Dual-Memory Model for Incremental Learning: The Handwriting Recognition Use Case

Melanie Piot, Berangere Bourdoulous, Jordan Gonzalez, Aurelia Deshayes, Lionel Prevost
Learning, Data & Robotics Lab, ESIEA, Paris, France

INTRODUCTION

Deep learning pros and cons:

☺ End-to-end learning, representation learning
☹ Current models don't easily adapt to new data/class;
☹ Their training only converges when fed with huge labeled databases;
☹ The need for energy and computing power is constantly increasing;

We propose here an original, dual memory model inspired from Baddeley model.
We use it to learn first (Task1) printed digits and then (Task2) handwritten digits, incrementally.

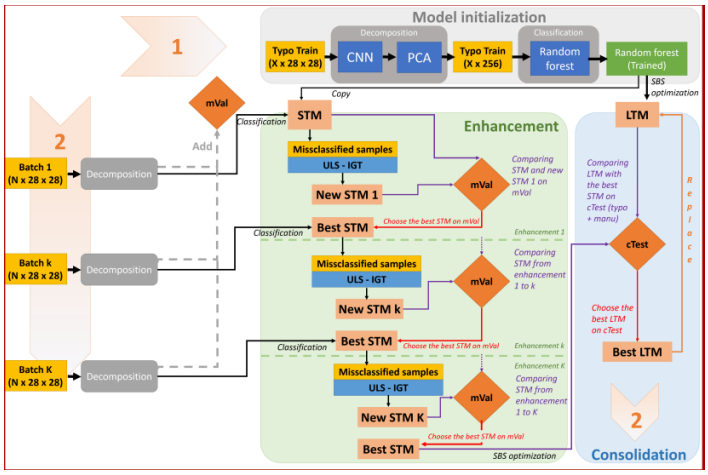
INITIAL LEARNING: TASK 1

The machine is like a 6-year-old child who is learning to distinguish (printed) digits.

❶ **Visual sketchpad**: A CNN (2 conv/pooling layers +2 dense layers) is trained to recognize digits (+2K fonts) → **feature extraction (256 dimensions)**.

❷ **Implicit memory**: Long-Term Memory (LTM) is a random forest (100 trees, max-depth=22) trained on these feature vectors.
A copy of the LTM is stored in the **Short-Term Memory** (STM).

❸ **Explicit memory**: Features are decorrelated (using PCA) and statistical 1D-parameters (μ, σ) are stored to regenerate Task1 samples (see *Enhancement*).



CONCLUSION

This study propose a translation of Baddeley's model into a model capable of learning incrementally as the human brain.

- It is able to learn incrementally new data and classes (recent development);
- It avoids catastrophic forgetting (of previous tasks);
- It gets closer to the functioning of human memory;
- It has limited computing power and energy requirements;
- We found a good approximation of the minimum data needed to learn incrementally Task2 (K = 5; N = 100).

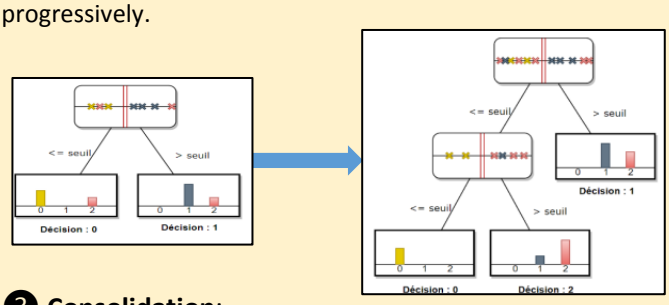
INCREMENTAL LEARNING: TASK 2

Our 6-year-old child lives experiences (class, homework) during the day.
At night, the child integrates all the day's experiences.

❶ **Enhancement**:

The **STM buffer** (FIFO batch: ***N*** samples/experiences) is used to train the STM using the *incremental growing tree* strategy.

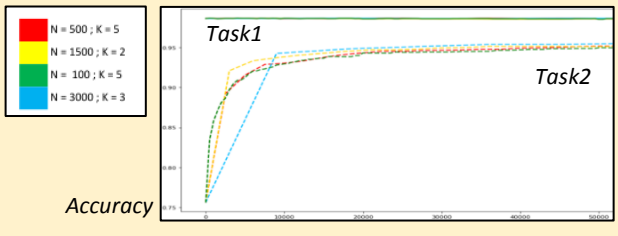
The explicit memory is used to avoid *catastrophic forgetting*. During the ***K*** enhancement (***K*** Batches), the STM is updated progressively.



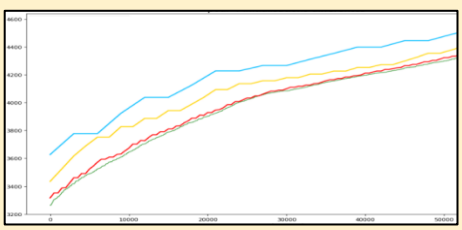
❸ **Consolidation**:

The STM is optimized by applying *Sequential Backward Selection* on trees.

If it performs better than the LTM on both tasks, it is stored in the LTM.



Accuracy



#leaf/tree

	Task 1	Task 2	Model size
Initial training	0.99	0.77	100%
Incremental training	0.99	0.95	130%