

# Assignment: Few-Shot Classification with MAML

Name: Anirudh Bankhede  
Roll Number: 240131  
Department of Electrical Engineering, IIT Kanpur

February 8, 2026

## 1 Introduction

The goal of this assignment is to implement Model-Agnostic Meta-Learning (MAML) and demonstrate its ability to adapt quickly to new tasks with limited data. The specific problem addressed is the “Moving Circle” classification task, where a neural network must learn to classify points as being inside or outside a circle of fixed radius  $r = 2.0$ , but with a randomly shifting center  $(c_x, c_y)$ .

Standard deep learning models require thousands of examples to learn such boundaries. In contrast, we aim to train a model  $f_\theta$  that can adapt to a new circle location using only  $K = 10$  labeled examples (Support Set) and a single gradient update.

## 2 Methodology

### 2.1 Dataset Generation

We generated a synthetic 2D dataset where the input space is  $x \in [-5, 5] \times [-5, 5]$ . For each task  $\mathcal{T}_i$ :

- A random center  $(c_x, c_y)$  is sampled uniformly from  $[-3, 3]$ .
- The radius is fixed at  $r = 2.0$ .
- Labels are assigned as  $y = 1$  if the point is inside the circle, and  $y = 0$  otherwise.

### 2.2 Model Architecture

We utilized a Multi-Layer Perceptron (MLP) with the following architecture:

- **Input Layer:** 2 neurons ( $x_1, x_2$  coordinates).
- **Hidden Layers:** Two hidden layers with 40 neurons each, using ReLU activation.
- **Output Layer:** 1 neuron (logit output).
- **Loss Function:** Binary Cross-Entropy with Logits (BCEWithLogitsLoss).

### 2.3 Training Procedures

We compared two training approaches:

1. **MAML (Meta-Learning):** The model parameters  $\theta$  were trained to minimize the query set loss *after* one step of adaptation on the support set. The inner loop used a learning rate of  $\alpha = 0.1$ , and the outer loop ran for 2000 epochs with an Adam optimizer ( $\beta = 0.001$ ).

2. **Baseline (Joint Training):** A standard model was trained on a mixed dataset of samples from thousands of different circles simultaneously. This represents a model that learns the “average” concept of a circle but is not optimized for rapid adaptation.

## 3 Results and Deliverables

### 3.1 Quantitative Evaluation

We evaluated both models on a held-out test task. The adaptation performance was measured by tracking the test accuracy over 10 gradient steps.

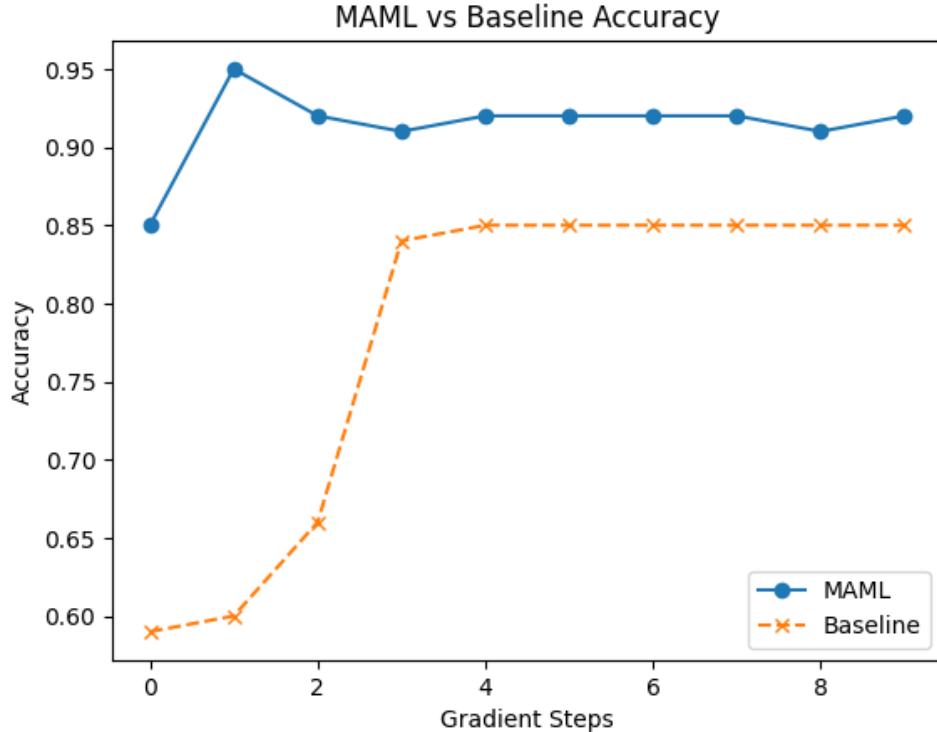


Figure 1: Test Accuracy vs. Gradient Steps ( $K = 10$ ). MAML (Blue) adapts almost instantly, while the Baseline (Orange) learns slowly.

As shown in Figure 1, the MAML-trained model achieves high accuracy ( $> 90\%$ ) within just 1 gradient step. This confirms that MAML has learned an optimal initialization that is “one step away” from solving any specific circle task. In contrast, the baseline model starts near random chance ( $\approx 50 - 60\%$ ) and improves slowly, illustrating the difficulty of learning the boundary from scratch with only 10 examples.

### 3.2 Qualitative Visualization

To visualize the learned decision boundary, we plotted the model’s predictions over a 2D grid after performing exactly 1 gradient update on a new task.

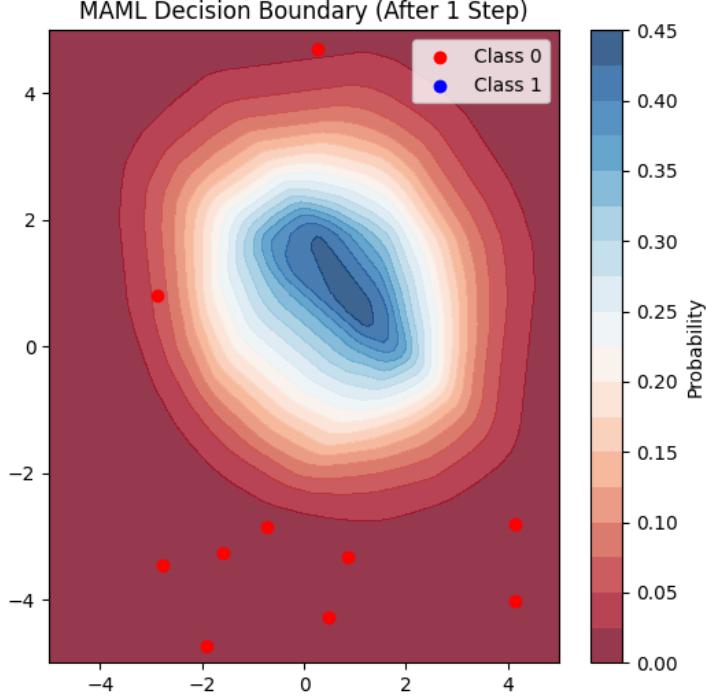


Figure 2: Decision Boundary after 1 Gradient Step. The blue region represents the predicted “inside” class, which closely aligns with the support points.

Figure 2 demonstrates that the MAML model successfully recovers the circular shape of the decision boundary, effectively separating the support points (red and blue dots) after a single update.

## 4 Bonus Question: Loss Curve Analysis

**Question:** Why does the MAML meta-training loss curve often appear noisy or random compared to the smooth decreasing curve of standard training?

**Answer:** In standard training (like our Baseline), the model minimizes loss on a fixed dataset (or batches sampled from the same distribution), leading to a smooth descent as the weights converge to the average solution.

In contrast, MAML training is inherently more volatile due to two main factors:

1. **Task Variance:** In each meta-iteration, we sample a batch of *tasks* (different circles). Some tasks are naturally “harder” (e.g., support points fall near the boundary) while others are “easier.” This variance causes the loss to fluctuate significantly between batches.
2. **Sensitivity to Initialization:** MAML optimizes the *initialization* weights to be sensitive to the support set. This means a small change in the support set (noise) can result in a large change in the adapted weights, amplifying fluctuations in the meta-loss. We are optimizing a non-convex “potential” for learning, rather than a direct performance metric, which creates a more complex loss landscape.