NICE with Knowledge Distillation

A20586593 Seonghwan Lim

1. Project Information

1.1 Purpose

The goal is to enhance the accuracy of the neurogenesis-inspired NICE method in class incremental learning by using knowledge distillation techniques.

1.2. Selected Paper

Gurbuz, M. B., Moorman, J. M., & Dovrolis, C. (2024). NICE: Neurogenesis Inspired Contextual Encoding for Replay-free Class Incremental Learning. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 23659–23669.

NICE proposes a method for enabling class incremental learning without the need for replay, effectively retaining previously learned knowledge while learning new classes. To enhance the model's flexibility, it introduces the concept of neurogenesis, designing a dynamic network adjustment mechanism based on the age and importance of neurons. By categorizing neurons by their age, older neurons are frozen to preserve memory, while younger neurons are allocated for new learning, thereby reducing interference between tasks. Additionally, NICE demonstrates the ability to predict specific episode data effectively and suppress irrelevant outputs through its context detection mechanism. This approach has been experimentally shown to effectively address the problem of catastrophic forgetting, even without relying on traditional replay-based methods.

1.3. Code Modifications

train_eval.py

Line 63: Added distillation_loss method to implement knowledge distillation.

Line 78: Integrated distillation_loss into the phase_training_ce method for teacher-student learning.

learner.py

Line 43: Updated start_episode method to load the teacher model from the earlier episode.

Line 53: Modified end_episode method to save the current episode model as the teacher for the next episode.

architecture.py

Line 72: Enhanced forward method to copy activations from each layer and store them in the activations.

2. Problem Definition

2.1. Description of the Problem to Be Solved

In class incremental learning, catastrophic forgetting leads to the loss of previously learned knowledge as the model learns new classes, unless specific measures are implemented. NICE addresses this issue by drawing inspiration from neurogenesis to keep prior knowledge. However, despite its effectiveness, NICE has several limitations that impact its adaptability and performance. First, once neurons reach a frozen state (age 2 or older), they are no longer updated, which limits the model's ability to adapt to new information. This rigidity can be problematic in scenarios where class distributions overlap or evolve over time. Second, during the transition of neurons from young to mature states, some contextual information may be partially lost, especially for challenging tasks where preserving fine-grained representations is critical. Lastly, NICE relies heavily on activation-based selection for young neurons, which does not explicitly incorporate knowledge of inter-class relationships or previously learned distributions. This can lead to suboptimal use of the network's capacity, particularly in the early stages of training new tasks. To address these issues, Knowledge Distillation (KD) provides a complementary approach. KD enables the transfer of knowledge from a teacher model, trained on previous classes, to a student model learning new classes. By aligning the student model's output with the teacher's softened probability distributions, KD helps guide the activation patterns of young neurons, reducing ambiguity during early training. Furthermore, KD preserves the inter-class relationships encoded in the teacher model, supplementing NICE's age-based neuron management. This explicit knowledge transfer mechanism ensures that the student model retains prior knowledge more effectively while adapting to new tasks.

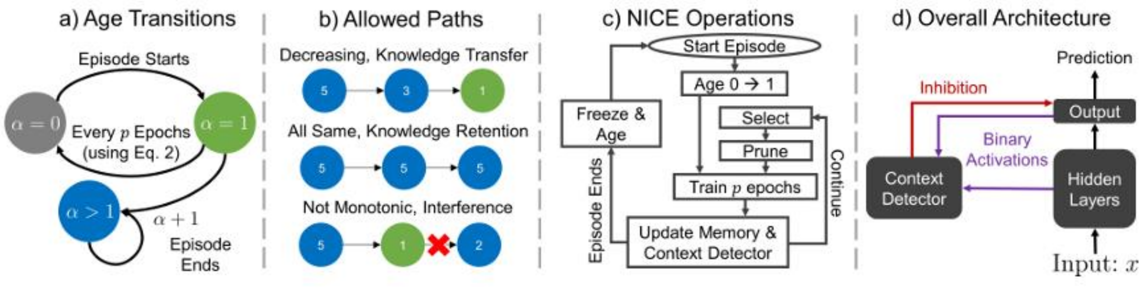
2.2. Importance and Relevance of the Problem

Class incremental learning is a critical challenge in real-world tasks, where models must continuously learn and adapt to new information while retaining previously acquired knowledge. Methods like NICE address this challenge by managing the model's capacity through neurogenesis-inspired neuron freezing and pruning. However, NICE's limitations, particularly its lack of explicit mechanisms to guide inter-class relationship preservation, can hinder its performance in complex, real-world scenarios.

By integrating Knowledge Distillation with NICE, we aim to overcome these limitations. KD provides a structured way to transfer and preserve knowledge, enhancing NICE's ability to mitigate catastrophic forgetting and improving its overall accuracy and adaptability. This combination represents a step forward in creating more robust and efficient class incremental learning models for practical applications.

3. Methodology and Implementation Details

3.1. NICE



3.1.1. Neuron Ages

NICE uses neuron age and activation to evaluate importance within the network, considering age-0 neurons as surplus ability that can transition to age-1 as needed. During training, neuron activations are analyzed, and a subset contributing to at least a specified fraction (τ) of total activation is selected as essential. At the end of each episode, some age-1 neurons revert to age-0, while others mature to age-2 or higher. This approach ensures that necessary neurons are kept while redundant ones are pruned, effectively managing network plasticity and ability.

3.1.2. Avoiding Interference Between Neurons

NICE groups neurons by their age, freezing and pruning neurons aged 2 or older to preserve past knowledge, while using age-0 neurons as surplus capacity, age-1 neurons for learning, and age-2 or older neurons as memory, effectively preventing interference between neurons and ensuring zero forgetting during learning.

3.1.3. NICE Operations

At the start of NICE operations, age-0 neurons are transitioned to age-1, and important neurons are selected based on activation values, while unimportant neurons are pruned. The network is then trained for p epochs, followed by updating the memory and context detector. At the end of the episode, neurons aged 1 or older are frozen and incremented in age to become mature neurons, and the process repeats for the next episode.

3.1.4. Context-Detector

NICE infers the proper episode for predictions by learning conditional probabilities based on neuron activation patterns. It periodically stores information about highly activated neurons as binary vectors in memory and uses logistic regression to calculate conditional probabilities for each episode. These probabilities are then chained to decide the likelihood that a new sample belongs to a specific episode, enabling the model to find the most relevant episode while suppressing irrelevant outputs. The neurons used in this process are already frozen, ensuring stable memory retention during training.

3.2. Knowledge Distillation (Proposed Method)

3.2.1. Knowledge Distillation

Knowledge Distillation is a technique that transfers knowledge from a large model (teacher) to a smaller model (student). The teacher is a complex and highly performant model that provides predicted probability distributions (soft labels) derived from the training data. The student model learns from these soft labels, aiming to achieve similar performance to the teacher, even with a smaller and more efficient architecture.

3.2.2. Knowledge Distillation Loss

$$L_{\text{total}} = \alpha \cdot L_{\text{CE}} + (1 - \alpha) \cdot L_{\text{KD}}$$

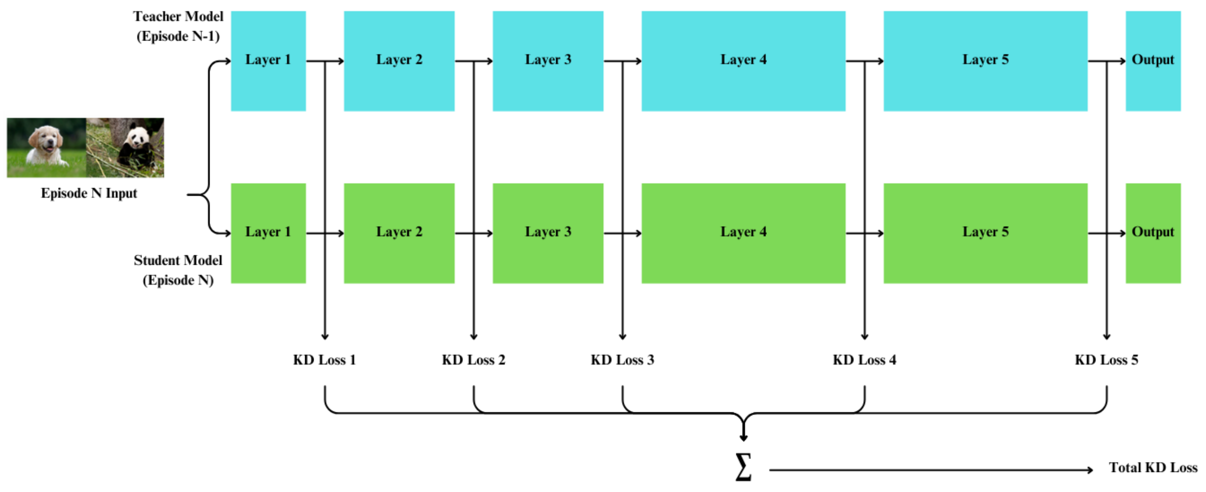
The loss function used in knowledge distillation is designed to train the student model to mimic the probability distribution of the teacher model, combining cross-entropy loss (L_{CE}) and knowledge distillation loss (L_{KD}). Cross-entropy loss minimizes the difference between the student model's predictions and the ground truth labels, helping the student learn accurate predictions. Knowledge distillation loss minimizes the difference between the teacher model's soft labels and the student model's predicted probability distribution, guiding the student to learn the teacher's knowledge. The loss weight (α) adjusts the balance between the teacher's knowledge and the ground truth, determining the emphasis the student model places on each.

$$L_{\text{KD}} = T^2 \cdot \text{KL}\left(\text{SoftMax}\left(\frac{z_t}{T}\right) \parallel \text{Softmax}\left(\frac{z_s}{T}\right)\right)$$

$$\text{KL}(P \parallel Q) = \sum_i P(i) \cdot \log\left(\frac{P(i)}{Q(i)}\right)$$

The knowledge distillation loss (L_{KD}) is primarily defined based on KL divergence, with a temperature parameter (T) applied to smooth the teacher model's output probabilities, making inter-class relationships more pronounced. Here, P stands for the teacher model's soft labels, and Q is the student model's soft labels.

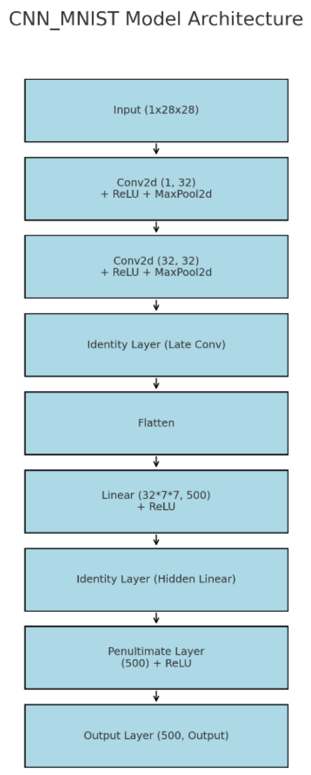
3.3 NICE with Knowledge Distillation



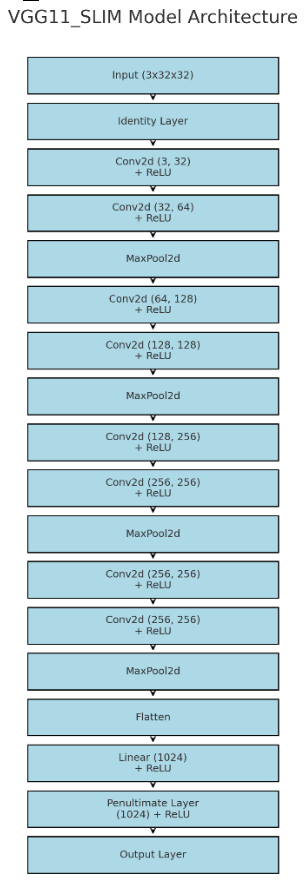
In Episode (N), the input images are simultaneously fed into the teacher model (trained on Episode (N-1)) and the student model (trained on Episode (N)). The teacher model, represented by blue layers and assumed to consist of (Layer 1) to (Layer 5), has already been trained in Episode (N-1), and its outputs are used as "soft labels" or target activations for the student model. The student model, represented by green layers and assumed to have the same architecture as the teacher model, also consists of (Layer~1) to (Layer~5). It learns from the Episode (N) data while being trained to mimic the teacher model's outputs. For each corresponding layer ((Layer 1, Layer 2, ... Layer 5)), the outputs of the teacher and student models are compared to calculate the Layer-wise Knowledge Distillation Loss (KD Loss), which guides the student to emulate the teacher's behavior. The total KD loss is then computed by summing up all layer-wise losses, and the student model is trained to minimize this total loss.

3.4. Model Architecture

3.4.1. CNN_MNIST


CNN_MNIST Model Architecture
Input (1x28x28)
Conv2d (1, 32) + ReLU + MaxPool2d
Conv2d (32, 32) + ReLU + MaxPool2d
Identity Layer (Late Conv)
Flatten
Linear (32*7*7, 500) + ReLU
Identity Layer (Hidden Linear)
Penultimate Layer (500) + ReLU
Output Layer (500, Output)

3.4.2. VGG11_SLIM



3.5. Instructions for Running the Code

Install the libraries listed in Requirement.txt and then run the run.sh file. If you want to change the hyperparameters or settings, you can modify them directly in the run.sh file.

3.6 Hyperparameters

The following hyperparameters were used to configure the experiments:

General Settings

- Seed: 0
- Dataset: MNIST (Experiment 1), CIFAR10 (Experiment 2)
- Number of Tasks: 5
- Number of Classes: 10

Context Detector

- Memo per Class Context: 100 (MNIST), 300 (CIFAR10)
- Context Layers: [0, 1, 2, 3, 4] (MNIST), [0, 1, ..., 11] (CIFAR10)
- Context Learner: Logistic Regression (random_state=0, max_iter=300, C=0.4 for MNIST, C=0.01 for CIFAR10)

Model Architecture

- Model: CNN_MNIST for MNIST, VGG11_SLIM for CIFAR10

Training

- Optimizer: SGD
- Learning Rate: 0.01
- Batch Size: 32 (MNIST), 128 (CIFAR10)
- Weight Decay: 0.0 (MNIST), 0.0001 (CIFAR10)
- SGD Momentum: 0.90
- Phase Epochs: 5 (MNIST), 15 (CIFAR10)

Algorithm-Specific Parameters

- Activation Percentage: 95.0%
- Maximum Phases: 5
- Knowledge Distillation Temperature: 3
- Knowledge Distillation Alpha: 0.5
- Replay Buffer Size: 100

4. Datasets

4.1. Dataset Sources

MNIST: A handwritten digit dataset, downloadable via torchvision.datasets.MNIST. It consists of images of digits from 0 to 9.	**CIFAR-10**: A general object image dataset, downloadable via torchvision.datasets.CIFAR10. It has 10 categories (e.g., airplane, car, bird, etc.).

4.2. Data Structure

MNIST:	**CIFAR-10**:
Total samples: 70,000 (60,000 for training, 10,000 for testing).	Total samples: 60,000 (50,000 for training, 10,000 for testing).
Image size: 28×28 (grayscale, 1 channel).	Image size: 32×32 (color, 3 channels).
Each sample is stored as a tuple: (image, label).	Each sample is stored as a tuple: (image, label).

4.3. Data Preprocessing

MNIST:	**CIFAR**-10:
transforms.ToTensor(): Converts images into Torch Tensors.	transforms.ToTensor(): Converts images into Torch Tensors.
	transforms.Normalize(mean, std): Normalizes the data using the following mean and standard deviation values:
	Mean: (0.4914, 0.4822, 0.4465), Std: (0.247, 0.243, 0.261)

4.4. Sample Count and Distribution

Task Splitting (n_tasks = 5):
Both MNIST and CIFAR-10 split the 10 classes into 5 tasks, with each task containing 2 classes.

MNIST Data Distribution:	**CIFAR-10 Data Distribution**:
Training data: 60,000 → 12,000 per task.	Training data: 50,000 → 10,000 per task.
Validation data: 10% of training data → 1,200 per task.	Validation data: 10% of training data → 1,000 per task.
Testing data: 10,000 → 2,000 per task.	Testing data: 10,000 → 2,000 per task.

4.5. Task-Class Mapping

MNIST:	**CIFAR-10**:
Task 1: [0, 1], Task 2: [2, 3], ..., Task 5: [8, 9].	Task 1: [0, 1], Task 2: [2, 3], ..., Task 5: [8, 9].

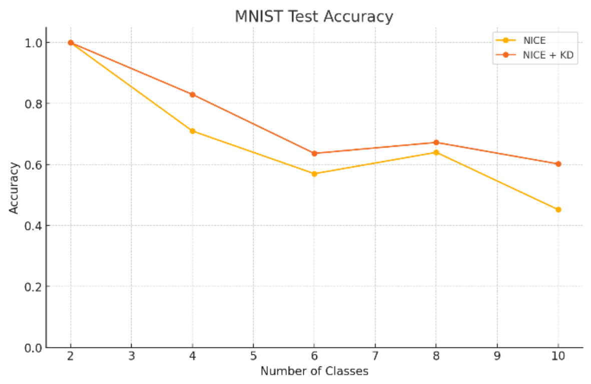
5. Results

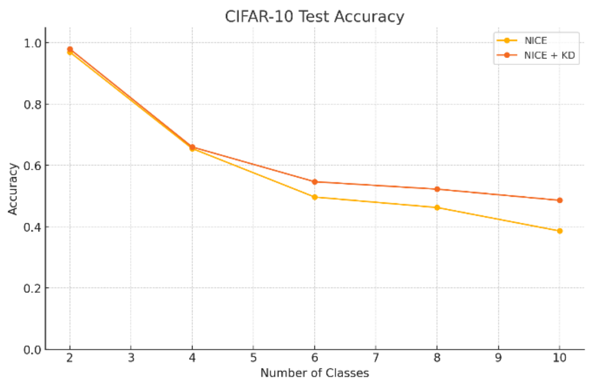
5.1. Accuracy

The results show that integrating Knowledge Distillation (KD) significantly improves NICE's performance on both MNIST and CIFAR-10. For MNIST, NICE achieves an average test accuracy of 44%, while NICE + KD boosts this to 59%, an improvement of 15 percentage points. This highlights KD's effectiveness in mitigating catastrophic forgetting in simpler datasets. For CIFAR-10, NICE achieves 39%, and NICE + KD improves it to 49%, a gain of 10 percentage points. Although the improvement is smaller than in MNIST, it demonstrates KD's ability to preserve inter-class relationships in more complex datasets.

	Datasets	
Method	MNIST	CIFAR-10
NICE	0.44	0.39
NICE + KD	**0.59**	**0.49**

Overall, the combination of NICE with KD achieves more stable performance across tasks, reducing accuracy drop-offs and effectively balancing knowledge retention and new learning. This confirms KD's role in enhancing NICE's robustness in class incremental learning.

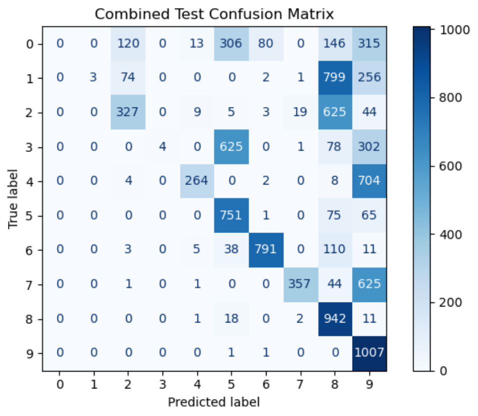




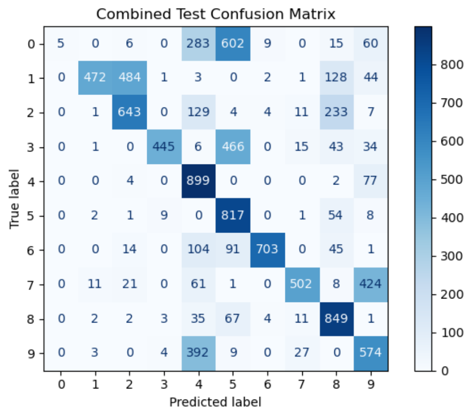
These graphs show that both methods experience a decline in test accuracy as the number of classes increases, which is expected in class incremental learning due to the challenge of keeping prior knowledge while learning new tasks. However, the NICE + KD method consistently outperforms the NICE method across all episodes, proving the effectiveness of Knowledge Distillation in mitigating catastrophic forgetting. In the earlier episodes, NICE + KD shows a significant improvement, showing its strong impact during the first stages of incremental learning. As the number of classes grows, the accuracy of both methods stabilizes, but NICE exhibits a sharper decline, highlighting its difficulty in keeping earlier knowledge. Overall, the NICE + KD method achieves more gradual accuracy degradation, effectively balancing new learning and knowledge retention.

5.2. Confusion Matrix (MNIST)

NICE

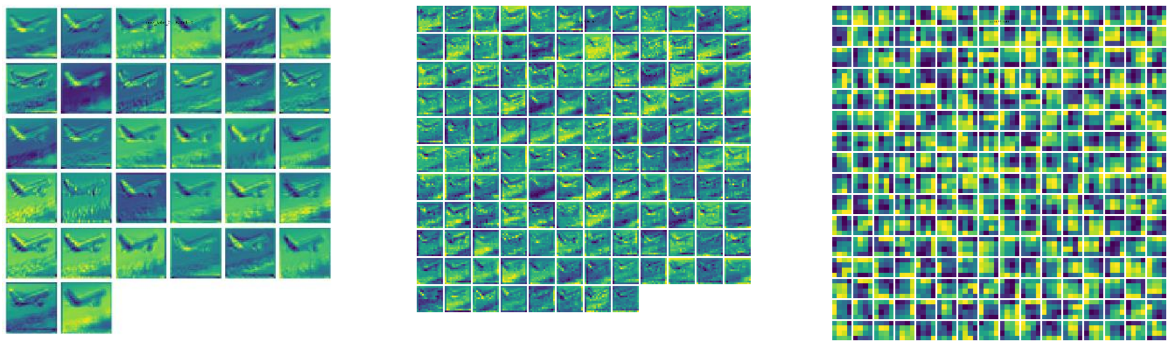


NICE with Knowledge Distillation



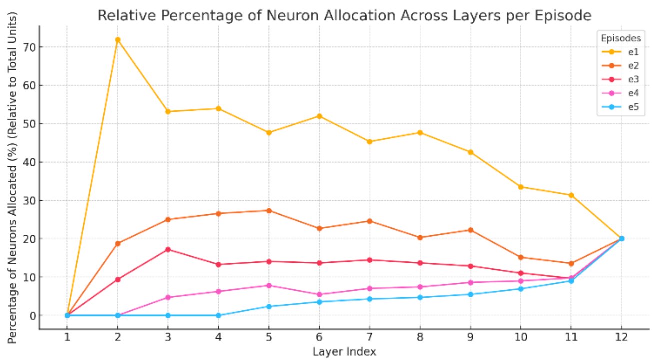
The data compares the performance of NICE and NICE + KD on classification tasks in an incremental learning scenario. The confusion matrices reveal how well each model predicts correct classes (diagonal values) and handles misclassifications (off-diagonal values). NICE shows more misclassifications, particularly for earlier tasks, indicating greater interference and catastrophic forgetting as new tasks are introduced. In contrast, NICE + KD demonstrates stronger diagonal dominance, preserving knowledge of earlier classes while adapting to new ones. By transferring inter-class relationships from the teacher model, Knowledge Distillation reduces confusion between similar classes and improves overall accuracy. These results highlight NICE + KD's ability to balance new learning and knowledge retention effectively, ensuring more stable performance across tasks.

5.3. Feature Map



Feature maps illustrate the patterns and features extracted from input data as it flows through the layers of a neural network. In the provided images, the feature maps generated by the earlier layers (on the left) primarily capture fundamental characteristics, such as edges, corners, textures, and simple patterns. These lower layers focus on identifying general features that form the building blocks for understanding the input.As we progress toward the deeper layers (on the right), the feature maps evolve to represent more intricate and abstract features. These include shapes, object parts, and high-level structures that are crucial for differentiating between classes. This progression demonstrates the hierarchical nature of feature learning in neural networks, where each layer contributes to refining the representation of the input data.

5.4. Layer Occupancy



This plot illustrates the percentage of neurons utilized in each layer across episodes (e1 to e5) relative to the total neurons available in each layer. On the x-axis, the layer indices (1 to 12) are displayed, while the y-axis represents the proportion of neurons allocated in each layer. The results are derived from the CIFAR-10 dataset using the VGG11_SLIM model. In the initial episodes, a larger proportion of neurons is assigned to the shallow layers, indicating the network's emphasis on extracting fundamental features such as edges and textures. As the learning progresses to later episodes, the focus gradually shifts toward deeper layers. This shift reflects the model's increasing reliance on more complex feature representations, such as object parts and abstract patterns, required for distinguishing between classes in CIFAR-10.

6. Conclusion

6.1. Summary

The results highlight the significant impact of combining NICE with Knowledge Distillation (KD) on improving incremental learning performance across various metrics. NICE + KD demonstrates notable enhancements in overall accuracy, more precise class-wise predictions, and improved feature extraction capabilities. By leveraging Knowledge Distillation, the model effectively preserves prior knowledge from earlier tasks while adapting to new ones, minimizing the effects of catastrophic forgetting. This is evident through consistent accuracy improvements, fewer misclassifications across tasks, and a more efficient allocation of neurons, ensuring that network resources are dynamically optimized for both new and retained knowledge. Additionally, the analysis of feature maps reveals the network's ability to progressively extract hierarchical patterns, from basic features in the shallow layers to more complex and abstract representations in the deeper layers. Similarly, the layer occupancy patterns highlight the network's adaptability, as it allocates neurons dynamically based on task demands and episode progression. These findings underscore the robustness and versatility of the NICE + KD approach, showcasing its ability to balance the retention of prior knowledge with the integration of new information, ultimately enabling more effective and scalable incremental learning.

6.2. Future Works

Future work can focus on improving the neuron selection process during each episode. Currently, shallow layers often dominate neuron selection, leaving fewer neurons available for deeper layers in later episodes. Introducing a dynamic allocation strategy that balances neuron usage across layers could enhance network adaptability over time. Improving the accuracy of the Context Detector is another important direction. Replacing logistic regression with a small neural network could better capture non-linear relationships in activation patterns. Additionally, integrating attention mechanisms could allow the model to focus on the most relevant neurons, improving robustness in noisy or overlapping datasets. Dynamic context representations, using modules like LSTMs or Transformers, could replace fixed binary vectors, enabling the Context Detector to adapt to evolving task relationships. A hierarchical approach to context detection, grouping similar tasks into clusters, could further simplify and enhance scalability for complex learning scenarios. These improvements could significantly boost the model's performance in incremental learning tasks.

7. References

Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge Distillation: A Survey. In Pattern Recognition, 123, 107377.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network. arXiv preprint arXiv:1503.0253.

Kang, M., Park, J., & Han, B. (2022). Class-Incremental Learning by Knowledge Distillation with Adaptive Feature Consolidation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 16071–16080).

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., et al. (2017). Overcoming Catastrophic Forgetting in Neural Networks. In Proceedings of the National Academy of Sciences (PNAS), 114(13), 3521–3526.

Lopez-Paz, D., & Ranzato, M. (2017). Gradient Episodic Memory for Continual Learning. In Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS) (pp. 6467–6476).

Rebuffi, S. A., Kolesnikov, A., Sperl, G., & Lampert, C. H. (2017). iCaRL: Incremental Classifier and Representation Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 587–595).

Riemer, M., Cases, I., Ajemian, R., et al. (2019). Learning to Learn Without Forgetting by Maximizing Transfer and Minimizing Interference. In Proceedings of the International Conference on Learning Representations (ICLR) (pp. 1–15).

Van de Ven, G. M., & Tolias, A. S. (2019). Three Scenarios for Continual Learning. In Proceedings of the International Conference on Learning Representations (ICLR) (pp. 1–15).