

Welcome to Aura

Aura is your AI Energy Efficiency Companion. It helps you understand and optimize the energy consumption of your AI models. Use the navigation menu or click the cards below to explore features like Energy Prediction, Model Optimization, Training Simulations, and more.

Energy Predictor

Estimate model energy usage

Input your model's architecture, framework, and data size to get an AI-powered prediction of its energy footprint.

Model Optimizer

Get tips to shrink your models

Describe your model and receive AI suggestions like pruning and quantization to make it smaller and more efficient.

- Home
- Energy Predictor
- Model Optimizer
- Aura Chat
- Saving Tips
- Model Comparison
- Saved
- Report Generator
- Training Simulation
- Settings

Training Simulation

Model & Pruning Details

Model: MobileNetV2 (pretrained on ImageNet) with its final classifier layer adjusted to 10 output classes. **Device:** CUDA if available, else CPU.
Pruning: Simple unstructured L1 pruning applied with amount 0.3 to Conv2d and Linear layers before training.

Simulated Training Process

Simulated training for 5 epochs.
Optimizer: Adam (lr=0.001).
Loss Function: CrossEntropyLoss.
Data: Simulated using torchvision.datasets.FakeData with Resize and ToTensor transforms.
Batch Size: 32.
Mixed Precision (AMP): Enabled if CUDA is available, using GradScaler.
Device for Autocast: CUDA if available, else CPU.

Epoch Energy Logs

Epoch 1: Estimated Energy Usage - 0.50 kWh
Epoch 2: Estimated Energy Usage - 1.00 kWh
Epoch 3: Estimated Energy Usage - 1.50 kWh
Epoch 4: Estimated Energy Usage - 2.00 kWh
Epoch 5: Estimated Energy Usage - 2.50 kWh

Training Simulation

Energy Consumption visualizations

Bar Chart

Line Chart

Histogram

Pie Chart

Energy per Epoch



Final Message

✓ Optimized model saved as 'optimized_model.pth'

Model Training Simulation

Simulate a PyTorch model training process including pruning and energy logging. This demonstrates conceptual steps and does not perform actual GPU/CPU intensive training.

Simulation Overview

This tool simulates key aspects of an energy-aware model training pipeline: loading a pre-trained model (MobileNetV2), applying pruning, running a mock training loop with Automatic Mixed Precision (AMP) if CUDA were present, and logging estimated energy usage. The actual training calculations and GPU operations are not performed.

▶ Start Simulation

- Isolation Forest for anomaly detection

Performance Metrics

Parameter	Before AI	After AI	Improvement
Average Energy Consumption	500 kWh/month	325 kWh/month	35%
Peak Load	80 kW	49.6 kW	38%
Prediction Accuracy	N/A	92.3%	—
CO ₂ Emissions	25 tons/year	15.38 tons/year	38.5%

Conclusion

The AI-based energy efficiency system significantly outperforms traditional manual optimization by providing data-driven, automated, and adaptive strategies. The model can be scaled to buildings, data centers, or industrial setups.

AI Energy Efficiency Report Generator

Enter your system's 'Before Optimization' values to generate a report.

Average Energy Consumption (kWh/month)

500

Peak Load (kW)

80

CO₂ Emissions (tons/year)

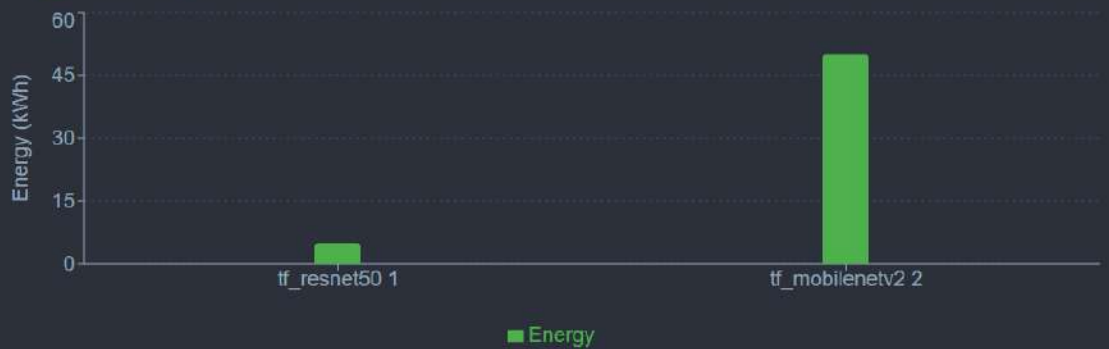
25



Generate Report

Model Comparison


Energy Consumption Chart



Note: If energy units differ between models, the Y-axis shows the unit of the first model. Tooltips display individual units.

 Save Report to App

 Export Report (CSV)

 Download Chart (PNG)

Comparison Results

Side-by-side energy prediction and key differences for 2 models.

tf_resnet50 1

Framework: TENSORFLOW
Base Model: tf_resnet50
Architecture: CNN
Data Size: 10GB

Predicted Energy: 5 kWh
Confidence: low

tf_mobilenetv2 2

Framework: PYTORCH
Base Model: tf_mobilenetv2
Architecture: Transformer
Data Size: 10GB

Predicted Energy: 50 kWh
Confidence: low

Energy Consumption Chart

Model Comparison

Model 1

Framework

TensorFlow

Optional: Base Model

ResNet50

May pre-fill architecture.

Architecture

CNN

Data Size

10GB

Model 2

Framework

PyTorch

Optional: Base Model

MobileNetV2

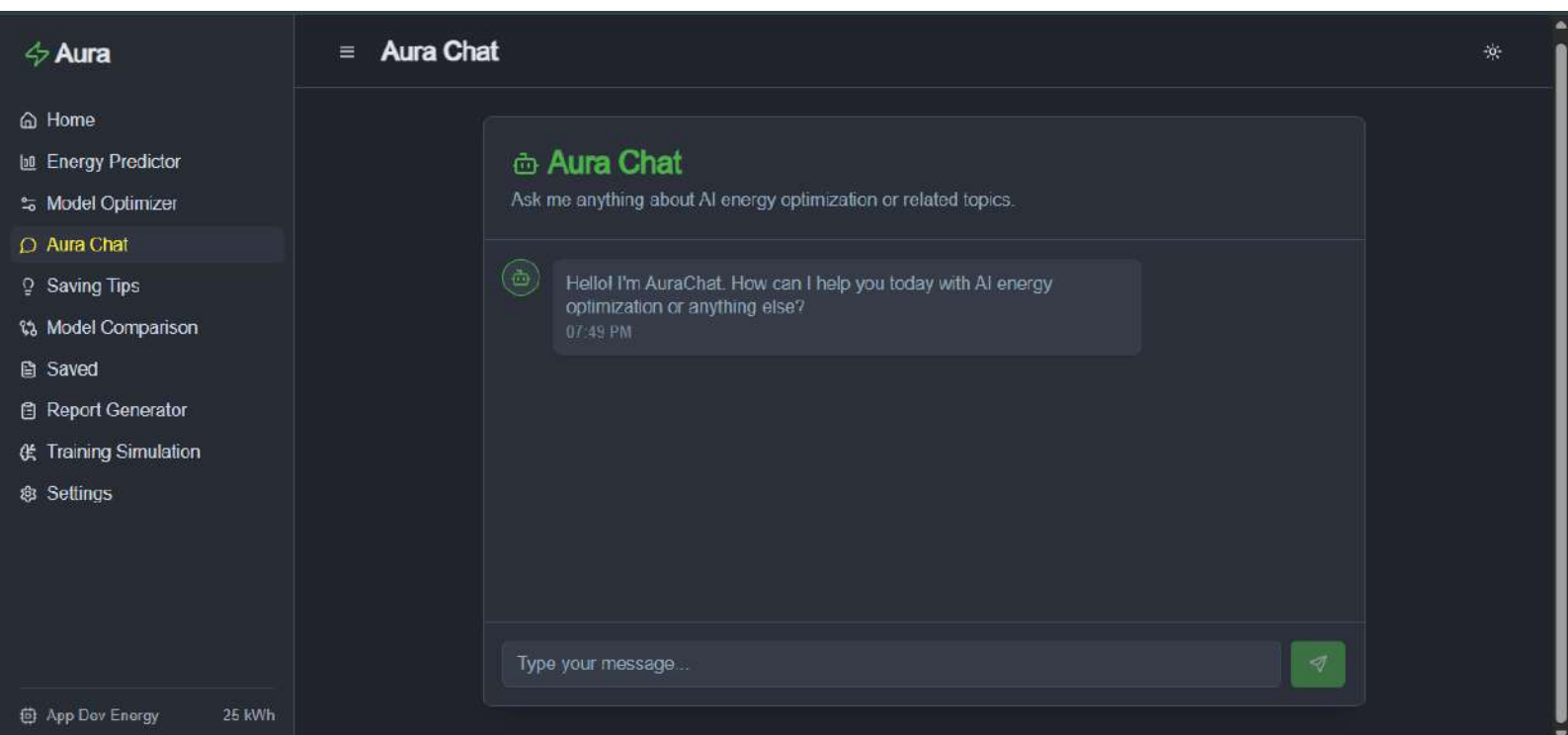
May pre-fill architecture.

Architecture

Transformer

Data Size

10GB



Get Optimization Suggestions

💡 Optimization Suggestions

AI-powered recommendations to improve your model's efficiency.

🕒 Suggested Techniques

↗️ Expected Benefits

Applying these techniques can significantly reduce the model size (potentially by 2x to 10x or more, depending on the aggressiveness of the optimization) and improve inference speed, especially on resource-constrained devices. Quantization may introduce a small drop in accuracy (e.g., 1-3%), but this can often be mitigated through quantization-aware training. Pruning can also lead to accuracy loss if not done carefully; iterative pruning with fine-tuning is recommended. Knowledge distillation can even improve accuracy in some cases by transferring knowledge from a larger, more accurate model to a smaller one.

Optimization Suggestions

AI-powered recommendations to improve your model's efficiency.

Suggested Techniques

- Pruning:** Reduce the number of connections in the network by removing weights with small magnitudes. Implement magnitude-based pruning and iterative pruning with fine-tuning.
- Quantization:** Convert the model's weights and activations from floating-point numbers (e.g., float32) to lower-precision integers (e.g., int8). Use post-training quantization or quantization-aware training.
- Knowledge Distillation:** Train a smaller "student" model to mimic the behavior of the larger ResNet50 "teacher" model.
- Weight Clustering:** Group similar weights together and represent them with a single value, reducing the number of unique weights stored.
- Layer Fusion:** Combine multiple convolutional layers, batch normalization layers, and activation functions into a single layer, reducing the number of operations and memory transfers.
- Depthwise Separable Convolutions:** Replace standard convolutional layers with depthwise separable convolutions to reduce the number of parameters. Consider replacing some standard convolutional layers with depthwise separable ones, especially in the later layers of the network.

Model Optimizer

Describe your AI model to receive optimization suggestions.

Select AI Framework

TensorFlow

Select the primary framework. This helps tailor suggestions and may pre-fill parts of the description.

Optionally, select a base model

ResNet50

Selecting a model may pre-fill parts of the description and suggest a framework.

Model Description

Framework: Tensorflow
Model: tf_resnet50
Architecture: CNN (ResNet-like)
Dataset: [Specify dataset]
Task: Image Classification
This is a Convolutional Neural Network commonly used for image classification.

Optimization Suggestions Received
AI has provided model optimization tips.

Energy Predictor

Input Data Size

100GB

Specify the size of the data the model processes.

Predict Energy Consumption

Prediction Results

Estimated energy consumption for: tf_resnet50

Predicted Consumption

50 kWh

Confidence: low

Save Prediction to Reports

Aura

Home

Energy Predictor

Model Optimizer

Aura Chat

Saving Tips

Model Comparison

Saved

Report Generator

Training Simulation

Settings

App Dev Energy

25 kWh

Energy Predictor

Energy Predictor

Enter AI model details to predict its energy consumption.

Select AI Framework

TensorFlow

Select the primary framework used for the model.

Optionally, select a base model

ResNet50

Selecting a model may pre-fill architecture and suggest a framework.

Model Architecture

CNN (ResNet-like)

Describe the type of architecture. If you selected a framework, enter the architecture details here, or in the name.

Input Data Size

Prediction Successful

Energy consumption predicted.

