BTECH THESIS PRESENTATION
on
Online Continual Learning

CHIRAG [NLN]

2 May 2024

Introduction

The traditional machine learning model involves training an algorithm on a static dataset to generate a predictive model, which is then used in real-world applications. This approach, known as isolated learning, treats each task as independent and does not leverage past knowledge, leading to limitations in dynamic, evolving environments.

Contrastingly, human learning is adaptive and cumulative. We integrate new information with existing knowledge, applying what we've learned to new challenges and continuously improving our capabilities. Inspired by this, online continual learning aims to enable artificial systems to learn from continuous data streams, building upon previous knowledge and adapting to new situations without forgetting past information.

Online continual learning represents a paradigm shift towards AI systems that mimic human cognitive flexibility, learning continuously and adaptively. This is crucial in dynamic fields such as healthcare and autonomous driving, where AI must evolve with changing data and conditions. The next sections will explore how this approach not only enhances the robustness and adaptability of machine learning models but also meets the complex demands of modern applications.

Problem Statement

The central challenge in Online Continual Learning (OCL) as it applies to our project is the development of machine learning models that adapt to evolving data distributions without the ability to be retrained on historical data. In dynamic environments such as streaming data services, online platforms, or IoT devices, data continuously flows and changes. Traditional learning models, which often rely on multiple epochs over fixed datasets for training, cannot be applied because we assume that data cannot be stored for privacy, security, or resource constraints. This presents a unique subset of problems:

- One-Pass Learning: The model must learn effectively from data in a single pass, without the luxury of revisiting the same data. This is a significant shift from typical machine learning approaches that iterate over the data multiple times to optimize model performance.
- No Data Retention: Since storing data for future use is not feasible in our scenario, the model must extract and consolidate knowledge from incoming data in real-time. This necessitates developing strategies that allow the model to maintain a balance between adapting to new data (plasticity) and preserving existing knowledge (stability) without the aid of memory-based techniques.
- Algorithm Efficiency and Robustness: The algorithms must be efficient, processing data swiftly as it arrives, and robust enough to handle varied and evolving data distributions. This requires sophisticated mechanisms that can adjust learning rules based on the data's current context and importance, ensuring continual learning and adaptation without degradation of previously learned information.

By addressing these challenges, our project aims to create a machine learning framework capable of continuous adaptation and learning in environments where data privacy, security, or storage limitations prevent traditional learning approaches. This framework will pave the way for more scalable, adaptive, and efficient AI systems suitable for real-world applications where data dynamics are the norm rather than the exception.

Challenges

Online Continual Learning (OCL) faces a number of significant challenges that stem from its core objective to enable models to learn continuously from a stream of incoming data. These challenges include:

1. Catastrophic Forgetting: A critical issue where models overwrite previously learned knowledge upon encountering new data. This leads to a rapid degradation in performance on older tasks when new tasks are learned, jeopardizing the stability of the learning process. Preventing catastrophic forgetting is essential for developing systems that can retain and utilize knowledge over time.

2. Task Interference: As new tasks are introduced, the features and representations learned for these tasks may conflict with those learned for previous tasks. This interference can impair the model's ability to perform well across tasks, complicating the training process and requiring sophisticated techniques to manage task-specific knowledge.

3. Scalability: The ability to manage and learn from an increasing number of tasks without suffering from catastrophic forgetting is a significant hurdle. Scalability not only concerns the model's ability to extend to many tasks but also involves the design of efficient algorithms and architectures that can handle this expansion without a proportional increase in resource demands.

4. Resource Constraints: Continual learning systems often require substantial computational resources, including both memory and processing power. These requirements can limit the deployment of such models in resource-constrained environments, such as mobile devices or embedded systems. Addressing these constraints is crucial for creating practical applications that benefit from continual learning.

5. Data Efficiency: Efficiently learning from limited data is particularly challenging in continual learning scenarios. The model must not only learn effectively from small amounts of data for new tasks but also ensure that the learning process does not disrupt the knowledge acquired from previous tasks. Achieving high data efficiency is key to enabling continual learning in real-world applications where data may be sparse or incomplete.

6. Knowledge Transfer: Developing efficient methods for transferring knowledge from previously learned tasks to new tasks is essential for enhancing model performance and accelerating learning processes. Effective knowledge transfer can mitigate the effects of catastrophic forgetting by reinforcing relevant knowledge and adapting it to new contexts.

Related Work

Continual learning challenges have driven the development of various strategies to prevent catastrophic forgetting and ensure knowledge retention across tasks. Researchers have primarily focused on three methods: Parameter Isolation, Memory-Based Replay, and Regularization.

1. Parameter Isolation:

   - Concept: This approach involves isolating and protecting the parameters that are crucial for previous tasks from being altered when learning new tasks. The idea is to compartmentalize knowledge so that updates do not interfere with what the model has already learned.
   - Examples and Variations: Techniques such as "PathNet" utilize neural pathways to selectively freeze certain parts of the network during the training of new tasks, allowing other parts to evolve (Fernando et al., 2017). Another example is the use of "hard attention to the task" where different subsets of parameters are used for different tasks (Serra et al., 2018).

- Advantages: Effective in preserving specific knowledge related to particular tasks without interference.
- Limitations: Can lead to inefficient use of model parameters and increased model size as new tasks require new parameters.

2. Memory Based Replay:

   - Concept: Also known as rehearsal learning, this strategy involves storing data or representations from previous tasks and revisiting them when learning new tasks. This replay prevents forgetting by maintaining exposure to older knowledge.
   - Examples and Variations: Methods like "Gradient Episodic Memory" (GEM) which stores a subset of the data from previous tasks and uses them to constrain the optimization problem, ensuring that new learning does not interfere with stored examples (Lopez-Paz & Ranzato, 2017).
   - Advantages: Directly tackles catastrophic forgetting by continually incorporating previous tasks into the training process.
   - Limitations: Requires storage space for old data, which can be problematic in terms of scalability and privacy.

3. Regularization Techniques:

   - Concept: Regularization methods add constraints to the learning process to subtly retain old knowledge while acquiring new information. These methods typically introduce a penalty for significant deviations from previous model parameters.
   - Examples and Variations: Elastic Weight Consolidation (EWC) employs a penalty term based on the importance of parameters to previous tasks, which protects them from drastic changes during new learning (Kirkpatrick et al., 2017). Synaptic Intelligence (SI) computes a similar importance measure dynamically as the model learns (Zenke et al., 2017).
   - Advantages: Allows for more flexible use of the model's capacity by modifying parameters in a controlled manner.
   - Limitations: Often requires careful tuning of hyperparameters and may not entirely prevent forgetting if the model is subject to continuous and varied new task exposures.

These approaches reflect the ongoing efforts and diverse strategies in the field to address the inherent challenges of continual learning. Each method has its trade-offs and may be more suitable for different scenarios depending on the specific requirements regarding memory usage, computational efficiency, and task structure. Researchers continue to explore these and other innovative solutions, aiming to create more robust and adaptable learning systems.

Research Gap/Potential Improvements

While Parameter Isolation, Memory-Based Replay, and Regularization have significantly advanced the field of continual learning, each method has limitations that can impact their effectiveness, particularly in dynamic environments like those in our problem scenario.
Disadvantages of Current Techniques:

1. **Parameter Isolation:**

   - **Disadvantage:** While effective at preventing catastrophic forgetting by isolating and protecting certain parameters, this technique can lead to inefficient use of the model's capacity. It often results in a rigid architecture where only a subset of parameters is utilized for each task, which can limit the model's ability to learn complex patterns across tasks.
   - **Impact:** In environments where tasks share underlying patterns or where the ability to generalize across tasks is crucial, this rigidity can hinder performance, preventing the model from leveraging commonalities between tasks.

2. **Memory-Based Replay:**

   - **Disadvantage:** This approach requires storing previous data or representations, which can lead to issues related to privacy, security, and scalability. Additionally, it relies on the availability and quality of the stored examples, which may not always represent the diversity of real-world data adequately.
   - **Impact:** In settings where data cannot be stored or is too diverse to capture effectively with limited samples, Memory-Based Replay can struggle to maintain performance, particularly as the number of tasks grows.

3. **Regularization:**

   - **Disadvantage:** Regularization techniques like Elastic Weight Consolidation (EWC) often require precise tuning and can be overly conservative, penalizing changes to parameters that might be beneficial for new tasks. This can unnecessarily constrain the learning of new tasks, especially in settings where tasks are highly diverse.
   - **Impact:** In dynamic environments where rapid adaptation to new, potentially unrelated tasks is required, the constraints imposed by regularization can prevent the model from adapting effectively, leading to poor performance on newer tasks while trying to preserve knowledge about older ones.

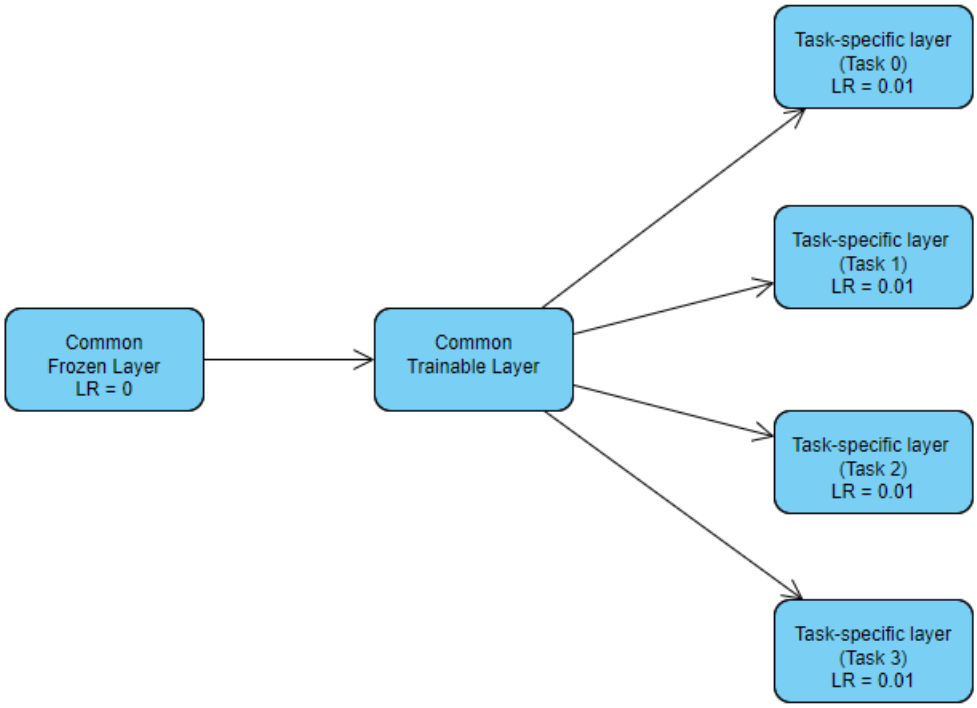
Our Approach

In addressing the challenges of Online Continual Learning, we strategically modified a ResNet-18 model, which was initially pretrained on the ImageNet-1k dataset. Our modifications were aimed at accommodating continual learning through a division of the network's layers into three specific types, each serving distinct roles within the learning process.

Architecture Overview

- Pretrained Model Utilization: Utilizing a pretrained ResNet-18 provides a strong foundational base due to its established capability in extracting robust features from a diverse dataset like ImageNet-1k.
- Layer Categorization: To adapt ResNet-18 for continual learning, we have restructured the network into three categories of layers, each designed to fulfill a specific function in the learning continuum:

  1. Common Frozen Layers: These initial layers remain frozen throughout all tasks. Acting as basic feature extractors, they retain their learned parameters from the ImageNet-1k training. This preservation ensures that the general visual understanding capability is maintained, aiding in stability and preventing the loss of previously acquired knowledge.
  2. Common Trainable Layers: Positioned after the frozen layers, this segment is trainable across all tasks. It serves to refine the general features extracted by the frozen layers into more adaptable forms without being task-specific. These layers help to share useful information across tasks, supporting the generalization capabilities of the model.
  3. Task-Specific Trainable Layers: Unique to our approach is the introduction of separate task-specific layers for each task. These layers are added at the end of the network and are trained with a high learning rate, enabling rapid adaptation to new, specific task requirements. This arrangement allows each set of task-specific layers to develop specialized knowledge without interfering with the learning of other tasks, effectively managing the plasticity needed for new tasks while maintaining stability for learned tasks.



Strategic Learning Rates

- Differential Learning Approach: The learning rates are tailored to the function of each layer group:
  1. Common Trainable Layers: These layers receive a moderate learning rate to allow slow, steady updates that incorporate new learnings without drastic changes that could lead to forgetting.
  2. Task-Specific Layers: High learning rates are applied to these layers to ensure they can quickly adapt to the nuances of each new task. This differentiation in learning speeds is critical for balancing the acquisition of new knowledge with the retention of existing knowledge.

Implications

This architecture ensures our model is not only versatile across various tasks but also efficient in its learning approach. The separation of task-specific layers per task allows for targeted learning and adaptation, which is vital in environments where the model continuously encounters different types of data and tasks. This setup significantly enhances the model's ability to perform continual learning by providing a clear distinction between general knowledge retention and task-specific learning.

Detailed Experimental Setup

Backbone Architecture

- Model Used: ResNet-18, pretrained on the ImageNet-1k dataset. This model serves as the foundational architecture for further adaptation and learning on new tasks in our continual learning experiments.

Datasets

- MNIST: A classic dataset containing grayscale images of handwritten digits.

- CIFAR-10: Comprises 60,000 32x32 color images in 10 different classes, representing various objects.
- CIFAR-100: Similar to CIFAR-10 but with 100 classes.
- Fashion MNIST (FMNIST): Consists of 70,000 grayscale images of 10 fashion categories.

Scenarios

- Setting 1 (Sequential Learning): The model is trained sequentially on entire datasets, beginning with MNIST, followed by CIFAR-10, CIFAR-100, and finally FMNIST. This setting tests the model's ability to retain knowledge across different data distributions and types.
- Setting 2 (Random Batch Learning): After identifying which splits perform well in Setting 1, those splits are tested under a new regime where training batches are randomly selected from a merged pool of all datasets. This tests the robustness and adaptability of the model under less controlled, more stochastic data flows.

Splits for Common Frozen Layers, Common Trainable Layers, and Task-Specific Layers Splits (in millions of parameters):

- 3M - Common Frozen Layers, 3.5M - Common Trainable Layers, 4.8M - Task-Specific Layers
- 3M - Common Frozen Layers, 1M - Common Trainable Layers, 7.2M - Task-Specific Layers
- 4M - Common Frozen Layers, 2.5M - Common Trainable Layers, 4.8M - Task-Specific Layers
- 6.5M - Common Frozen Layers, 2.3M - Common Trainable Layers, 2.4M - Task-Specific Layers
- 4M - Common Frozen Layers, 4.8M - Common Trainable Layers, 2.4M - Task-Specific Layers

Batch Size and Learning Rate

- Batch Size:
  - Setting 1: 256, 64
  - Setting 2: 512, to accommodate the increased variability and volume of data from merged datasets.
- Learning Rate:
  - Common Trainable Layers: 0.000001 to 0.01.
  - Task-Specific Layers: Fixed at 0.01 to promote rapid adaptation to new tasks without overwriting previously learned knowledge too quickly.

Experiments Run

Overview

In order to assess the effectiveness and robustness of our continual learning model, experiments were conducted under two distinct settings. Each setting was designed to test different aspects of the model's ability to adapt and retain information over multiple tasks.

Setting 1: Sequential Task Learning

Procedure: The model sequentially learned from different datasets in a fixed order. The training started with the MNIST dataset, followed by CIFAR-10, CIFAR-100, and finally Fashion MNIST (FMNIST). This setting evaluates the model's ability to retain previously learned knowledge while adapting to new, diverse datasets.

Configurations Tested:

- Splits: Each predefined split configuration (3-3.5-4.8, 3-1-7.2, 4-2.5-4.8, 6.5-2.3-2.4, 4-4.8-2.4 in millions of parameters for Common Frozen Layers, Common Trainable Layers, and Task-Specific Layers, respectively) was tested to determine its impact on learning and memory retention.
- Batch Sizes: Two different batch sizes, 256 and 64, were tested for each split to observe how batch size influences learning dynamics and model stability.
- Learning Rates: A range of learning rates for Common Trainable Layers (from 0.000001 to 0.01) were systematically tested to optimize the balance between new task learning and retention of old knowledge. A fixed learning rate of 0.01 was maintained for Task-Specific Layers to ensure rapid adaptation to new tasks.

Setting 2: Integrated Task Learning

Procedure: Based on the performance in Setting 1, the most effective split configurations were selected for this experiment. Here, the datasets were merged into a single pool from which training batches were randomly sampled. This simulates a more realistic scenario where data from various distributions might be encountered in an unpredictable order.

Configurations Tested:

- Splits: Only the splits that showed superior performance in Setting 1 were tested to focus on optimizing the best configurations under more challenging conditions.
- Batch Sizes: A larger batch size of 512 was used to handle the increased variability and complexity of the merged dataset, facilitating more stable gradient estimates across diverse data samples.
- Learning Rates: The learning rate for Common Trainable Layers of 0.00001 was fixed (for less shift in the weights from their previous configurations). A fixed learning rate of 0.01 was maintained for Task-Specific Layers to ensure rapid adaptation to new tasks.

Rationale and Expected Outcomes

Setting 1 aimed at examining the model's susceptibility to catastrophic forgetting and its ability to transfer knowledge across sequentially introduced tasks.
Setting 2 tested the model's adaptability and robustness when faced with a non-sequential, integrated learning environment, which is more indicative of real-world scenarios where data may not be encountered in a structured sequence.

Each experiment was carefully monitored, with performance metrics recorded after the introduction of each new dataset. This approach allowed us to evaluate both immediate and long-term effects of continual learning strategies under varied conditions.

Results

This section presents the experimental results achieved after testing our continual learning model across various splits, batch sizes, and validation stages. We aimed to assess the impact of different configurations on the model's ability to retain knowledge and adapt to new tasks without catastrophic forgetting. The results are organized into the table, displaying the testing accuracies for each configuration immediately after training on each task (Val 1) and after all tasks have been sequentially trained (Val 2).

Setting 1

The table details the validation accuracies for each network split immediately after training on a specific task (Val 1) and after all tasks have been completed (Val 2). Each row represents a different split configuration, highlighting how the arrangement of parameters influenced the model's short-term and long-term memory

capabilities. B1 refers to the batch size of 256 and B2 refers to the batch size of 64. LR1 refers to the learning rate of common trainable layer and LR2 refers to the learning rate of task-specific layer.

Table 1: Performance Metrics for Different Datasets and Splits (LR1 = 0.00001, LR2 = 0.01)

Split (in M)	Dataset	B1-Val1	B1-Val2	B2-Val1	B2-Val2
3-3.5-4.8	MNIST	98	89	99	81
3-3.5-4.8	CIFAR10	68	45	68	39
3-3.5-4.8	CIFAR100	35	30	39	33
3-3.5-4.8	FMNIST	89	89	90	90
3-1-7.2	MNIST	98	75	99	65
3-1-7.2	CIFAR10	68	36	69	29
3-1-7.2	CIFAR100	36	31	39	33
3-1-7.2	FMNIST	89	89	91	91
4-2.5-4.8	MNIST	98	89	99	60
4-2.5-4.8	CIFAR10	67	28	68	23
4-2.5-4.8	CIFAR100	32	27	38	31
4-2.5-4.8	FMNIST	89	89	90	90
6.5-2.3-2.4	MNIST	96	96	97	96
6.5-2.3-2.4	CIFAR10	62	55	63	50
6.5-2.3-2.4	CIFAR100	32	32	37	36
6.5-2.3-2.4	FMNIST	86	86	87	87
4-4.8-2.4	MNIST	96	95	97	93
4-4.8-2.4	CIFAR10	59	55	60	52
4-4.8-2.4	CIFAR100	29	28	32	32
4-4.8-2.4	FMNIST	85	85	86	86

Key Observations:

- Initial Learning vs. Retention: The last two split configurations, particularly 6.5-2.3-2.4, demonstrated superior performance in both initial learning and knowledge retention across all tasks. For instance, this split showed almost no forgetting in MNIST and very minimal forgetting across CIFAR10 and CIFAR100.
- Impact of Batch Size: Larger batch sizes generally resulted in less forgetting, particularly noticeable in the 3-3.5-4.8 split for MNIST, where B1 experienced significantly less forgetting compared to B2.
- Key Observations from Top Performing Splits

    1. **Stability Across Learning Rates:** Both splits, 6.5-2.3-2.4 and 4-4.8-2.4, demonstrated remarkable stability and minimal forgetting when the common trainable layer learning rate (LR1) varied from 0.000001 to 0.01.
    2. **Impact of Higher Learning Rates:** As LR1 increased, both splits exhibited a general improvement in handling newer tasks like CIFAR10 and CIFAR100, especially when comparing the more challenging CIFAR100 outcomes at the highest learning rate (LR1 = 0.01). This suggests that a higher learning rate might help the model better adapt to more complex data distributions, although at some risk of increased forgetting in certain cases, as seen with the CIFAR datasets.
    3. **Batch Size Differences:** While both batch sizes (B1 and B2) generally showed consistent learning outcomes, larger batch sizes (B1) tended to offer slightly better stability between Val1 and Val2, particularly for the CIFAR datasets. This indicates that larger batches may help mitigate the impact of rapid parameter updates that can lead to forgetting when learning new tasks

Setting 2

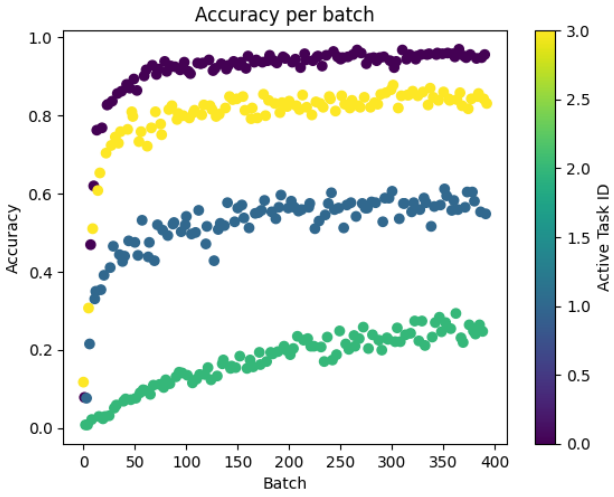


Figure 1: 4-4.8-2.4 split

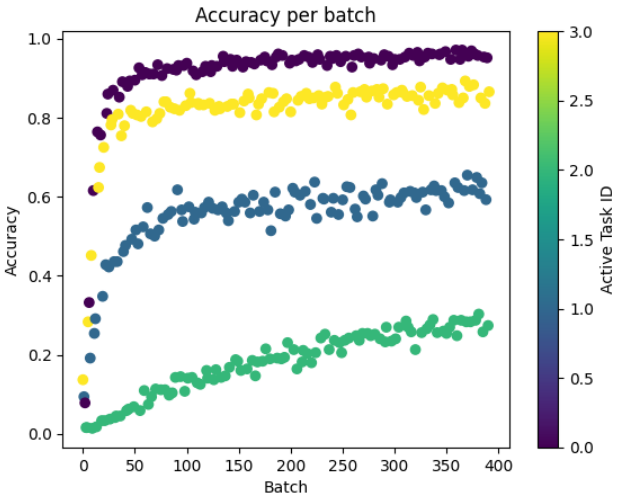


Figure 2: 6.5-2.3-2.4 split

A key aspect of our continual learning investigation was comparing the model's performance across two distinct settings. While Setting 1 involved sequential training on each dataset individually, Setting 2 tested the model's capability to handle mixed data batches drawn randomly from all datasets. This setup was designed to mimic real-world scenarios where data may not arrive in a structured or predictable sequence.

Observations on Task 2 Performance:

- Accuracy Drop in Setting 2: It was observed that the accuracy in Setting 2 was consistently lower compared to Setting 1. Specifically, for Task 2, which involved learning from the CIFAR10 dataset, accuracy in Setting 2 ranged approximately from 20-25%. In contrast, Setting 1 showed a somewhat higher accuracy range of about 30-34%.
- Further Analysis: This performance gap suggests a need for enhanced strategies that could better support learning under less predictable data flow conditions. Approaches such as more sophisticated data handling, improved adaptive learning mechanisms, or even augmented memory systems might be necessary to bolster performance in such settings.

Analysis

Our experiments provide key insights into how batch size, learning rate, and network architecture choices influence performance and memory retention in continual learning scenarios.

- **Impact of Batch Size and Learning Rate on Catastrophic Forgetting**
  - **Finding:** Higher learning rates with smaller batch sizes led to increased catastrophic forgetting.
  - **Implication:** This suggests the need for a lower learning rate when training with smaller batches to stabilize updates and prevent overwriting critical knowledge from previous tasks.
- **Optimization of Task-Specific Layers**
  - **Finding:** Models with thinner task-specific layers experienced less forgetting.
  - **Implication:** Reducing the complexity of task-specific layers helps in retaining prior learning more effectively, suggesting a streamlined approach could be beneficial for managing memory in continual learning frameworks.
- **Learning Rate Insensitivity with Larger Batch Sizes**

  - **Finding:** Changing the learning rate for the common trainable layer had little effect on performance with larger batch sizes.
  - **Implication:** This indicates that larger batches stabilize learning enough that variations in learning rate do not significantly affect model performance, which could guide hyperparameter tuning in environments where computational efficiency is critical.

- **Comparative Performance between Different Settings**
  - **Finding:** Accuracy in Setting 2 was consistently lower than in Setting 1, with accuracy for task 2 being notably lower in the random batch setting.
  - **Implication:** This drop in performance highlights the challenges faced by models in environments with non-sequential and random task exposure, pointing towards a need for mechanisms that can better handle or adapt to random data flows.

- **Optimal Configuration of Network Splits**
  - **Finding:** The network configuration with splits of 6.5-2.3-2.4 (millions of parameters for common frozen, common trainable, and task-specific layers, respectively) achieved the best balance between high accuracy and low forgetting.
  - **Implication:** This split effectively utilizes computational resources and suggests that a balanced allocation of parameters across layers optimizes both learning and memory retention, which is crucial for practical applications.

These findings provide a roadmap for improving continual learning systems. They highlight the importance of optimizing learning rates based on batch size, the benefits of slim task-specific layers, and the need for balanced network configurations to achieve efficient learning and robust memory retention. These insights will guide future adjustments in model training strategies and architecture designs to enhance overall system performance and adaptability.

Conclusion

In this report, we explored adapting a ResNet-18 model, pre-trained on ImageNet-1k, for Online Continual Learning (OCL). We restructured the model into three specific layers—Common Frozen Layers, Common Trainable Layers, and Task-Specific Trainable Layers—to address challenges like catastrophic forgetting and task interference effectively.

Our architectural strategy balanced stability and plasticity within the learning process. Common Frozen Layers preserved essential features, Common Trainable Layers enabled incremental learning across tasks, and Task-Specific Trainable Layers allowed rapid adaptation to new tasks without compromising previously learned information.

Experimental results highlighted how different configurations and batch sizes influence learning and retention, demonstrating that structured learning sequences minimize forgetting, whereas more dynamic environments present challenges that highlight areas for improvement.

Moving forward, our focus will be on enhancing the model's adaptability to random and non-sequential tasks, potentially integrating meta-learning and advanced regularization techniques. This project sets the stage for further development in continual learning, aiming to create AI systems that can learn and evolve in complex, real-world environments efficiently.

Future Works

1. **Testing on Real-World Datasets**
   - **Objective:** To validate and enhance the applicability of our continual learning model in real-world contexts.

- **Details:** Future experiments will involve applying the developed continual learning approach to more complex, real-world datasets such as those for skin disease diagnosis and datasets from the WILDS collection, which are specifically designed to benchmark models against distribution shifts typical of real-world data. This will help assess the model's robustness and practical utility in healthcare and other critical fields.
- **Expected Impact:** Such testing can provide insights into challenges like class imbalance, noisy labels, and non-IID data distributions, which are common in real-world scenarios.

2. **Automated Detection Algorithm for Task Identification**

   - **Objective:** To develop an automated mechanism for identifying the specific task a new batch of data belongs to, thus facilitating dynamic task adaptation.
   - **Details:** Investigate and implement an automated detection algorithm that can accurately predict the task category of incoming data batches. This system would use characteristics of the data to infer its source or type, enabling the model to dynamically switch between different specialized learning configurations.
   - **Expected Impact:** Enhancing the model's adaptability and efficiency, this feature is crucial for deployment in environments where data from multiple tasks is intermingled or not explicitly labeled.

3. **Task-Dependent Splits and Architectures**

   - **Objective:** To tailor the model architecture more precisely for each specific task.
   - **Details:** Develop task-dependent splits in the neural network architecture, where each task would have customized layers or modules best suited to its particular requirements. This could extend to experimenting with different types of neural network layers or configurations that are optimized for specific types of data or learning tasks.
   - **Expected Impact:** By refining the architecture for each task, the model can potentially achieve better performance, more efficient learning, and greater overall system efficiency.

4. **Mixture of Experts Based Architecture**

   - **Objective:** To integrate multiple specialized models for improved learning and inference.
   - **Details:** Explore the implementation of a Mixture of Experts (MoE) architecture, where multiple sub-models (experts) are trained on different tasks but are merged to make collective inferences. This approach allows the system to leverage specialized knowledge from different models while maintaining the ability to handle a broad range of tasks.
   - **Expected Impact:** Such an architecture would not only enhance model performance on individual tasks by utilizing expert knowledge but also improve generalization across tasks through collaborative inference.