AWS Certified Machine Learning Engineer - Associate Exam Guide

# Section 2.2 — Train and Refine Models (In-Depth)

This section focuses on training machine learning models efficiently and refining them for better performance. It covers training strategies, performance optimization, hyperparameter tuning, and AWS services for automation.

## 1. Training Fundamentals

• Training process: epochs, batch size, learning rate, optimizers (SGD, Adam)

• Data splits: training, validation, test

• Early stopping: terminate training when validation metric stops improving

• Distributed training: multi-GPU/multi-node for large datasets

## 2. Performance Optimization

• Regularization: dropout, weight decay, L1/L2 penalties

• Data augmentation: generate synthetic variations (images, text) to improve generalization

• Model complexity trade-offs: deeper networks vs overfitting risk

• Prevent catastrophic forgetting: careful fine-tuning of pre-trained models

## 3. Hyperparameter Tuning

• Hyperparameters (set before training): learning rate, max depth, # of layers, regularization coefficients

• Methods: grid search (exhaustive), random search (efficient), Bayesian optimization (smart exploration)

• Objective metric: define clear validation metric (accuracy, RMSE, etc.)

## 4. AWS Services

• Amazon SageMaker Training Jobs → scalable managed training service

• SageMaker Script Mode → run PyTorch/TensorFlow scripts directly

• SageMaker Distributed Training → scale to multi-GPU and multi-node

• SageMaker Debugger → monitor training jobs, detect overfitting/underfitting, analyze gradients

• SageMaker Automatic Model Tuning (HPO) → runs parallel jobs with different hyperparameters

• SageMaker Autopilot → automatically trains and tunes baseline models

• Amazon EC2 with Deep Learning AMIs → alternative for custom training

## 5. Refinement Techniques

• Fine-tuning pre-trained models (JumpStart, Bedrock foundation models)

• Ensembling: bagging, boosting, stacking to improve performance

• Model compression/pruning: reduce model size for cost efficiency

• Feature selection: remove irrelevant/noisy features

## ✅ Exam Tips

• Autopilot = quick baseline models, AMT = systematic hyperparameter tuning

• Debugger detects vanishing gradients, overfitting, divergence issues

• Distributed training is critical for very large datasets/models

• Fine-tuning pre-trained models saves time/cost compared to training from scratch

• Always define validation metrics before tuning