chosen teacher model using chain-of-thought prompts that elicit both answers and reasoning. Capture every available signal, including final outputs, reasoning traces, confidence scores, and hidden states when accessible.

4. Finally, review a random sample of around 10% of these examples to verify the quality, tone, and factual accuracy before scaling up.

Phase 2: knowledge transfer to student model

1. Begin data preparation and formatting by converting the teacher outputs into a clean format, with one field for prompts and one for targets. Create a validation split of 10% to 20%, and tokenise everything with the student model's vocabulary to ensure compatibility across different training environments.

2. Start training and monitoring by loading the chosen student model and configuring its hyperparameters with a modest learning rate and early stopping on validation loss. Using modern training frameworks such as HuggingFace will handle most technical details automatically so that you can focus on monitoring and evaluation assessments.

3. Implement the iterative refinement process by deploying the trained student model and testing it on new examples to identify failure modes. Collect additional teacher examples which target weak areas, then retrain the student with this augmented dataset. Plan for 2 or 3 cycles to close remaining accuracy gaps

while maintaining the model's compact size.

By following these steps, your resulting specialised model will typically achieve between 80% to 95% of teacher performance while being between 5 to 10 times faster and significantly more cost-effective for production deployment.

Conclusion

Knowledge distillation offers a pragmatic route to achieving sovereign and sustainable AI in the public sector. It delivers near-frontier language capabilities on existing hardware, significantly reduces inference costs and carbon emissions, and ensures that sensitive data remains within secure boundaries.

Knowledge distillation does add an additional step to the machine learning workflow, as you have to train the student model and update it as policies or datasets change. Rigorous bias and quality testing are also essential to prevent the compressed model from amplifying the teacher’s flaws.

However, when the use case starts with a narrow, well-defined task and results are carefully measured, the savings usually outweigh these caveats within a few weeks of production use.