General guidance

- Always write down everything, e.g. after you read a paper write some notes about it
- Use a list format for simplicity, later on you can write it better

(X) order of execution (suggested)

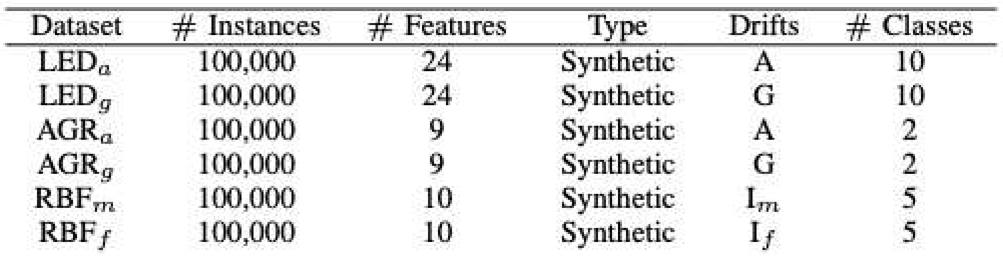
1. Preparation

(1) Problem setting

- What is the problem that we are trying to solve in here?
- What is the scope?
- What do we expect to happen?

(1) Practical

- Classification in data streams
- Prepare and execute some experiments with the contenders:
    - Hoeffding Trees (HT)
    - Extremely Fast Decision Trees (EFDT)
    - PLASTIC (also a tree algorithm, more recent than EFDT - 2024)
    - kNN (maybe use the moa book for reference https://moa.cms.waikato.ac.nz/book/)
    - Naive Bayes (maybe use the moa book for reference https://moa.cms.waikato.ac.nz/book/)
    - Adaptive Random Forest (ARF)
    - Streaming Random Patches (SRP)
- **Datasets**
    - The ones with capymoa links can be used from the API directly (i.e. from capymoa.datasets import Electricity)
    - Real datasets:
        - Electricity: https://capymoa.org/api/modules/capymoa.datasets.Electricity.html
        - CovtFD https://capymoa.org/api/modules/capymoa.datasets.CovtFD.html
        - Covtype (normalised) https://capymoa.org/api/modules/capymoa.datasets.CovtypeNorm.html
        - Airlines (needs to be downloaded and probably one-hot encoded): https://github.com/hmgomes/StreamingRandomPatches/blob/master/datasets/airlines.arff.zip
    - Synthetic data streams: https://capymoa.org/notebooks/04_drift_streams.html
        - Use the tutorial above to create the following synthetic streams
        - Each has 100k instances (number of features and classes is a characteristic of the generator itself, for these ones you might not be able to change, but if you can use the table below)
        - Each has 3 concept drifts (every 25k instances, i.e. 25k, 50k, 75k) which leads to 4 concepts
        - See the tutorial notebook on how to simulate the gradual and abrupt drifts, incremental drifts I_m and I_f depend on the RBF generator)
        - LED corresponds to https://capymoa.org/api/modules/capymoa.stream.generator.LEDGeneratorDrift.html
        - AGR corresponds to https://capymoa.org/api/modules/capymoa.stream.generator.AgrawalGenerator.html
        - RBF corresponds to https://capymoa.org/api/modules/capymoa.stream.generator.RandomRBFGenerator.html



Dataset	# Instances	# Features	Type	Drifts	# Classes
LED_a	100,000	24	Synthetic	A	10
LED_g	100,000	24	Synthetic	G	10
AGR_a	100,000	9	Synthetic	A	2
AGR_g	100,000	9	Synthetic	G	2
RBF_m	100,000	10	Synthetic	I_m	5
RBF_f	100,000	10	Synthetic	I_f	5

- Cumulative (table with the results) and windowed evaluations (plots for each dataset, so algorithm over time)
- Prepare the table with results and all plots, we might not end up using all plots later on, but it is good to have.

Related work

- **(2)** High level discussion of existing classification methods for data streams
    - Like the ones used in the benchmark above

- **(4)** The final paragraph of the related work should motivate the reader about your solution, something like: works A and B considered that X was true, but that limits it because of Y, so this motivates a solution that uses Transformers for…

(3) Background

- Data stream learning (assumptions and definitions) - What is classification for Data streams? Expectations and assumptions, the learning cycle, …
- Sequences vs. Time series vs. Data Streams (different assumptions, implications)
- Concept drift (what it is, detection, strategies for coping - proactively or reactively)
- Transformers and Neural networks: **Is using Transformers for sequences (like text) or time series (like temperature readings) the same as using it for data streams?**
- There might be some other things to be defined in here as well (coming from transformers)

2. (5) Transformers

Practical

- Use a simple MLP for the first experiments (there is a tutorial on capymoa on how to use PyTorch https://capymoa.org/notebooks/03_pytorch.html)
- Start trying with Transformers: compare against the results that we had from before

Updating the background:

- Recurrent neural networks (RNNs) and LSTMs might be discussed at least briefly because of their intersection with the problem setting
- What are transformers? Why they are useful? How they have been used

Updating the related work:

- Classical papers (like attention is all you need) and more recent papers using transformers for sequences / time series and maybe data streams (if there are any)