FACOLTÀ DI INGEGNERIA DELL'INFORMAZIONE, INFORMATICA E STATISTICA

Master Degree in Data Science

Thesis Defense

Evaluating Continual Learning strategies for audio classification

Candidate:
Alessandro Taglieri, 1890945

Thesis Advisor:
Prof. Simone Scardapane

Academic Year:
2020/2021

SAPIENZA
UNIVERSITÀ DI ROMA

20/01/2022

Agenda

- Motivations and current state of the art
- Properties of Continual Learning
- CL benchmark on ESC-50 dataset
- CL strategies on audio classification
- Metrics results
- Contributions and future works

Motivations and current state of the art

- From Deep Learning to Continual Learning:
  - *Strictly dependence on the quality of the data*
  - Learning from a continuous stream of data that may change through time
  - Avoiding Catastrophic Forgetting problem

- Why Audio classification?
  - *Unexplored data type in the context of Continual Learning*
  - Audio data easily changes over time
  - Starting point to make contributions to Avalanche project

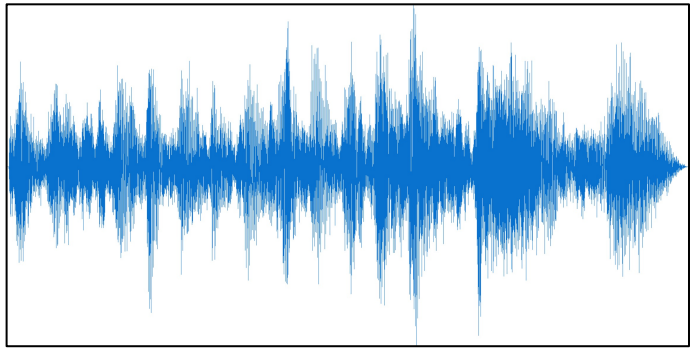


Fig1: Example of audio waveform

Properties of Continual Learning

- **A continual learning algorithm** A^{CL} is an algorithm with the following signature:

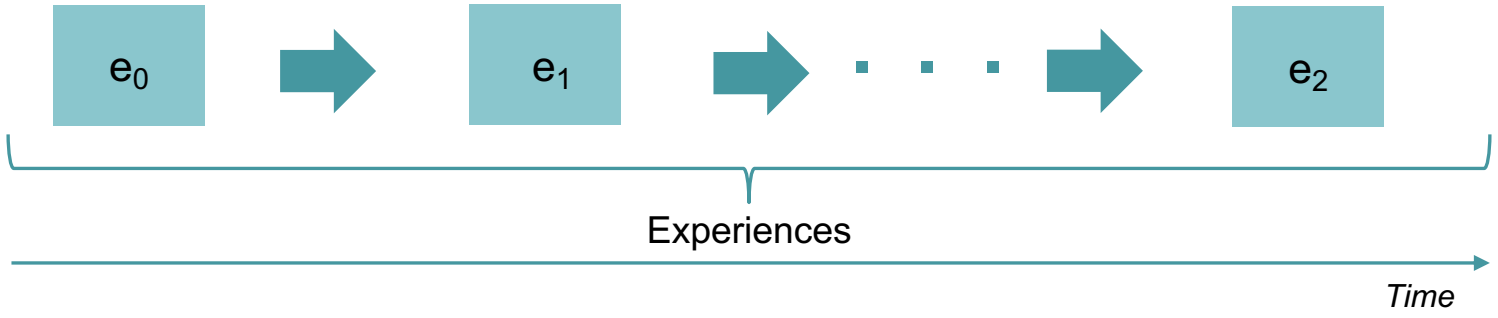
$$\forall D_i \in D, \quad A^{CL} : \langle h_{i-1}, B_i, M_{i-1}, t_i \rangle \rightarrow \langle h_i, M_i \rangle.$$

  - *h_i is the current hypothesis at timestamp i;*
  - *M_i is an external memory where we can save previous training examples;*
  - *t_i is a task label;*
  - *B_i is the training batch of examples.*

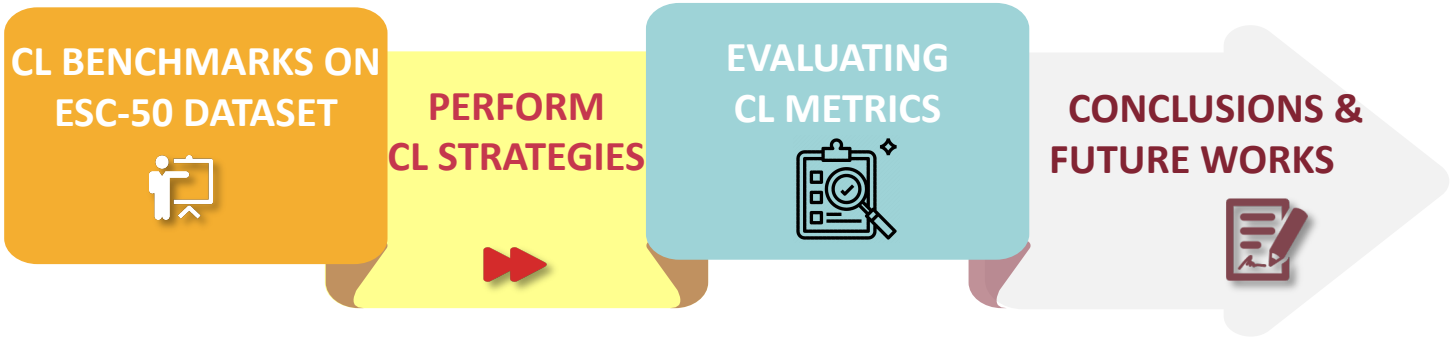
- The objective of a CL algorithm is to minimize the loss L_S over the entire stream of data L_S

$$L_S(h_n, n) = \frac{1}{\sum_{i=1}^{n} |B_{test}^i|} * \sum_{i=1}^{n} L_{exp}(h_n, B_{test}^i)$$

- ***Task**: It is a learning experience characterized by a unique task label t and its target function. Different scenarios of continual learning framework depends on how the concept of task is treated:*
  - *Single incremental task (SIT);*
  - *Multi task;*
  - *Multi incremental task.*



Exepriments set up



Creation of two type of **benchmarks** based on two CL scenarios:

- *Multi task incremental;*
- *Domain incremental*

Performing three different CL **strategies** on benchmarks previously created:

- *Replay;*
- *Naive;*
- *EWC (Elastic Wight Consolidation)*

Strategies evaluation on different CL **metrics** with *five* different executions:

- *Accuracy;*
- *Loss;*
- *Forgetting;*
- *BWT (Backward Transfer);*
- *System metrics*

Contributions, considerations and possible future works:

- *Testing more CL strategies;*
- *New benchmark on different data;*
- *More contributions in Avalanche*

Audio classification: ESC-50 Dataset

- The **ESC-50 dataset** consists of 2 000 labeled environmental recordings equally balanced between 50 classes (40 clips per class). For convenience, they are grouped in 5 loosely defined major categories (10 classes per category):
  - *animal sounds,*
  - *natural soundscapes and water sounds*
  - *human (non-speech) sounds,*
  - *interior/domestic sounds,*
  - *exterior/urban noises.*

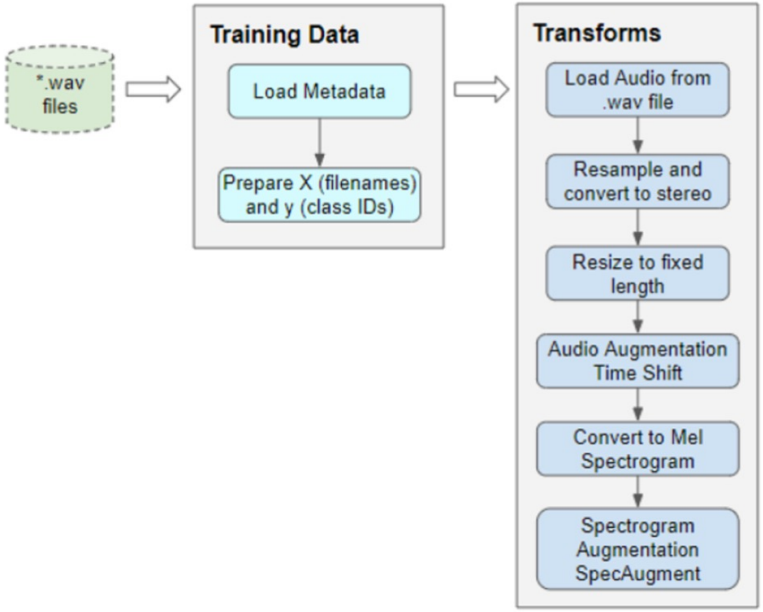


Fig3: Audio preprocessing

Animals	Natural soundscapes & water sounds	Human, non-speech sounds	Interior/domestic sounds	Exterior/urban noises
Dog	Rain	Crying baby	Door knock	Helicopter
Rooster	Sea waves	Sneezing	Mouse click	Chainsaw
Pig	Crackling fire	Clapping	Keyboard typing	Siren
Cow	Crickets	Breathing	Door, wood creaks	Car horn
Frog	Chirping birds	Coughing	Can opening	Engine
Cat	Water drops	Footsteps	Washing machine	Train
Hen	Wind	Laughing	Vacuum cleaner	Church bells
Insects (flying)	Pouring water	Brushing teeth	Clock alarm	Airplane
Sheep	Toilet flush	Snoring	Clock tick	Fireworks
Crow	Thunderstorm	Drinking, sipping	Glass breaking	Hand saw

Tab1: ESC-50 classes

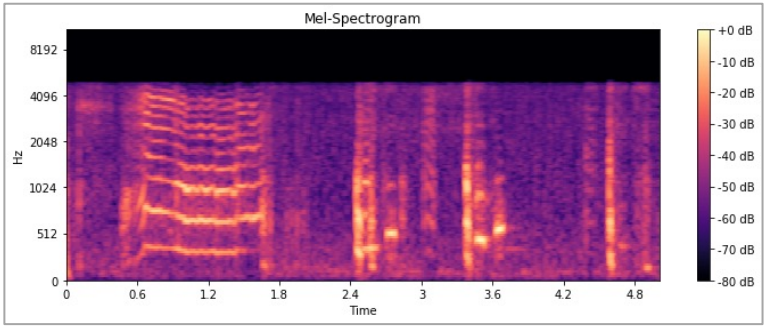


Fig4: Example of mel-spectogram

CL scenarios

- **Multi task incremental:** Given a sequence of train and test dataset, it creates the continual stream of data as a series of experiences. Each experience will contain all the instances belonging to a certain set of classes and a class won't be assigned to more than one experience.

- **Domain Incremental:** Given a sequence of train and test datasets, it creates the continual stream of data as a series of experiences. This generator splits the original training set by creating experiences containing an equal amount of patterns using a random allocation schema.

Scenario	*Required at test time*
Task-IL	Solve tasks so far, task-ID provided
Domain-IL	Solve tasks so far, task-ID not provided

CL strategies

- **Naive:** It just incrementally fine-tunes a single model without employing any method to contrast the catastrophic forgetting of previous knowledge.

- **EWC (Elastic Weight Consolidation):** It computes importance of each weight at the end of training on current experience. During training on each minibatch, the loss is augmented with a penalty which keeps the value of the current weights close to the value they had on previous experiences in proportion to their importance on that experience.

- **Replay:** the idea is that we do not preserve specific set of examples exactly as they are seen in the past but this approach train generative models on the data distribution. A typical way to implement a generative replay is through dual models.

Avalanche: an end-to-end library for Continual Learning

- Avalanche is an **End-to-End Continual Learning Library** based on *PyTorch*, born within **ContinualAI** with the unique goal of providing a collaborative and community-driven open-source (MIT licensed) codebase for **fast prototyping, training and reproducible evaluation** of continual learning algorithms.

- Avalanche can help Continual Learning researchers and practitioners in several ways:
  - *Write less code, prototype faster & reduce errors.*
  - *Improve reproducibility.*
  - *Improve modularity and reusability.*
  - *Increase code efficiency, scalability & portability.*
  - *Augment impact and usability of your research products.*

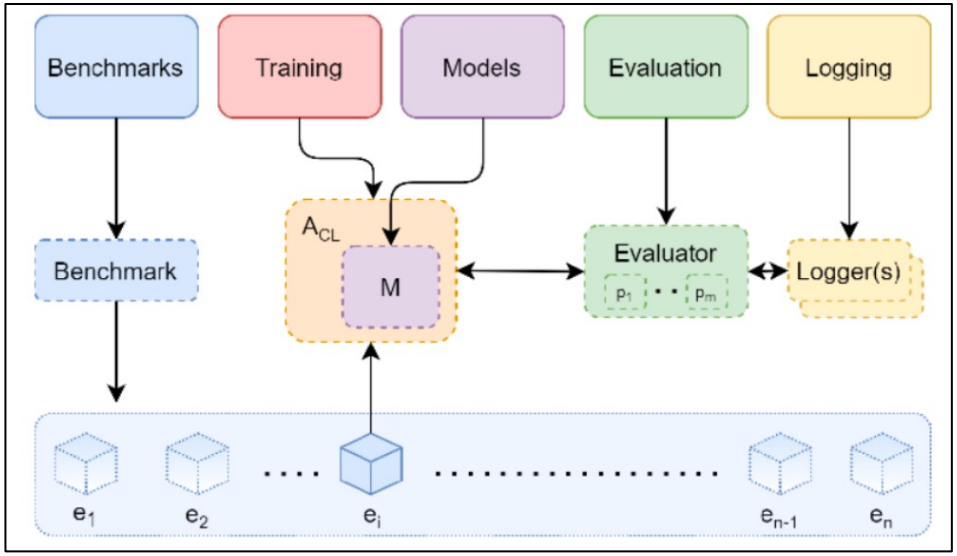


Fig2: Avalanche components

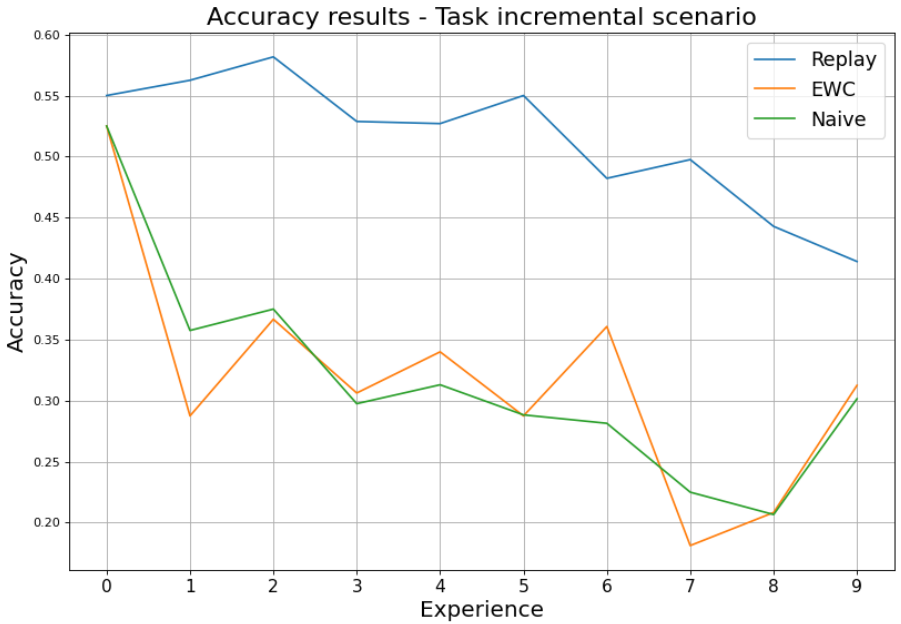
Avalanche

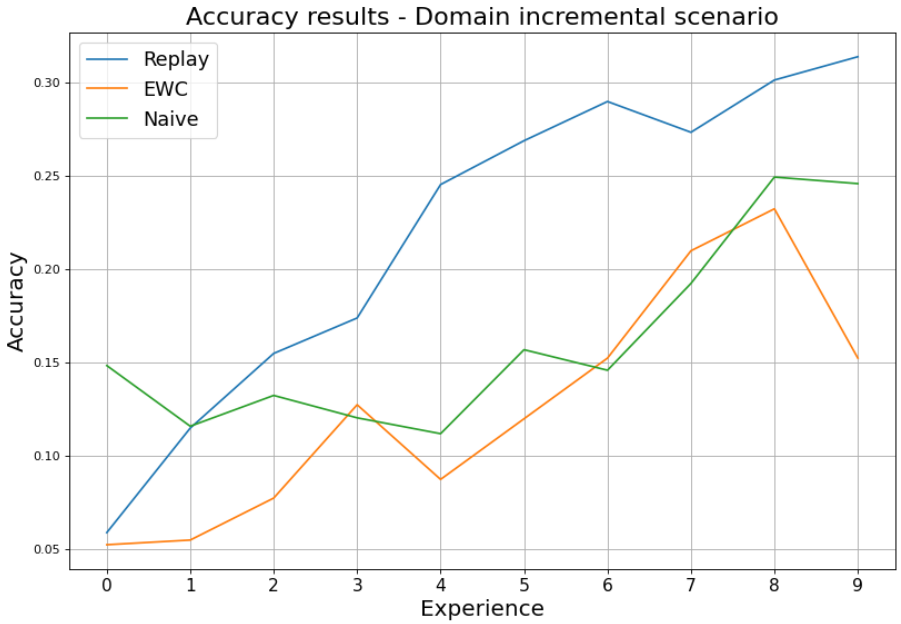
powered by

ContinualAI

Evalauting CL metrics: Accuracy

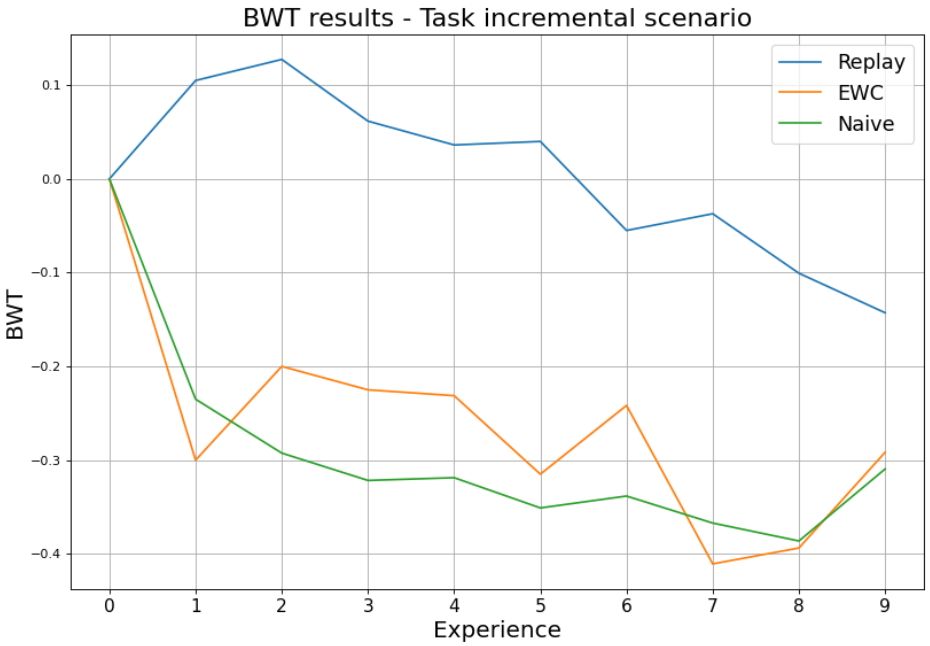
- **Accuracy**: In MIT, it is calculated in each iteration, after every training on different experience and we take in consideration the average of the accuracy calculated on some tasks. In DI this metric is calculated over global data set.

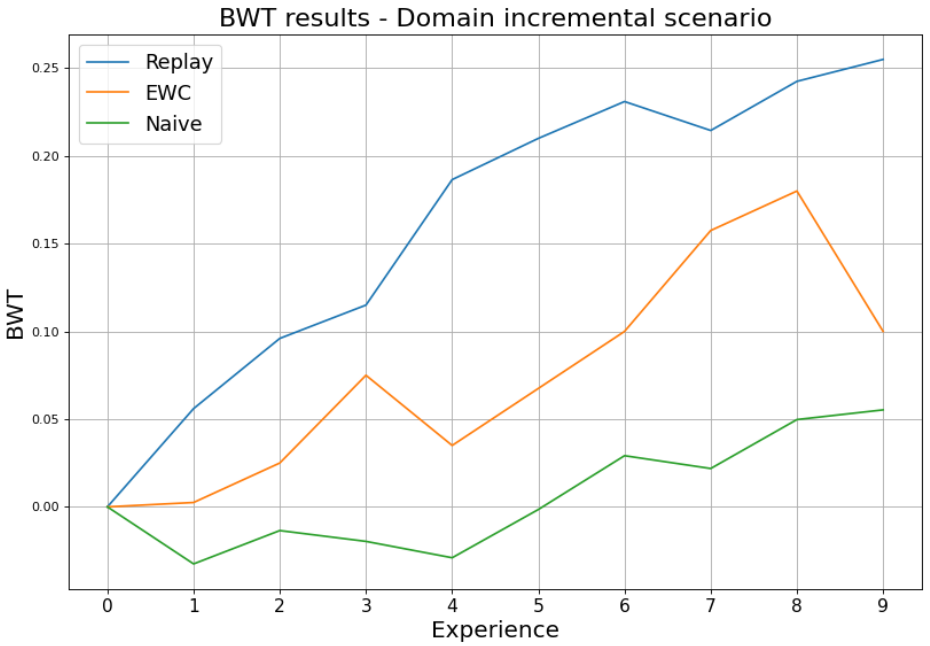




Evalauting CL metrics: Backward Transfer *2/4*

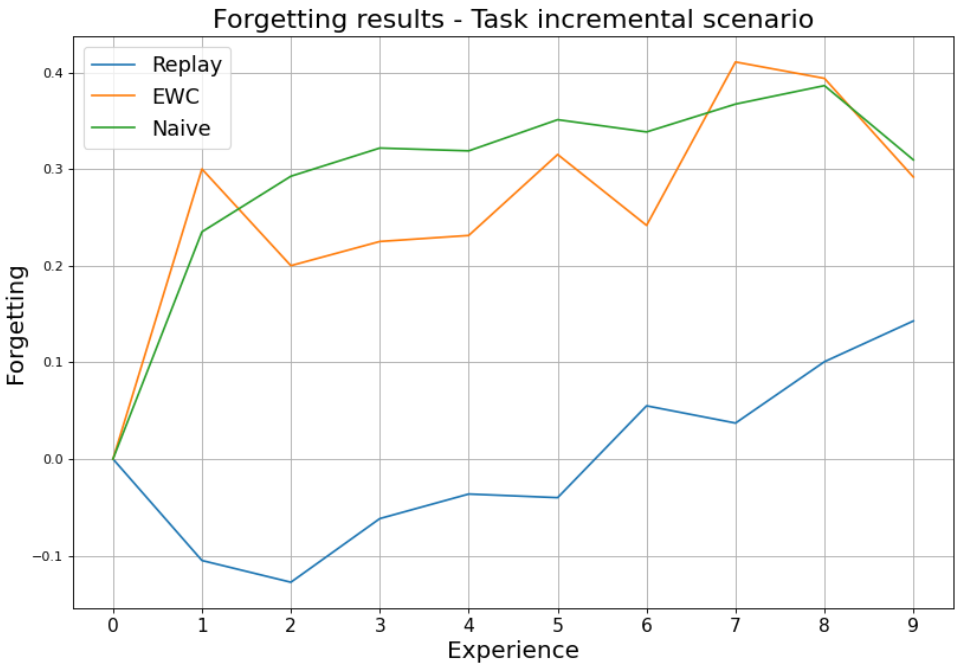
- **BWT (Backward Transfer):** it measures the influence that learning a task has on the performance on previous tasks. It is considered as the average of the backward transfer after each task.

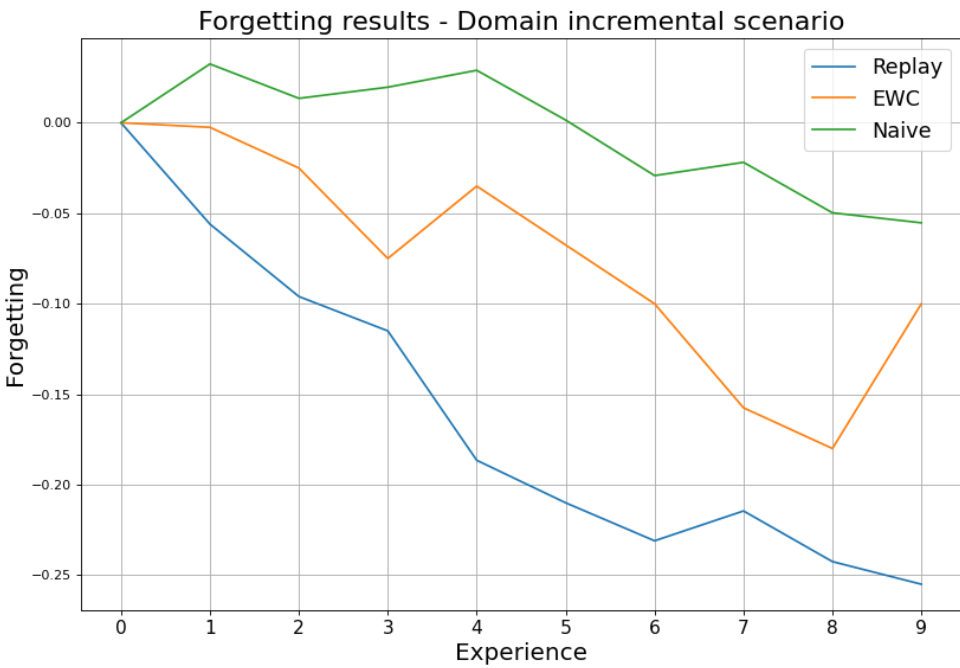




Evalauting CL metrics: Forgetting

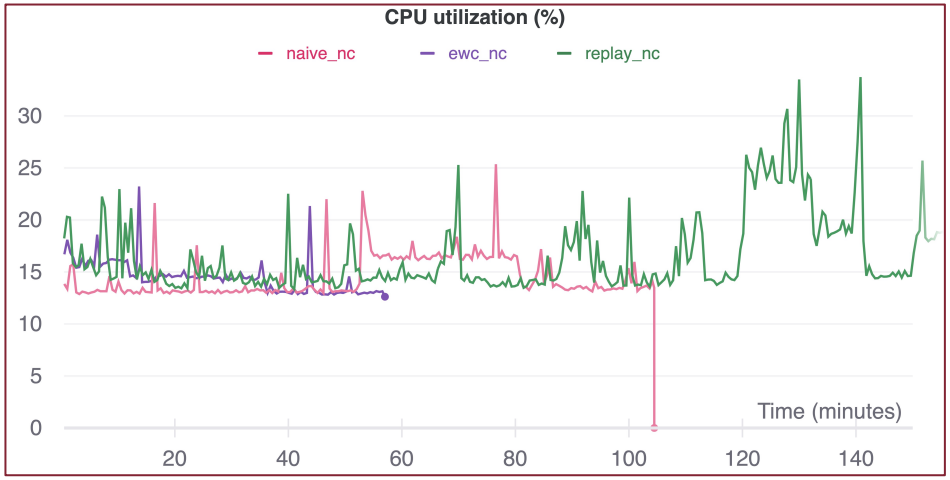
- **Forgetting**: it is calculated from BWT value and when it gets values below zero, it means that our strategy comes across the catastrophic phenomenon. It is considered as the average of the forgetting after each task.

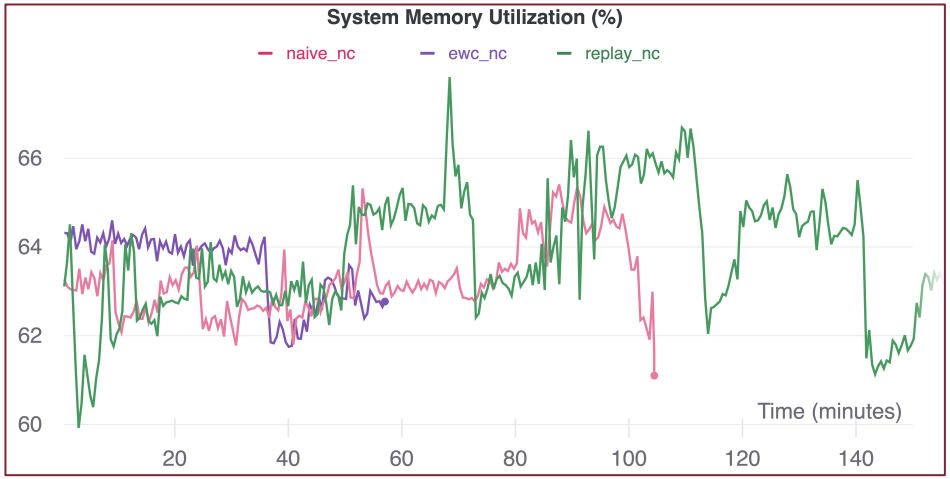




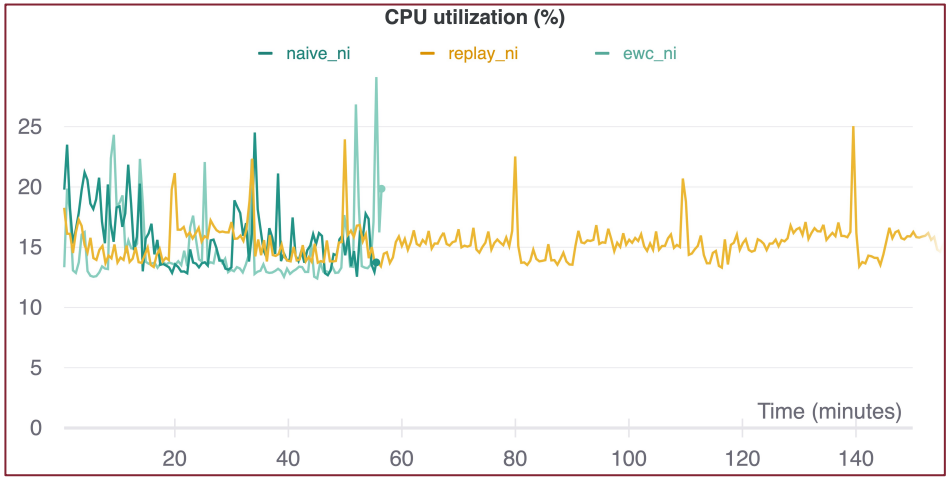
Evalauting CL metrics: System Metrics *4/4*

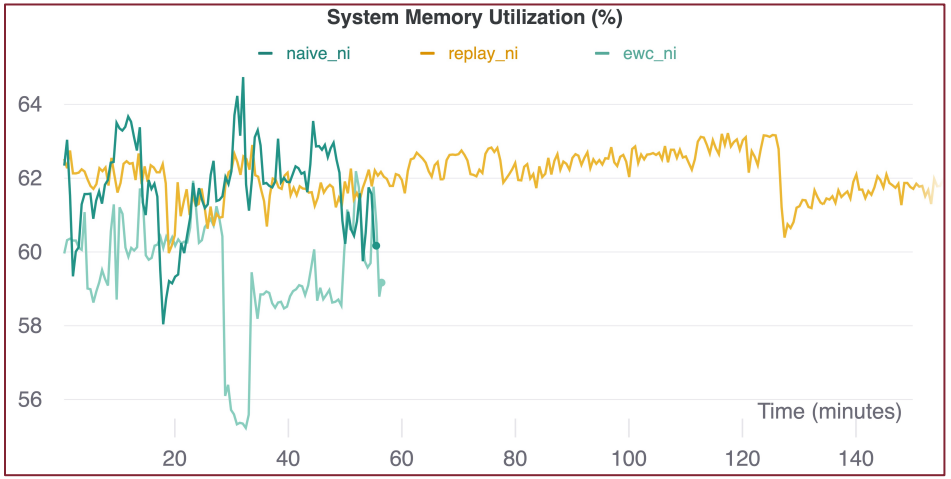
❑ Multi Task Incremental





❑ Domain Incremental





Conclusions and Contributions

From all results obtained by evaluation of Continual Learning strategies, **Replay algorithm** seems to be the best strategy, even if metric values are not very optimal, while **EWC** doesn't work well in the context of Audio Classification. Among two CL scenarios explored, during Domain Incremental, catastrophic forgetting phenomenal is not present; while in MIT it is a bit present but the performance (in terms of accuracy and loss) is better.

- ❑ Presented results can also be seen as a ***contribution of Avalanche*** for unexplored scope, such us audio classification.
- ❑ We gave a working implementation of two different benchmarks based on Continual Learning scenarios (*Multi task incremental* and *Domain Incremental*) on ESC-50 dataset for the Environment Sound Classification.

❑ Future Works:

- Evaluation of other Continual Learning strategies over the same benchmarks;
- *Optimization or creation from scratch of new CL strategies (possibly starting from Replay algorithm)*
- New contributions in Avalanche library in the context of audio classification, creating more benchmarks on different dataset and introducing possiblly new algorithm in order to test new benchmarks.

QUESTIONS?

(All thesis material - Github Repository)

THANK YOU FOR YOUR ATTENTION!