

---

# CHAMELEON: Curriculum Heuristic Adaptive Multi-agent Exploration Learning with Online Negotiation Supplementary Materials

---

## 1 Hyperparameter Details for CIFAR-10 Curriculum Learning

This section provides a comprehensive overview of the hyperparameters used in the CIFAR-10 curriculum learning implementation, ensuring transparency and reproducibility

### 1.1 Training Parameters

- **batch-size:** 128. The number of images processed in each training iteration. This influences training speed and resource usage.
- **epochs:** 200. The total number of passes through the entire training dataset.
- **lr:** 0.001. The learning rate determines the step size taken during optimization, impacting convergence speed and stability.
- **max-budget:** 25000. The maximum number of training samples used in the curriculum. This controls the total data utilized for learning.
- **initial-sample-size:** 1000. The initial number of samples used to start the curriculum. This sets the starting point for data increment.
- **samples-per-epoch:** 1000. The number of new samples added to the training set in each epoch. This governs the rate of curriculum progression.
- **print-freq:** 10. The frequency of printing training progress during each epoch.
- **seed:** None (default). A seed value for random number generation, ensuring reproducibility. If None, results may vary slightly between runs.
- **save-interval:** 10. The frequency of saving intermediate training results (model checkpoints, metrics, etc.).

### 1.2 Model Parameters

- **model:** ResNet18 pretrained on ImageNet1K\_V1. A popular convolutional neural network architecture used for image classification. Pretraining provides initial knowledge for faster convergence.
- **optimizer:** Adam. An adaptive optimization algorithm that adjusts learning rates for each parameter.

### 1.3 Curriculum Learning Parameters

- **strategies:** A dictionary defining the different sample selection strategies:
  - **S\_U:** Uncertainty sampling. Selects samples the model is most uncertain about.
  - **S\_C:** Class balancing. Selects samples to maintain a balanced class distribution.
  - **S\_B:** Boundary sampling (previously *get\_difficult\_samples*). Selects samples near the decision boundaries.
  - **S\_D:** Diversity sampling. Select samples that are diverse in feature space.
- **meta\_agent:** Manages the weights assigned to each strategy during curriculum learning.
  - **learning\_rate:** 0.1. The learning rate for adjusting strategy weights based on performance.

## 2 Hyperparameter Details for CIFAR-100 Curriculum Learning

This section provides a comprehensive overview of the hyperparameters used in the CIFAR-100 curriculum learning implementation, ensuring transparency and reproducibility (R1, R2, R3).

### 2.1 Training Parameters

- **batch-size:** 128. The number of images processed in each training iteration. This influences training speed and resource usage.
- **epochs:** 200. The total number of passes through the entire training dataset.
- **lr:** 0.001. The learning rate determines the step size taken during optimization, impacting convergence speed and stability.
- **max-budget:** 25000. The maximum number of training samples used in the curriculum. This controls the total data utilized for learning.
- **initial-sample-size:** 1000. The initial number of samples used to start the curriculum. This sets the starting point for data increment.
- **samples-per-epoch:** 1000. The number of new samples added to the training set in each epoch. This governs the rate of curriculum progression.
- **workers:** 4. The number of worker processes used for data loading. This affects data loading speed and efficiency.
- **print-freq:** 10. The frequency of printing training progress during each epoch.
- **seed:** None (default). A seed value for random number generation, ensuring reproducibility. If None, results may vary slightly between runs.
- **save-interval:** 10. The frequency of saving intermediate training results (model checkpoints, metrics, etc.).

### 2.2 Model Parameters

- **model:** ResNet18 pretrained on ImageNet1K\_V1. A popular convolutional neural network architecture used for image classification. Pre-training provides initial knowledge for faster convergence. The final fully connected layer is modified to output 100 classes (for CIFAR-100).
- **optimizer:** Adam. An adaptive optimization algorithm that adjusts the learning rates for each parameter.

---

## 2.3 Curriculum Learning Parameters

- **strategies:** A dictionary defining the different sample selection strategies:
  - **S\_U:** Uncertainty sampling. Selects samples the model is most uncertain about.
  - **S\_C:** Class balancing. Selects samples to maintain a balanced class distribution.
  - **S\_B:** Boundary sampling (previously *get\_difficult\_samples*). Selects samples near the decision boundaries.
  - **S\_D:** Diversity sampling. Selects samples that are diverse in feature space.
- **meta\_agent:** Manages the weights assigned to each strategy during curriculum learning.
  - **learning\_rate:** 0.1. The learning rate for adjusting strategy weights based on performance.

This detailed description of hyperparameters aims to provide complete transparency and enable others to reproduce and build upon this CIFAR-100 curriculum learning framework.

## 3 Hyperparameter Details for SVHN and FMNIST Curriculum Learning

This section provides a comprehensive overview of the hyperparameters used in the SVHN and FMNIST curriculum learning implementations, ensuring transparency and reproducibility (R1, R2, R3).

### 3.1 Training Parameters

These parameters are common to both SVHN and FMNIST:

- **batch-size:** 128. The number of images processed in each training iteration. This influences training speed and resource usage.
- **epochs:** 200. The total number of passes through the entire training dataset.
- **lr:** 0.001. The learning rate determines the step size taken during optimization, impacting convergence speed and stability.
- **max-budget:** 25000 (adjustable). The maximum number of training samples used in the curriculum. This controls the total data utilized for learning.
- **initial-sample-size:** 1000. The initial number of samples used to start the curriculum. This sets the starting point for data increment.
- **samples-per-epoch:** 1000. The number of new samples added to the training set in each epoch. This governs the rate of curriculum progression.
- **workers:** 4. The number of worker processes used for data loading. This affects data loading speed and efficiency.
- **print-freq:** 10. The frequency of printing training progress during each epoch.
- **seed:** None (default). A seed value for random number generation, ensuring reproducibility. If None, results may vary slightly between runs.
- **save-interval:** 10. The frequency of saving intermediate training results (model checkpoints, metrics, etc.).

### 3.2 Model Parameters

- **model:** ResNet18 pretrained on ImageNet1K-V1. A popular convolutional neural network architecture used for image classification. Pretraining provides initial knowledge for faster convergence. The final fully connected layer is modified to output the appropriate number of classes for the dataset (10 for SVHN, 10 for FMNIST).
- **optimizer:** Adam. An adaptive optimization algorithm that adjusts the learning rates for each parameter.

### 3.3 Curriculum Learning Parameters

- **strategies:** A dictionary defining the different sample selection strategies:
  - **S\_U:** Uncertainty sampling. Selects samples the model is most uncertain about.
  - **S\_C:** Class balancing. Selects samples to maintain a balanced class distribution.
  - **S\_B:** Boundary sampling (previously *get\_difficult\_samples*). Selects samples near the decision boundaries.
  - **S\_D:** Diversity sampling. Select samples that are diverse in feature space.
- **meta\_agent:** Manages the weights assigned to each strategy during curriculum learning.
  - **learning\_rate:** 0.1. The learning rate for adjusting strategy weights based on performance.

## 4 Prompt Design

Prompt design plays a crucial role in guiding the selection agents and the moderator within the curriculum learning framework. Carefully crafted prompts ensure that the agents understand their roles, the context of the learning process, and the factors to consider when making decisions.

### 4.1 Selection Agent Prompts

Each selection agent receives a structured prompt as follows:

```
As the {self.strategy} agent for CIFAR-10/100/SVHN/FMNIST image classification:
Current model accuracy: {model_performance:.2f}
Current curriculum stage: {curriculum_stage:.2f}
Class distribution: {class_distribution}
Propose how to select the next batch of samples and why. Consider:
1. The strengths of your strategy ({self.strategy})
2. The current model performance and curriculum stage
3. The current class distribution
Respond in 50 words or less.
```

#### Prompt Elements:

\* **Agent Role:** Clearly defines the agent’s strategy (e.g., uncertainty, class balance, boundary, diversity). \* **Performance Metrics:** Provides the current model accuracy and curriculum stage, giving the agent context about the learning progress. \* **Class Distribution:** Informs the agent about the current representation of classes in the training data. \* **Guiding Instructions:** Encourages the agent to consider its strategy’s strengths, the learning context, and the class distribution when proposing a sample selection approach. \* **Response Length Constraint:** Limits the response to 50 words, promoting conciseness and focus.

### 4.2 Moderator Agent Prompt

The moderator agent receives a prompt with the following structure:

```
As the moderator, review the proposals from different selection agents:
{proposals}
Current model accuracy: {model_performance:.2f}
Current curriculum stage: {curriculum_stage:.2f}
Current strategy weights: {current_weights_str}
Suggest adjustments to the current weights based on the proposals and current state.
The weights should sum to 1.
Respond with a Python dictionary: {"S_U": float, "S_C": float, "S_B": float, "S_D": float}
Provide a brief explanation for your decision.
```

---

### Prompt Elements:

\* **Moderator Role:** Clearly establishes the agent’s role as the moderator. \* **Agent Proposals:** Presents the proposals from all selection agents, allowing the moderator to compare and evaluate them. \* **Performance Metrics:** Provides the current model accuracy and curriculum stage. \* **Current Weights:** Shows the current weights assigned to each strategy. \* **Guiding Instructions:** Directs the moderator to analyze the proposals and the current learning state to suggest weight adjustments. \* **Output Format:** Specifies the desired output format (a Python dictionary with strategy weights), facilitating parsing and integration with the curriculum learning algorithm. \* **Explanation Request:** Prompts the moderator to justify the suggested weight adjustments.

### 4.3 Example Agent and Moderator Responses

Here’s an example of the interaction between the selection agents and the moderator during a training run:

#### Agent Proposals:

- **S\_U:** I recommend selecting samples with high prediction uncertainty, focusing on classes with lower representation in the current batch. This strategy leverages uncertainty to improve model accuracy and ensures balanced learning across classes, aligning with the curriculum’s early stage.
- **S\_C:** I would select samples from classes 0, 1, and 8 to balance representation, since they have lower counts. This will help improve model generalization. Considering the early curriculum stage and model accuracy, focusing on underrepresented classes can enhance learning diversity and mitigate bias.
- **S\_B:** I recommend selecting samples from classes with lower representation (e.g., classes 0, 7, 5, 9, 1, 8) to address class imbalance and challenge the model. This helps explore diverse features and improve generalization, enhancing the current model’s accuracy and progressing the curriculum effectively.
- **S\_D:** I would select samples from under-represented classes (e.g., classes 0, 7, 5) to increase diversity and balance in the training data. This will help improve model generalization and performance by addressing class imbalances and enriching the curriculum with varied examples.

**Moderator Suggested Weights:** {'S\_U': 0.1, 'S\_C': 0.15, 'S\_B': 0.2, 'S\_D': 0.55}

**Moderator Explanation:** I suggest adjusting the weights to increase the emphasis on selecting samples from classes with lower representation (proposed by S\_B and S\_D agents). By allocating more weight to these strategies, we can further address class imbalances, enhance model generalization, and introduce diversity in the training data. Balancing representation across classes like 0, 7, 5, 9, 1, and 8 will help challenge the model and improve its accuracy, aligning well with the curriculum’s early stage objectives.

**Calculated Rewards:** {'S\_U': 0.0, 'S\_C': 0.14003972681420315, 'S\_B': 0.02348218168340101, 'S\_D': 1.0}

These prompt designs aim to effectively guide the agents in their decision-making processes, promoting meaningful interactions and informed choices within the curriculum learning framework.

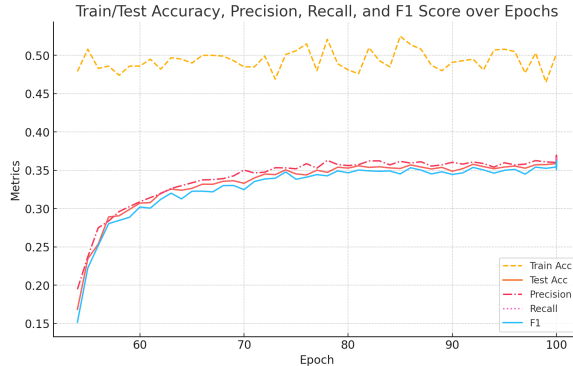


Figure 1: Training and Test Loss and Accuracy across 100 epochs.

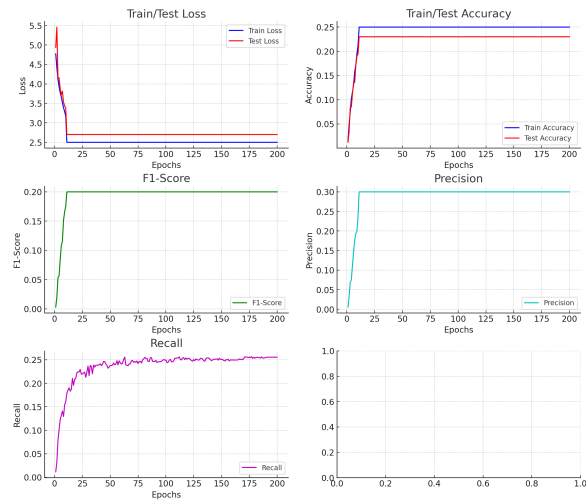


Figure 2: better to 3rd line of images as column and show the figures in full page F1-score, Precision, and Recall across 200 epochs.