Continual Learning

Course structure and motivation

Antonio Carta

antonio.carta@unipi.it

What is Continual Learning?

Motivations and origins of the field

Machine Learning Definition

"A computer program is said to **learn** from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**."

Tom Mitchell, Machine Learning

Offline ML vs Mitchell ML

In practice, up to now you have seen only offline training:

"A computer program is said to **learn** from a dataset **D** sampled **i.i.d.** from a task **T** and performance measure **P**, if its performance at task **T**, as measured by **P**, improves with dataset **D**."

Actual Machine Learning Methods

"A computer program is said to **learn** from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**."

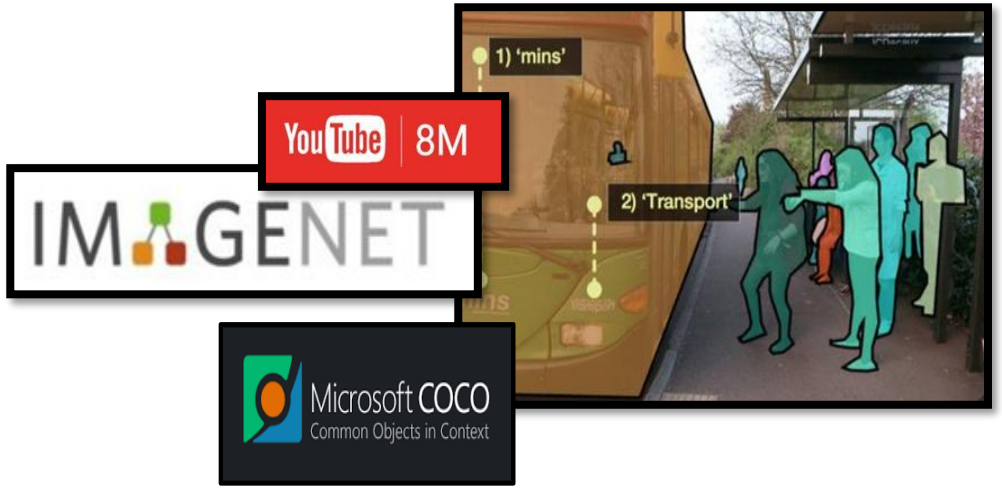
Tom Mitchell, Machine Learning

Offline ML=everything you have seen in ML and ISPR courses

What we will see in this course is much closer to this ideal definition, with classes of tasks and incremental learning over time

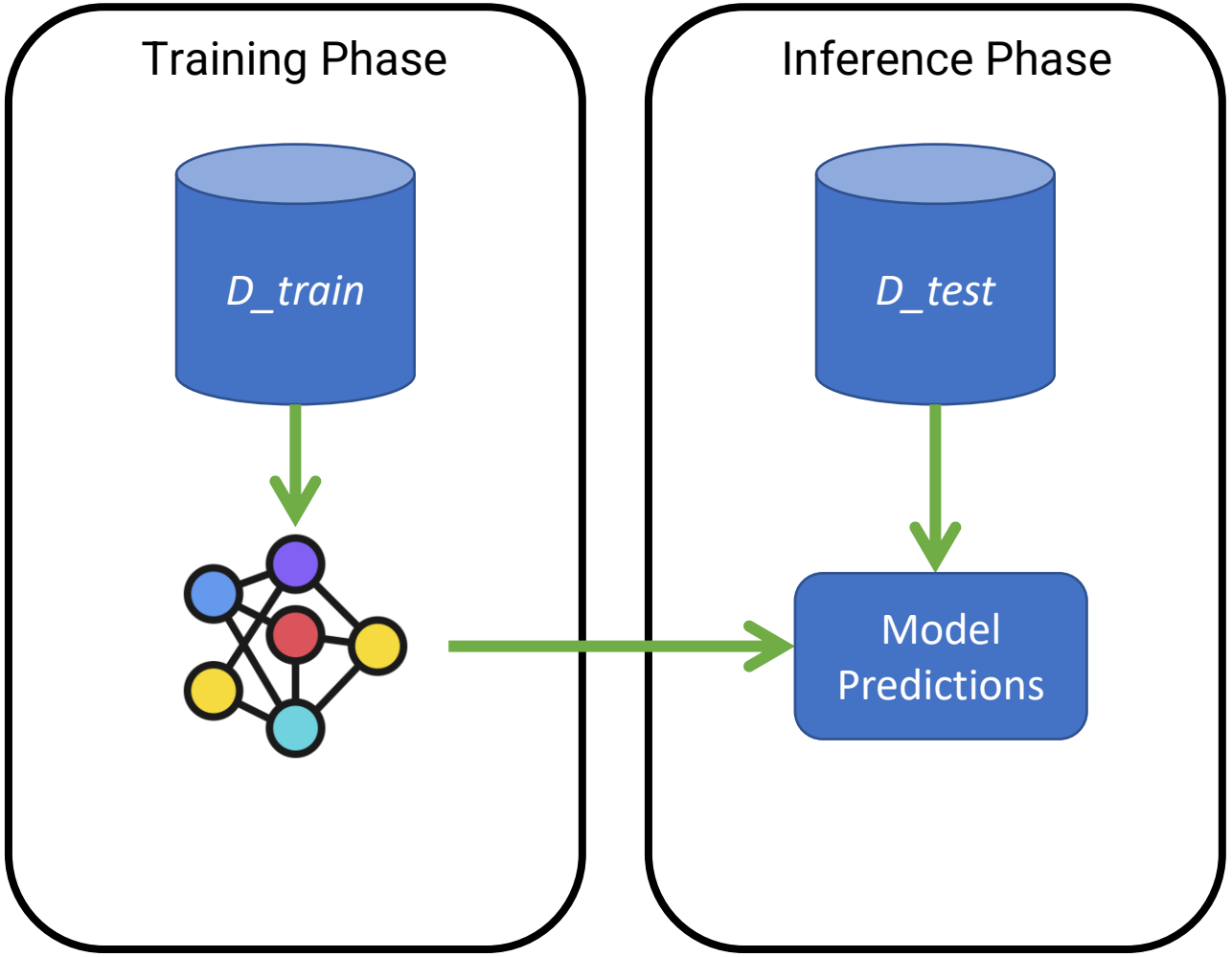
Offline Machine Learning – State of the Art

- Deep learning models today have a high performance in common vision benchmarks
- The key behind their success are:
  - Large models
  - Computational power (M$)
  - Huge amounts of high quality data for training (again, $$$ and time)
  - Having all the data before starting training ($$$ and time)
  - There are also other more theoretical reasons that we will see during the course



Offline Machine Learning

- What happens if I don't have a single dataset?
- How do I use a pretrained model or update a previous model?
- How do I pretrain the model?
- What if my train and test distributions differ?



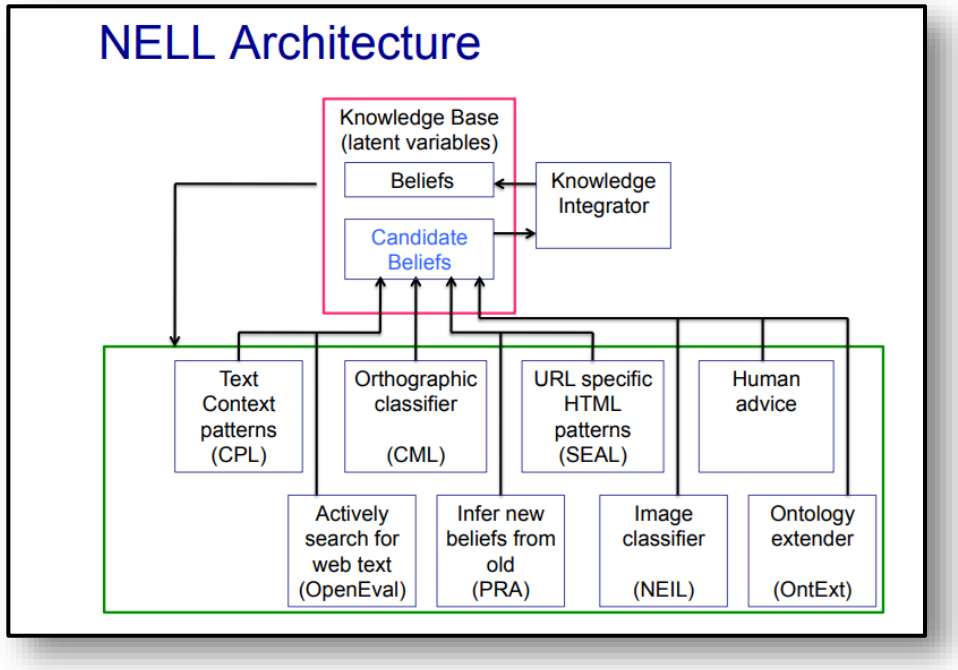
ML Desiderata

- **Efficiency**: learning shouldn't require burning the entire Amazon forest to train a single image classification model
- **Scalability**: learning algorithms should scale well with increasing model size and amount of data
- **Adaptability**: models should adapt to different train and test distributions
- **Reuse**: we want to be able to start learning from a model trained on related tasks
- **Incremental Learning**: we want to update the model over time

Offline ML does not satisfy any of these desiderata

Historical Example – NELL

- Semi-Supervised Learning System
  - Semi-supervised = limited need for labeling
- Ran 24x7, from January, 2010 to September 2018
  - Lifelong learning
- Combination of many learning algorithms (CPL, CML,SEAL, OpenEval, PRA, NEIL)
  - Modular system
- Intended as a case-study for a never-ending learning agent



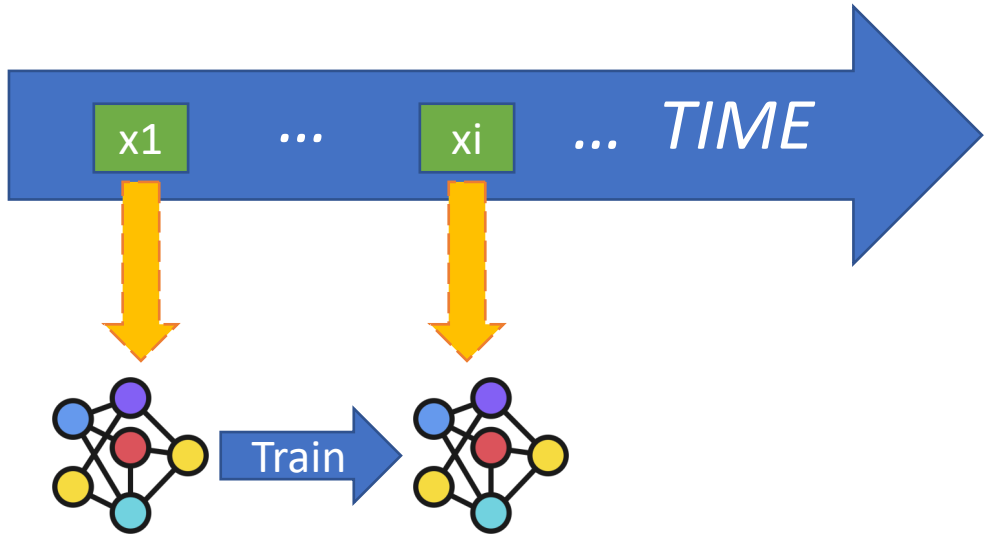
T. Mitchell et al. Never-Ending Learning. Communications of the ACM, 2018.

ML with nonstationary data

Real-world problems beyond static iid data

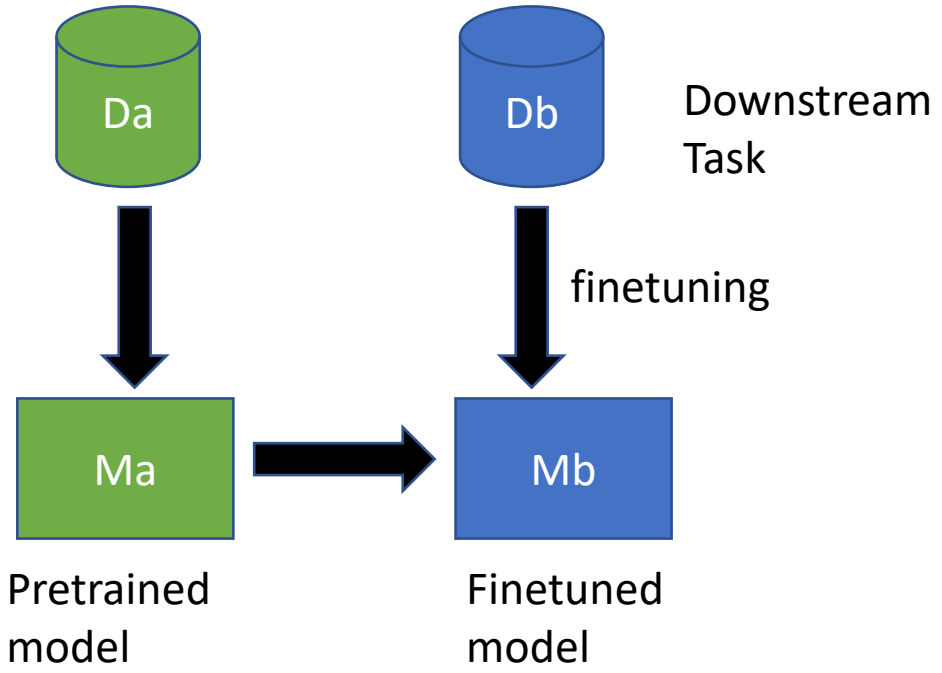
Online Machine Learning

- Learning one sample at a time
- Efficiently
- On nonstationary data (not i.i.d.)



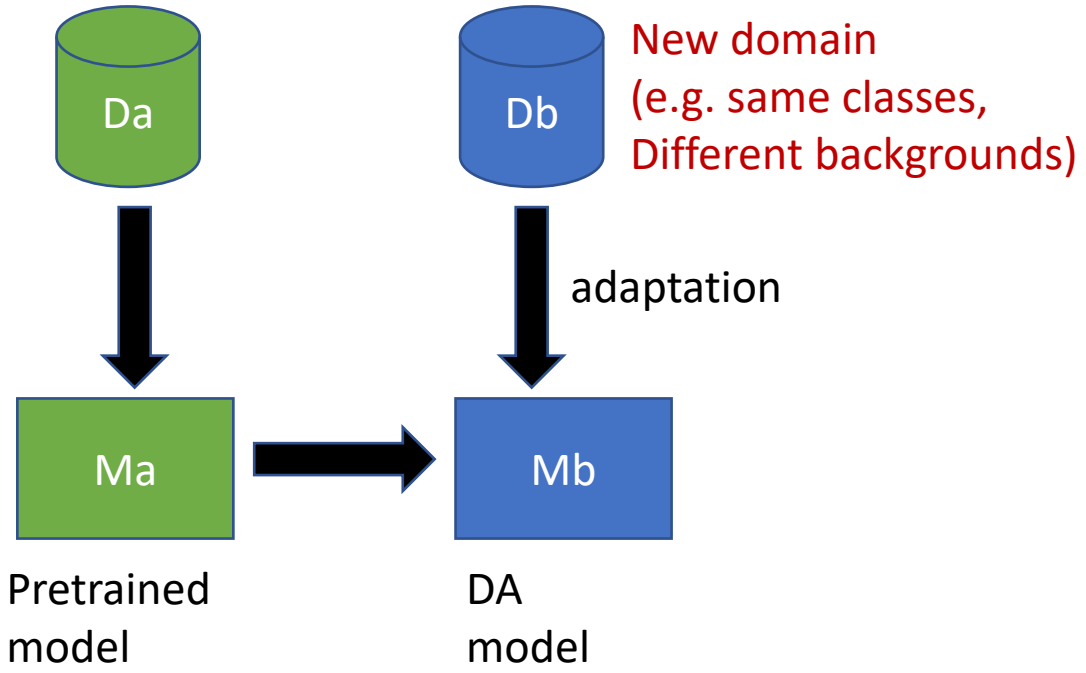
Transfer Learning

- Given a large pretrained model
  - Trained on a large dataset ($$$)
- Finetune a new model on a novel **task**
  - On a small dataset
- Example: given a language model, train a sentiment analysis model



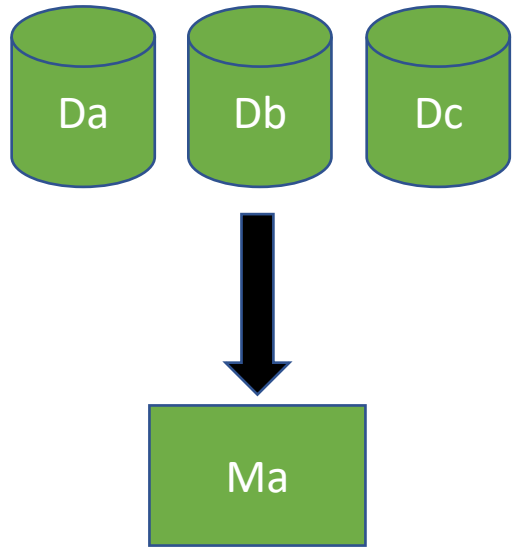
Domain Adaptation

- Given a large pretrained model
  - Trained on a large dataset ($$$)
- Finetune a new model on a novel **domain**
  - On a small labeled dataset
  - Sometimes, we may have access to unlabeled data
- Example: new images have a different background



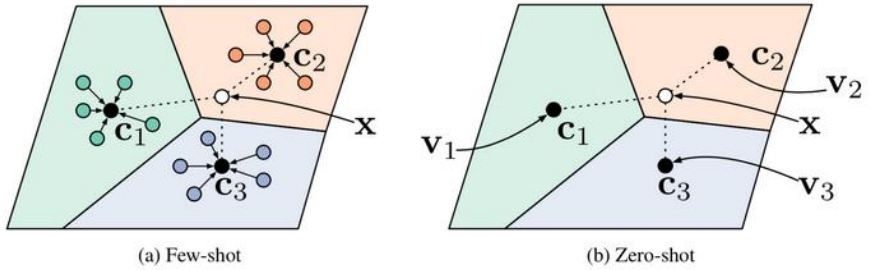
Multi-Task Learning

- Can we train a deep neural network that is able to solve multiple tasks together?
  - Can we share some layers?
  - How do we split the network into task-specific and general components?
  - How do we train the components?



Meta-Learning

- Meta-learning is the problem of learning the optimization algorithm itself
  - Learning an optimizer
  - Automatic hyperparameter optimization
  - Neural Architecture Search
  - Learning a model that quickly generalize to new tasks
- The underlying assumption is that there is a common structure in the family of tasks that we want to solve



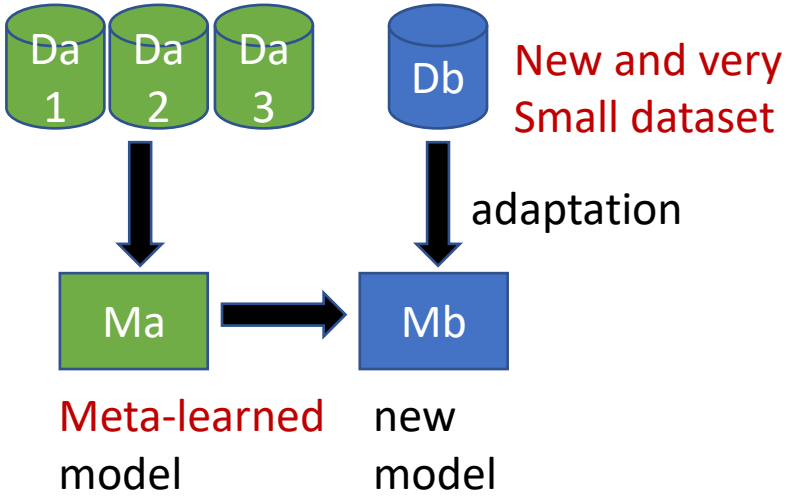
(a) Few-shot (b) Zero-shot

Example: learn an embedding space That generalize quickly to small novel tasks

Image: J. Snell, et al. 2017. "Prototypical Networks for Few-Shot Learning."

Few-shot learning

- What if the datasets are very small? E.g. 5 instances per class?
- We need to learn a generic model that can be quickly adapted
  - How do we learn the meta-model?
  - How do we adapt it?



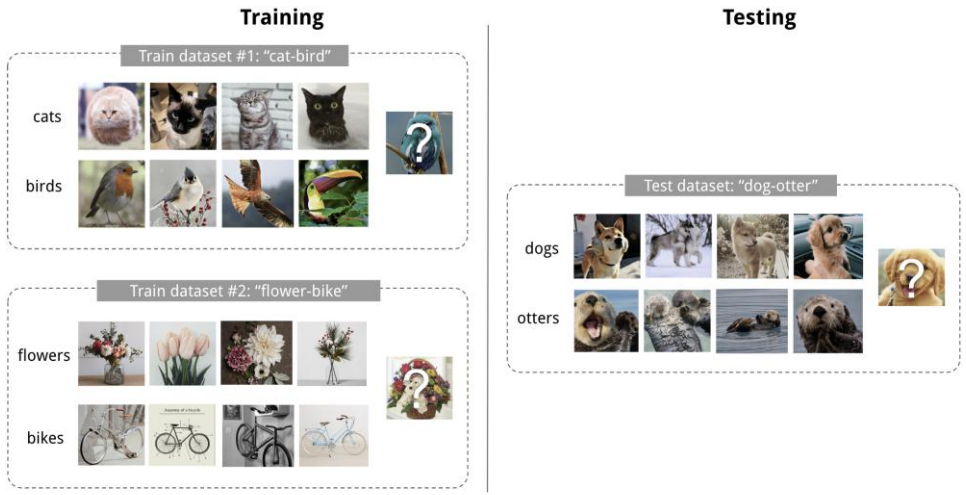
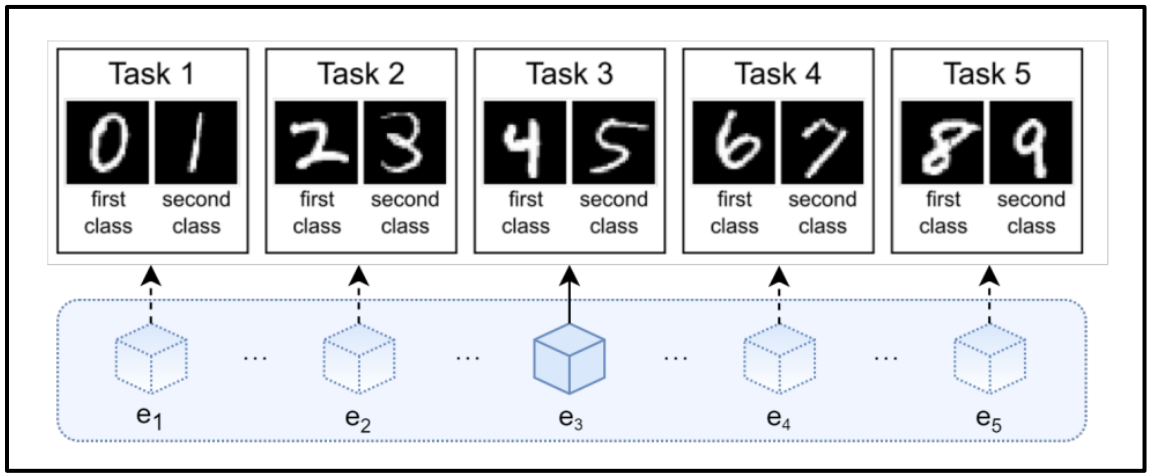


image source: lilianweng.io

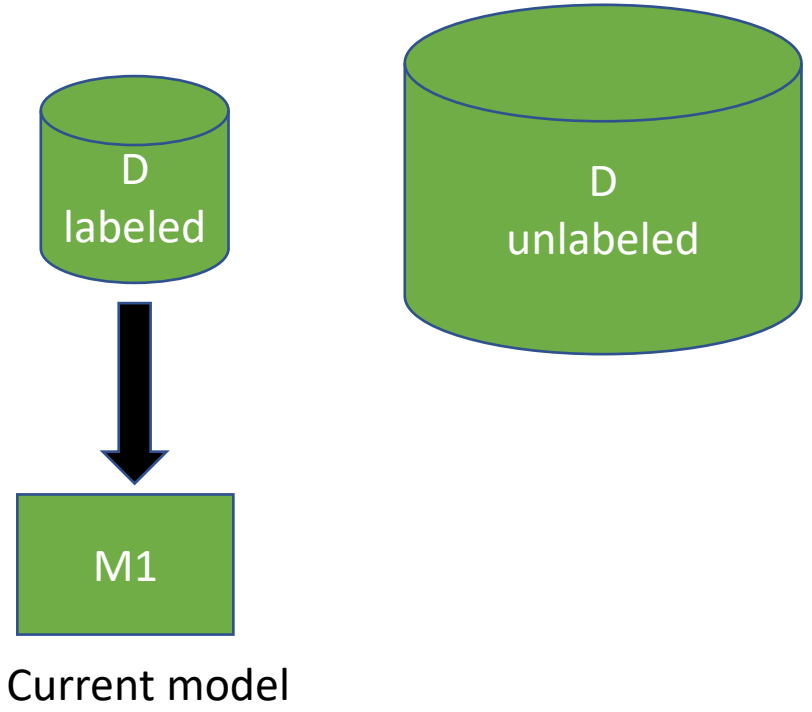
Continual Learning

- How do we incrementally train a deep neural network without forgetting the previous data?
  - Adaptation
  - Efficiency
  - forgetting



Active Learning

- We have a trained model, a labeled dataset, and a large unlabeled dataset
- Labeling is expensive
- How do we choose the best samples to label?



Out-of-Distribution Detection

- Deep neural networks are overconfident
- How do we detect examples that are outside the model's known domain?
- Can we compute confidence intervals for the network's output?

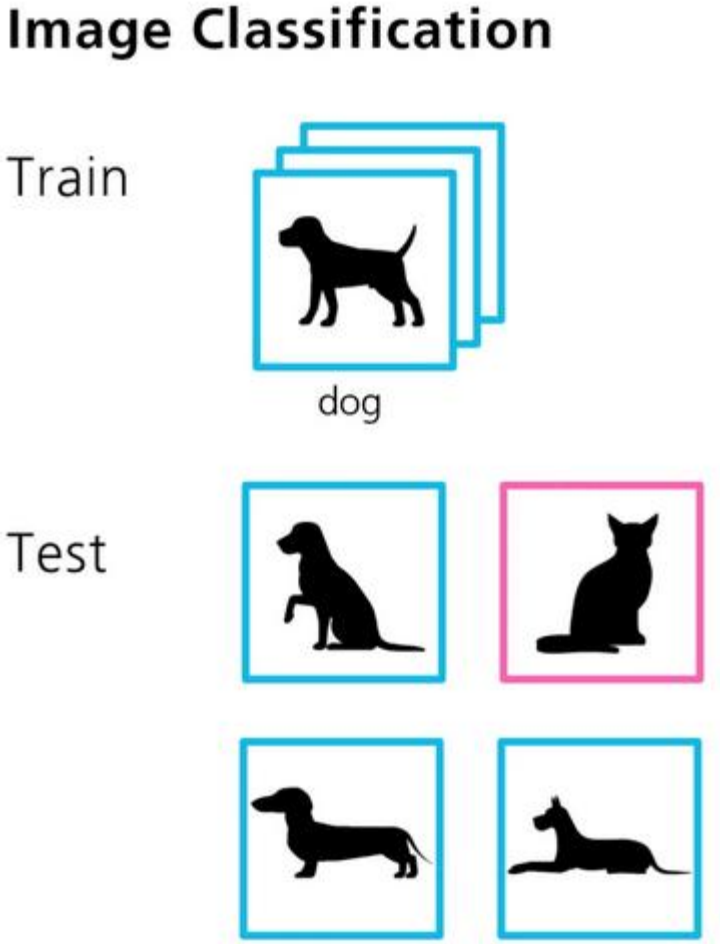
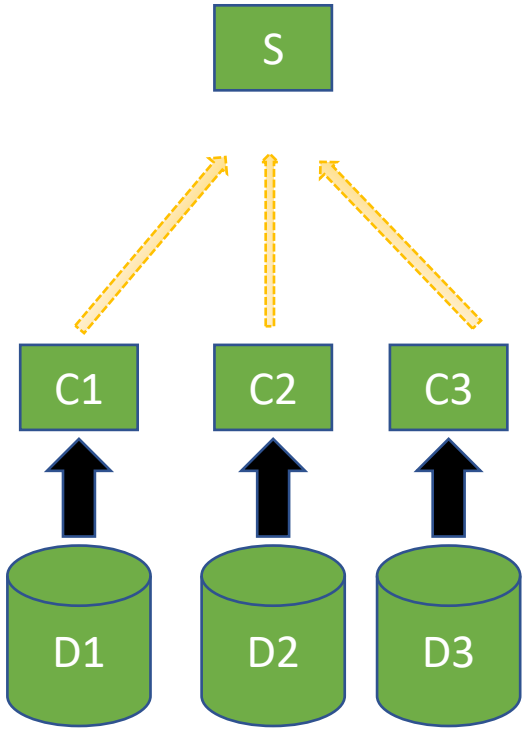


image source: https://safe-intelligence.fraunhofer.de/en/articles/out-of-distribution-detection-for-reinforcement-learning

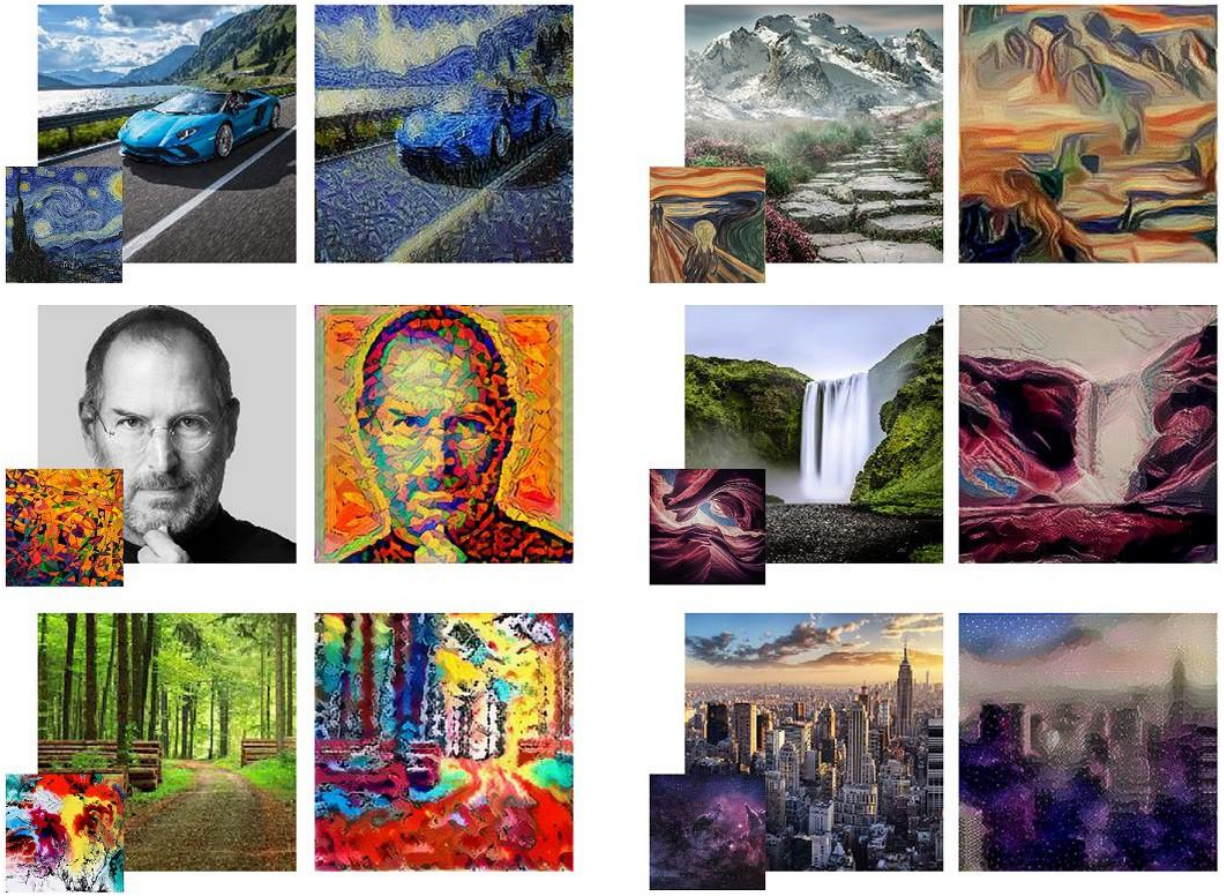
Federated Learning

- We have multiple devices
  - Each device is small
  - It sees only a small biased dataset
  - Data is private and it cannot be shared
- How do we train a single model without sharing the data?
  - We want to minimize the communication overhead too



Examples

Style Transfer

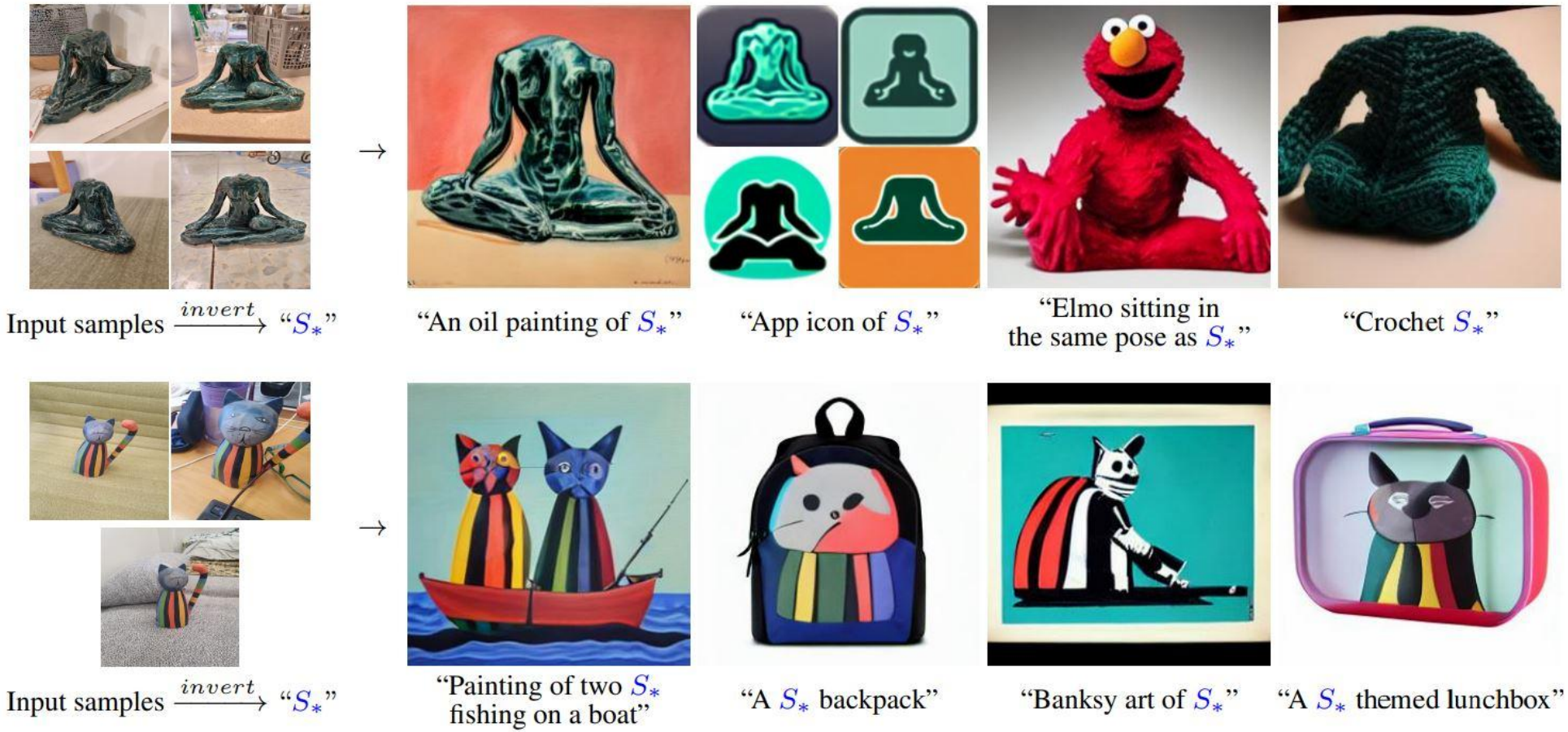


Example

- From https://lambdalabs.com/blog/how-to-fine-tune-stable-diffusion-how-we-made-the-text-to-pokemon-model-at-lambda
  - Starting from stable-diffusion v1.5
  - 2xA6000 GPUs, 15,000 steps, ~6 hours of training, at ~$10
  - almost a thousand images, automatically captioned

Girl with a pearl earring, Cute Obama creature, Donald Trump, Boris Johnson, Totoro, Hello Kitty

Textual Inversion



https://textual-inversion.github.io/

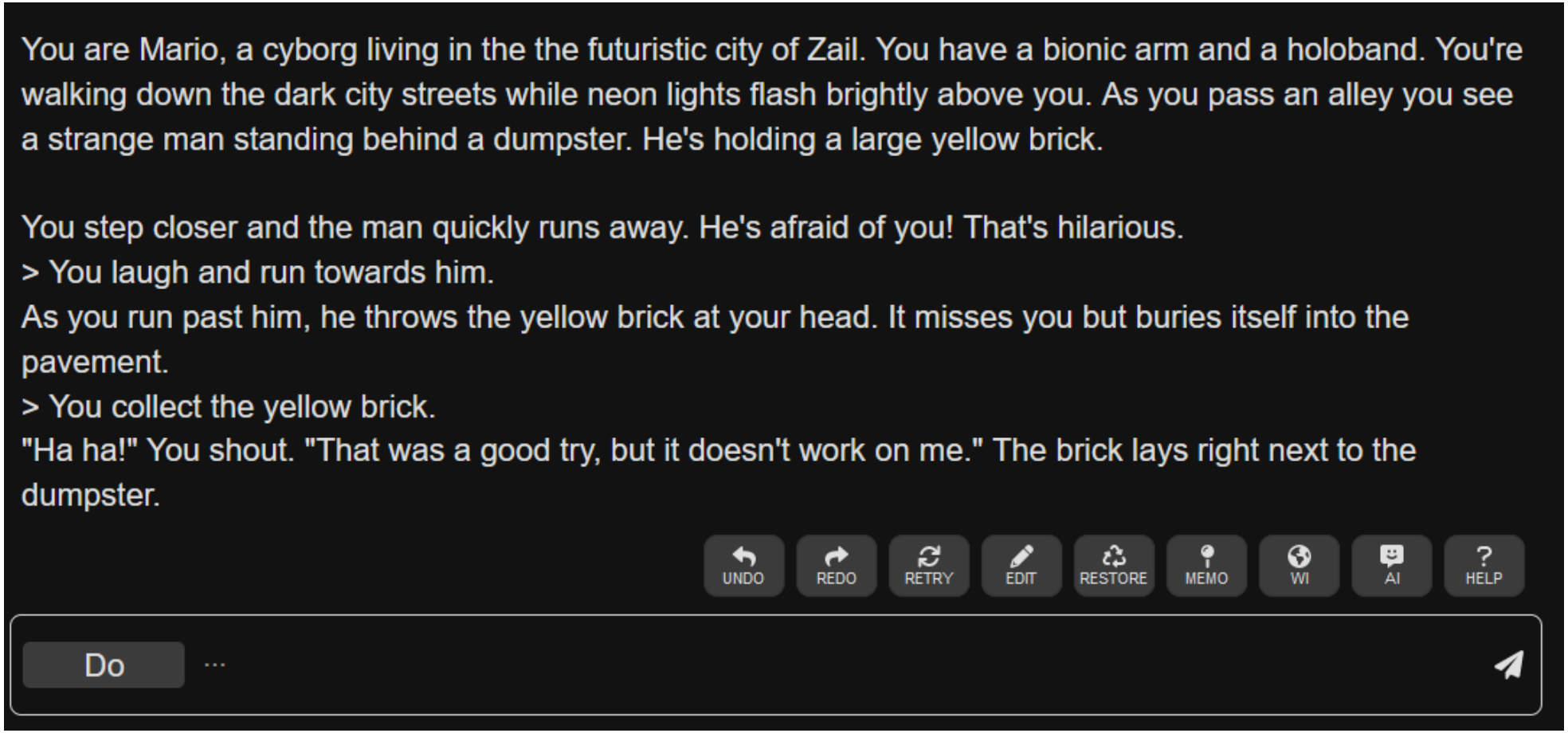
DreamBooth – Subject-Driven Generation

Figure 1. With just a few images (typically 3-5) of a subject (left), *DreamBooth*—our AI-powered photo booth—can generate a myriad of images of the subject in different contexts (right), using the guidance of a text prompt. The results exhibit natural interactions with the environment, as well as novel articulations and variation in lighting conditions, all while maintaining high fidelity to the key visual features of the subject.

CORe50

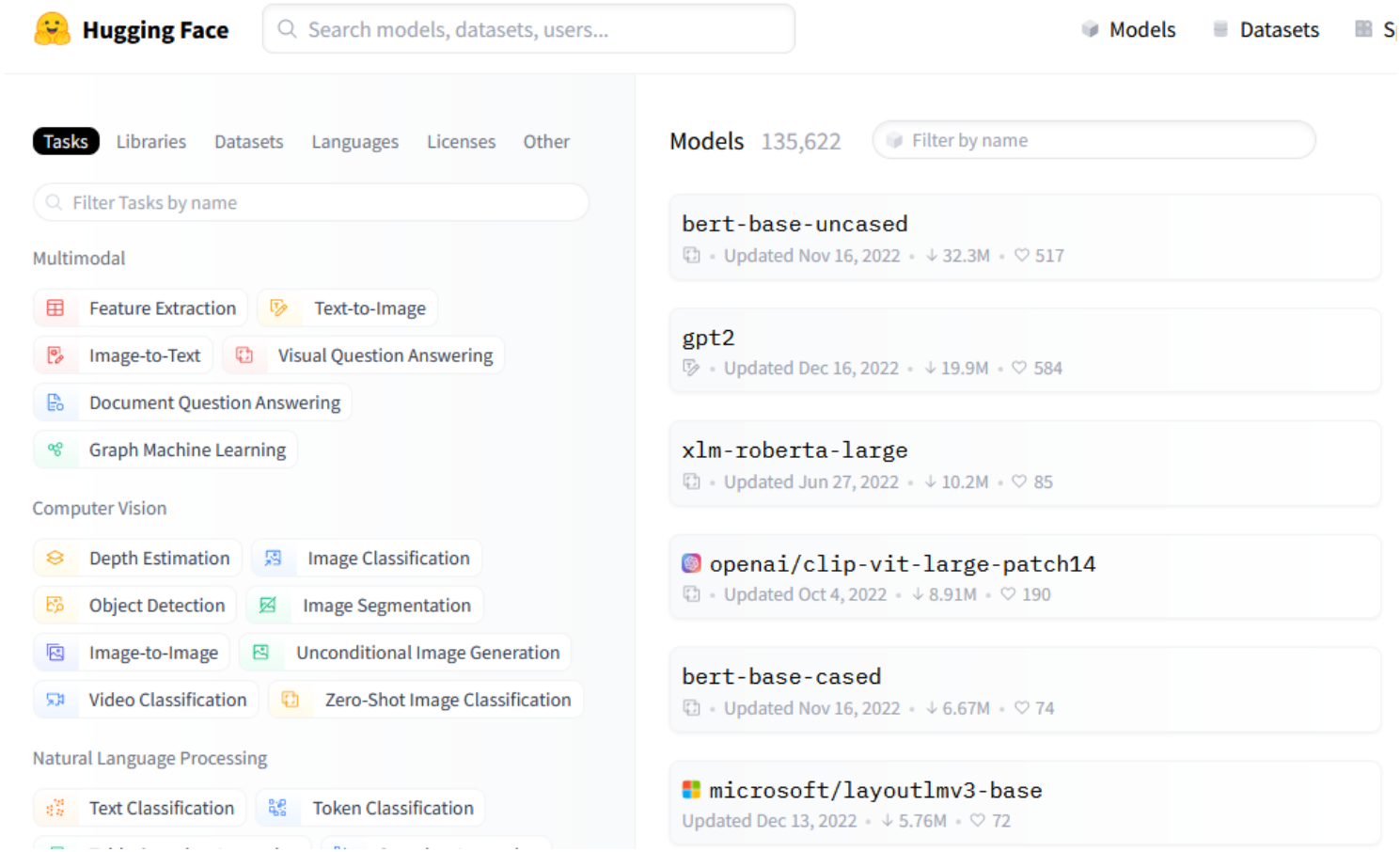
- Learning from small videos
- Non-iid minibatches
- Training on-device
- Can learn new classes over time

AI Dungeon



https://play.aidungeon.io

Hugging Face Models



https://huggingface.co/models

Course Structure

Objectives

Learn how to train ML models with streaming data:

- Nomeclature, definitions, problem settings
- Understanding the challenges of continual learning
- General solutions
- Practical applications of transfer/continual learning

Focus on:

- 1D time series data (first module)
- DNN for Computer vision

Expected Outcomes

Methodology:

- Gain an intuitive understanding of the different types of domain shifts and the problems they cause
- Understand effective solutions to train on non-stationary data
- Ability to code such solutions (e.g. with PyTorch)

Applications:

- Ability to reuse pretrained models for different applications. Examples:
  - Image classification -> style transfer
  - GPT-language-modeling -> Dungeon master
- Knowledge about efficient models that can be trained on-device

Prerequisites

- Knowledge of **Machine Learning** fundamentals: Followed ML course
- Basic knowledge of **deep learning**:
  - You have trained a simple DL model at least once (e.g. CNN on MNIST with Keras)
  - Suggested: ISPR course.
- Basic programming experience in Python
- **Math**: anything that you needed for the ML/ISPR courses

None of these requirements are strict but they help a lot. If in doubt just ask me.

Course Philosophy

- **You know how to train a machine learning model given a dataset. Most practical problems don't fit this simple scenario**.
  - Define problems that go beyond «offline training»
  - Highlight the challenges
  - Show the methods to solve them
- We will start with simple models and build on top of them incrementally (from linear models to ChatGPT, almost)
- Focus on practical approaches, using the theory to complement and explain the empirical findings and methodologies

Course Organization

Five Modules:

- Online Machine Learning
- Knowledge Transfer and Adaptation
- Open World, Uncertainty, and Robustness
- Deep Continual Learning
- Application and Frontiers

Each module builds on top of the previous ones and focuses on different aspects of learning with nonstationary data.

Module – Online Machine Learning

How do we learn a simple machine learning one sample at a time efficiently?

- Learning one sample at a time
  - 1D/small-dimensional time series data
  - Simple models: linear classifier, knn, decision trees…
- Concept drift
- Online classification models
- Ensemble methods
- Practical examples with River

Module – Knowledge Transfer and Adaptation

How can we reuse a pretrained DNN for a novel task?

- Deep learning with PyTorch
- Multi-Task Learning
- Transfer Learning and Domain Adaptation
- Self-supervised learning and large pretrained models
- Meta-learning, metric learning, few-shot learning
- Practical examples with PyTorch

Module – Open World, Uncertainty, Robustness

Can we trust the model's outputs? How do we treat uncertainty and unknown unknowns?

- Adversarial Robustness and Poisoning
- Model Calibration
- Out-of-distribution detection
- Open World Learning

Module – Deep Continual Learning

How can we retrain a DNN on a stream of data without forgetting?

- Continual Learning: problem definition, settings, evaluation
- Catastrophic forgetting
- Methodologies
  - Replay
  - Regularization methods
  - Architectural strategies
- Practical examples with Avalanche

Module – Applications and Frontiers

- Large Language Models and Continual Pretraining
- Model Patching
- Personalized Diffusion Models
- Invited seminars about research and applications in CL

Course Instructor

Antonio Carta

- Email: antonio.carta@unipi.it
- Office: Room 293
- Office hours: send me an email

Course Schedule

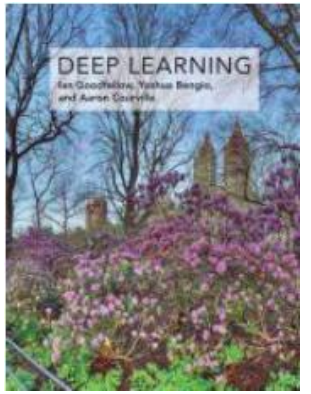
- Available on the master course website
  - Monday 11-13: L1 room
  - Thursday 16-18: M1 room
- 24 lectures
- In-person lectures

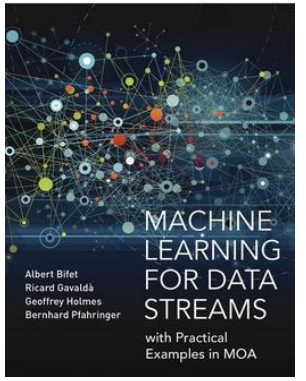
Homepage

- The reference page is on Moodle: Corso: Continual Learning (a.a. 2025-2026) | INF - e-learning - Dipartimento di Informatica
- Course information
  - Slides
  - Articles and references
- Teams is also available but we will use it only for general communication

References

- There is no single book
  - Deep learning preliminary knowledge:
    - Deep learning book, Goodfellow et al. (free pdf)
  - OML Module:
    - Machine Learning for Data Streams with Practical Examples in MOA. Bifet at al.
      - https://direct.mit.edu/books/book/4475/Machine-Learning-for-Data-Streamswith-Practical
  - I will give you articles and references for the other modules
  - Some lecture notes are also in preparation





- All the books and references used are open access and available for free
- Detailed references at the end of each lecture

Resources – ContinualAI

- ContinualAI is a non-profit organization with several resources about continual learning
- Course by Vincenzo Lomonaco: https://course.continualai.org/
- Avalanche: PyTorch-based library for continual laerning
- Seminars and lectures on the ContiualAI youtube channel
- Others:
  - ContinualAI Wiki: a shared and collaboratively maintained knowledge base for Continual Learning: tutorials, workshops, demos, tutorials, courses, etc.
  - Continual Learning Papers: curated list of CL papers & books with meta-data by ContinualAI
  - ContinualAI Forum + Slack: discussions / Q&As about Continual Learning
  - ContinualAI Research Consortium: networks of Top CL Labs across the world.

Software Library – River

- A python library for online machine learning
- API similar to scikit-learn
- We will use it for the first module
- https://riverml.xyz/0.14.0/

Software Library – PyTorch

- Deep Learning library
- We will use it for the deep learning code

PyTorch

Software Library – Avalanche

- PyTorch-based library for continual learning with deep neural networks
- We will use it in the Deep Continual Learning module
- Developed by ContinualAI and some people here at UNIPI, so feel free to ask any questions
- Docs: https://avalanche.continualai.org/
- Baseline experiments: https://github.com/ContinualAI/continual-learning-baselines

Avalanche

powered by

ContinualAI

Exams

- **Final Project + Oral Exam**
  - Final project: either a **written report** on a course-related topic or a **coding project**
  - Oral exam: a discussion of the final project and an examination on the course program.

- There will be **no midterm**. If you want to finish the project earlier, you can start the project before the end of the course.
- The **final vote** will be an average of the project and oral vote.
- Detailed **rules** on moodle.

Final Project – Software

- **Implementation of a method or empirical evaluation**
  - You can propose the topic. Some examples are available on moodle
- **Python/PyTorch/Avalanche are suggested** but not mandatory
- **You can (and should) reuse code** from public libraries and open repositories
  - Most projects are too big to develop from scratch, so starting from an existing codebase is allowed
  - Write explicitly what codebases you used as reference, if any
  - You should understand the code you are using and be able to explain it
  - Don't trust random repositories on github. If possible, look for primary sources (the original authors) or a trusted reference implementation (popular repository with solid evaluation)
- **Evaluated** on:
  - Correctness of the implementation
  - Your own understanding of the code and ability to explain it
  - Experiments: description and motivations for the experimental choices, such as benchmark used, hyperparameters, and comparison against the expected results from the literature

Final Project – Report

- A **written report/mini-review** on a course-related topic
  - List of **suggested topics** on moodle (coming soon)
  - You can **propose your own topic**, subject to my approval
  - Max 8 pages
  - More or less 5 papers not covered during lectures
- **Evaluated** on:
  - **Clarity** of the written report
  - **Understanding**: you should understand the reference material,
    - Can you identify the main points of the paper?
    - Find connection between them?
    - Open problems and limitations?

Oral Exam

- First part: final project discussion:
  - Software: brief description of the codebase and experimental results.
  - Report: presentation of the report
- Second part: open questions on the course material

Next Lecture

- Start of the Online ML module
  - What is OML
  - Prequential evaluation
- Notebook with example in River