UNIVERSITY OF WARSAW

WYDZIAŁ MATEMATYKI, INFORMATYKI I MECHANIKI

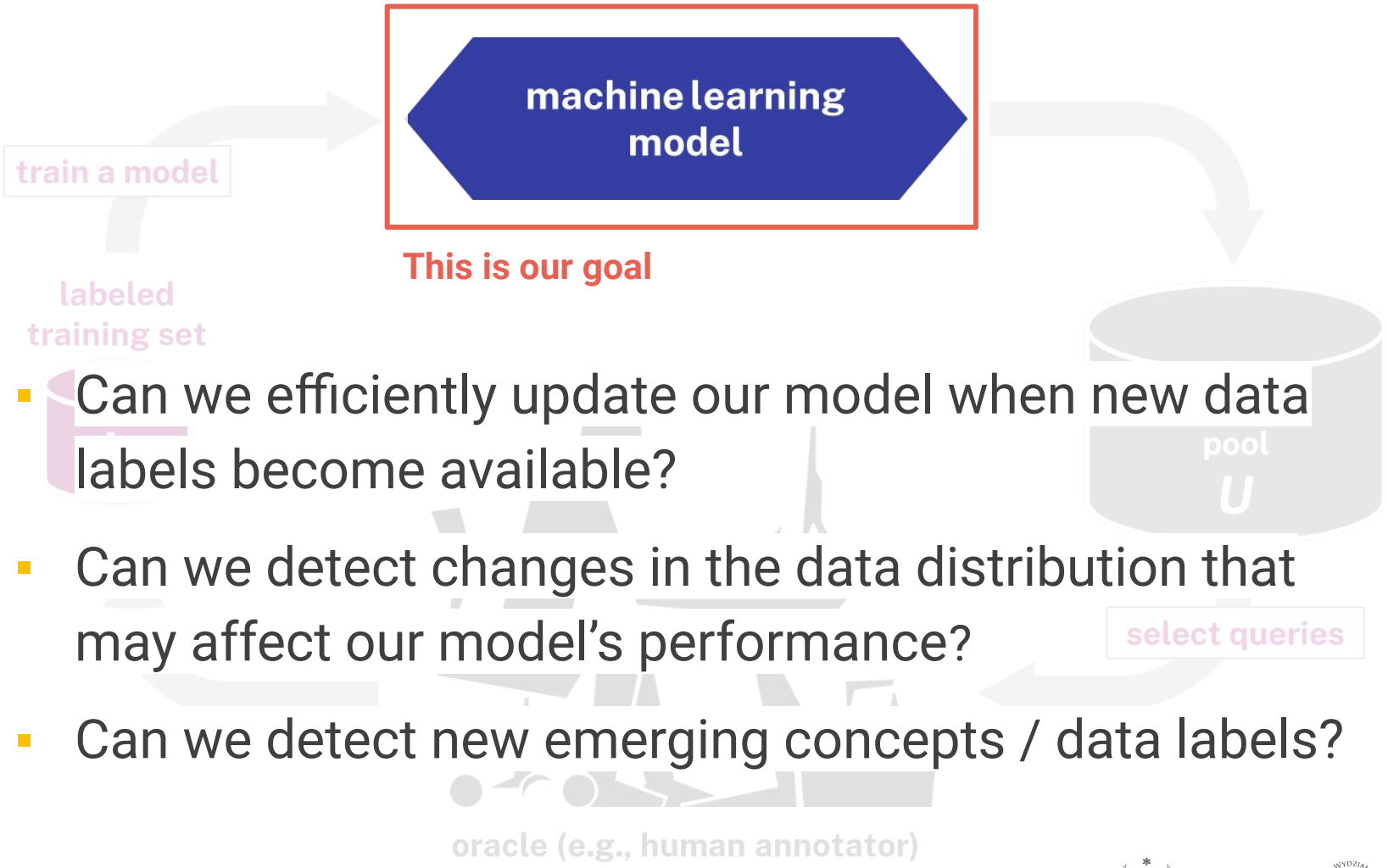
Incremental learning and concept drift

Andrzej Janusz
Daniel Kałuża

THE PLAN

- A recap of the previous lectures.
- The notion of incremental learning.
- Examples of incremental learning algorithms.
- Concept drift - a definition and detection methods.
- Active learning applications.
- Summary.

Previously - the active learning cycle

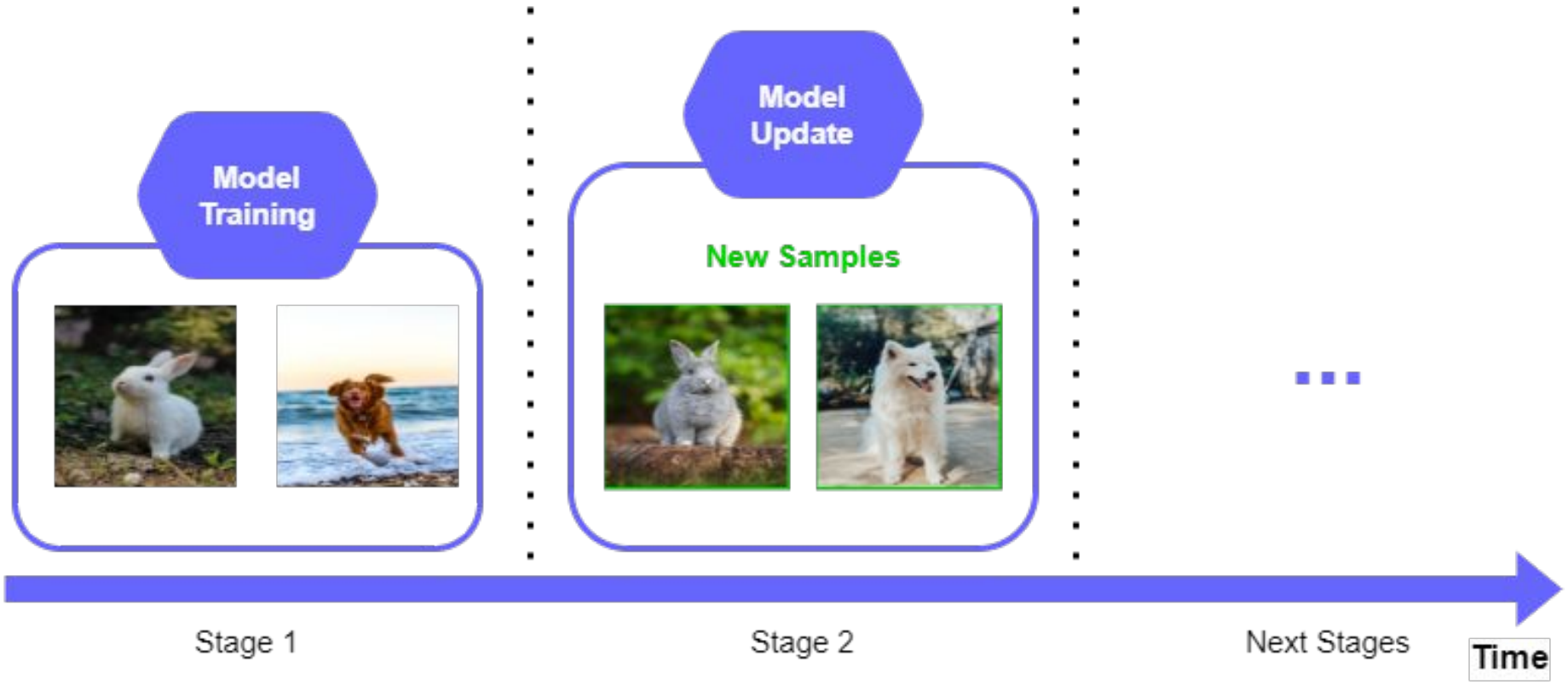


- Can we efficiently update our model when new data labels become available?
- Can we detect changes in the data distribution that may affect our model's performance?
- Can we detect new emerging concepts / data labels?

Incremental learning

- Supervised learning techniques that allow updating trained ML models using new portions of data.
  - Updating is usually faster than re-training from scratch.
  - Some models inherently allow incremental updates.
  - Various updating strategies may be applied.
- Incremental learning (IL) is sometimes confused with *online* or *real-time* learning:
  - IL algorithms do not have to work on data streams (but some of them can).
  - IL algorithms do not have to be very fast (work real-time) (but some of them are).

Incremental learning process



Inherently incremental learning models

- Some examples:
  - All type of a lazy prediction model.
  - Naive Bayes.
  - Incremental decision trees: ID4, ID5R.
  - Incremental SVM.
  - Neural networks.
    - Any model that can be trained using SGD…
  - Ensemble models (e.g., RF, GBM).

Lazy prediction algorithms

- Lazy algorithms do not produce a model.
  - New data is simply added to the reference pool.
    - k-Nearest Neighbors.
    - Lazy local rule-based classifiers.
- When available data set becomes large, some efficient indexing techniques are needed.
  - Incremental KD-trees, locally-sensitive hashing.
  - Efficient inverted attribute-value indexes.

Naive Bayes classifier

- Bayes theorem: $$p(Y|X) = \frac{p(Y)\,p(X|Y)}{p(X)}$$

  which translates into: $$\left(posterior = \frac{prior \times likelihood}{evidence}\right)$$

- If $X = (X_1, X_2, \ldots, X_k)$ and we naively assume that $X_1, X_2, \ldots, X_k$ are independent random variables, then:

$$p(X|Y) = p(X_1, X_2, \ldots, X_k|Y) = \prod_{i=1}^{k} p(X_i|Y)$$

- We get the naive Bayes (maximum a posteriori) prediction rule:

$$\hat{y}(x) = \arg\max_{y_i}\ p(Y = y_i) \prod_{j=1}^{k} p(X_j = x_j|Y = y_i)$$

Incremental updates of NB

- For each feature in our data set, we keep track of all value-decision pair counts.
- A stream of new data can be processed sample-by-sample.
  - We only need to perform a single increment for each count matrix.
- Typically, Laplace estimator is used to estimate the conditional probabilities.

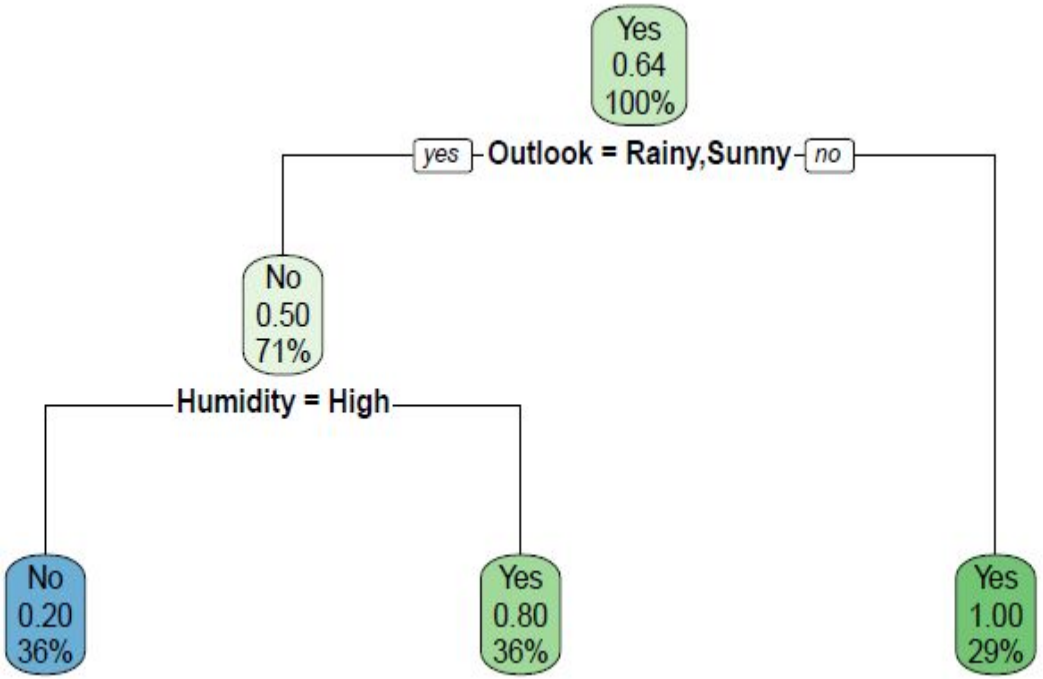
Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Counts table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

$$p(Weather = Rainy|Play = No) \approx \frac{3+1}{5+2}$$

Decision trees

- Recursive partitioning of data using splits on individual features.
- Can be used for various tasks and can handle all feature types.
- Typically, uncertainty reduction measures are used to select the splits.
- Pruning is often used to improve model generalization capabilities.

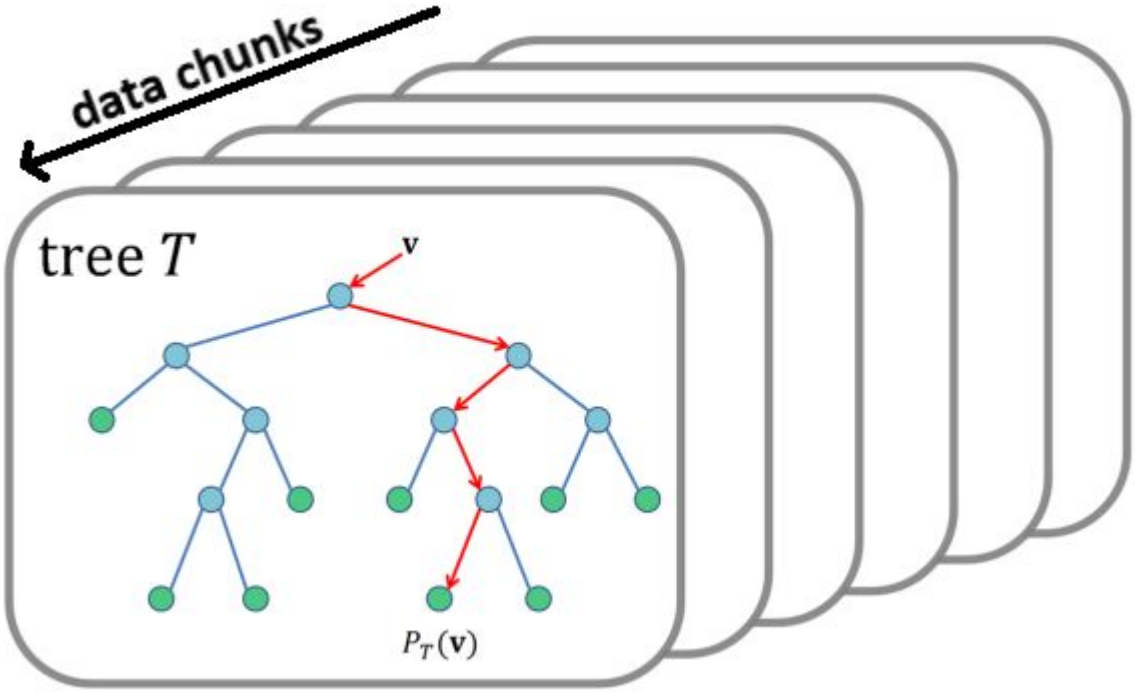


ID4

- The algorithm works for data sets with discrete features.
- At each tree node, we keep track of all decision-attribute-value tuple counts.
- For a new sample, we recursively update the tree nodes, starting from the root:
  - We update the decision-attribute-value counts.
  - If the current test attribute is no longer optimal, we change it to the one which provides the greatest information gain.
    - In such a case, we have to reconstruct the whole branch.
  - If the node is a leaf, we check whether it is necessary to expand it into a test node.
  - We go one level down in the tree.

Incremental random forest

- New data arrives in chunks.
- For each new data chunk, we grow K new trees using bootstrap samples.
- We may remove K oldest or worst performing trees to keep the total tree count fixed.



Learn++

- An algorithm inspired by the boosting approach.
- For each new data chunk D_k:
  - Boost weak learners for T_k iterations.
  - Accept a new learner only if its accuracy is greater than 0.5 (even for multi-class problems).
  - Add the boosted learner to the model.
- The final prediction rule is:

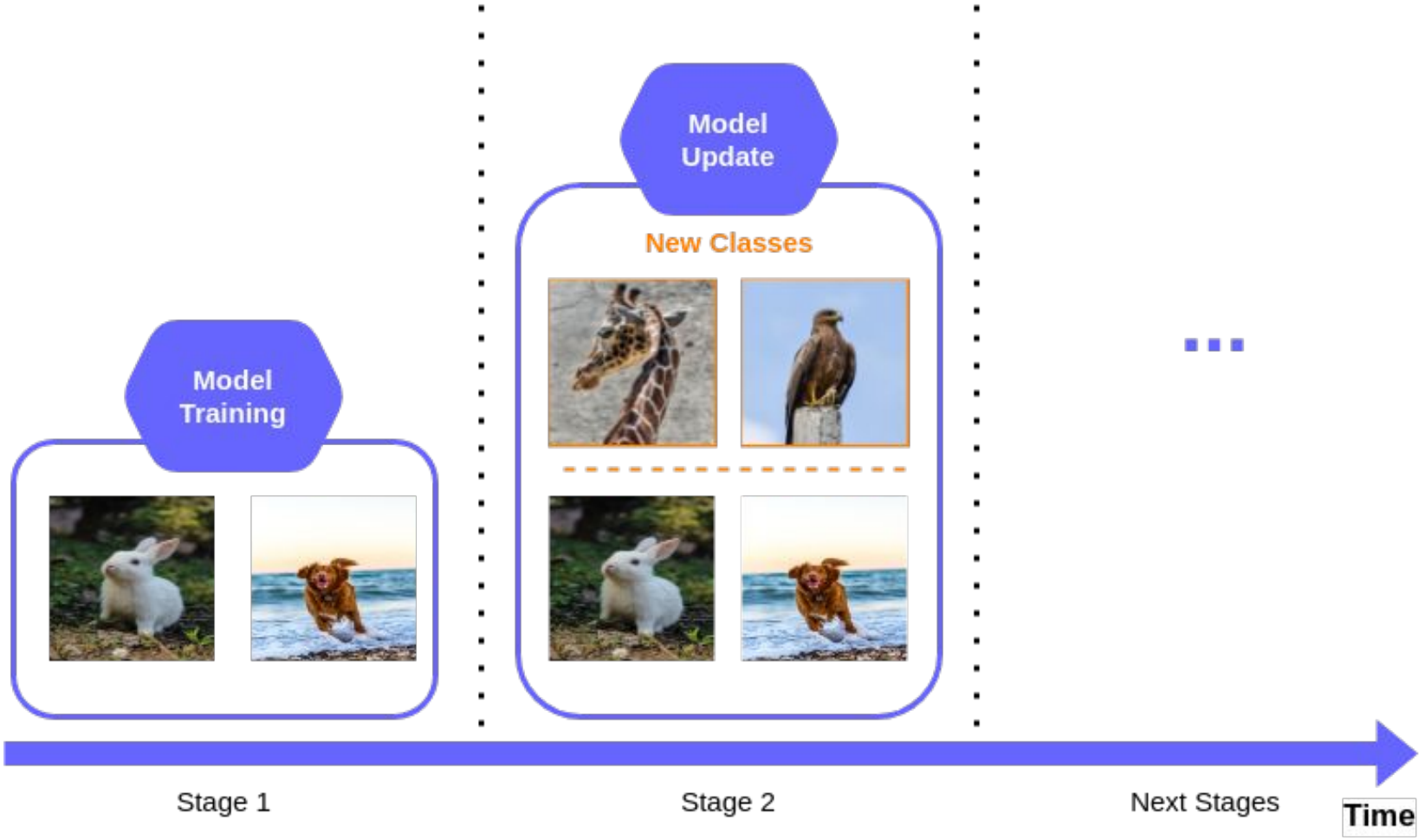
$$H_{final}(x) = \arg\max_{y \in Y} \sum_{k=1}^{K} \sum_{t:H_t^k(x)=y} \log\left(\frac{1 - E_t^k}{E_t^k}\right)$$

where E_t^k is an error estimate of the *t*-th weak learner trained on the *k*-th data chunk.

Incremental training of neural networks

- SGD is ideal for incremental training.
- Neural networks can be easily trained incrementally.
  - Assuming we have unlimited access to training samples, we don't need to see any sample twice…
  - In practice, however, we iterate through data many times.
  - If we want to train the network only using new data, we need to carefully choose the learning rate.
  - Typically, adding some older samples helps.
- In some scenarios, it might be necessary to modify the architecture of the network.
  - For instance, some additional filters in convolutional layers might be needed to increase the network capacity.

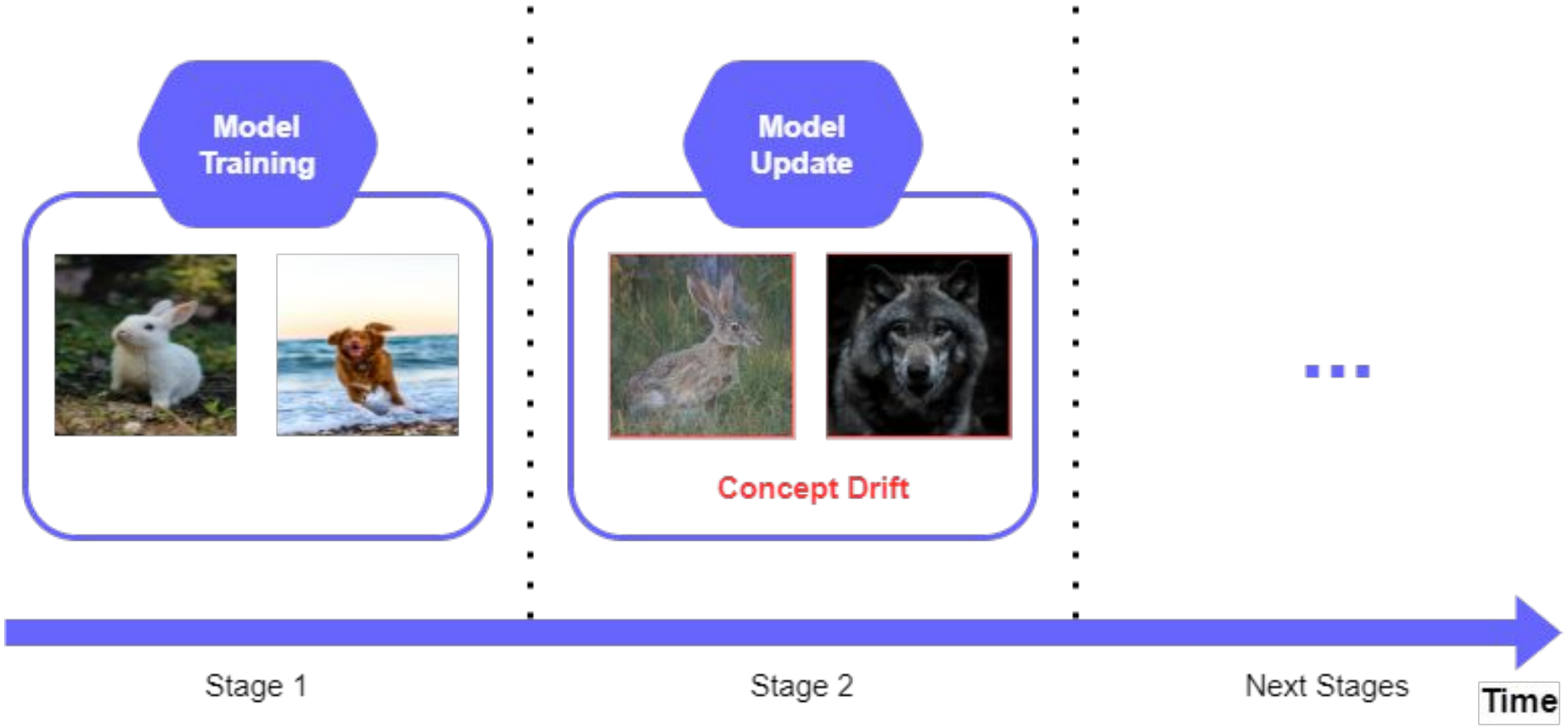
Learning new classes



Concept drift

- The distribution of data may change in time.
  - If the distribution is significantly different from the original training data, the assumption about IID sampling is invalid.
  - As a result, the distribution of target variable might be different.
  - New samples might be dissimilar to previous training cases.
- The meaning of concepts (e.g. classes) in the data may change in time.
  - Old “decision rules” becomes invalid or obsolete.
  - The same samples would have different targets if we labeled them again.
- In any case, the model needs to adapt.

Adapting to concept drift



Practical examples of concept drift

- Concept drift is common for time series data.
  - Non-stationary time series.
  - Complex seasonality patterns (or insufficient data history).
  - Some examples:
    - IJCRS'15 Data Challenge: Mining Data from Coal Mines.
    - ISMIS'17 Data Mining Competition: Trading Based on Recommendations.
    - FedCSIS 2020 Challenge: Network Device Workload Prediction.
- Concept drift might be caused by other factors.
  - Different sets of patients/subjects.
  - Some examples:
    - AAIA'15 Data Mining Competition: Tagging Firefighter Activities at a Fire Scene.
    - IEEE BigData 2020 Cup: Predicting Escalations in Customer Support.

Concept drift detection

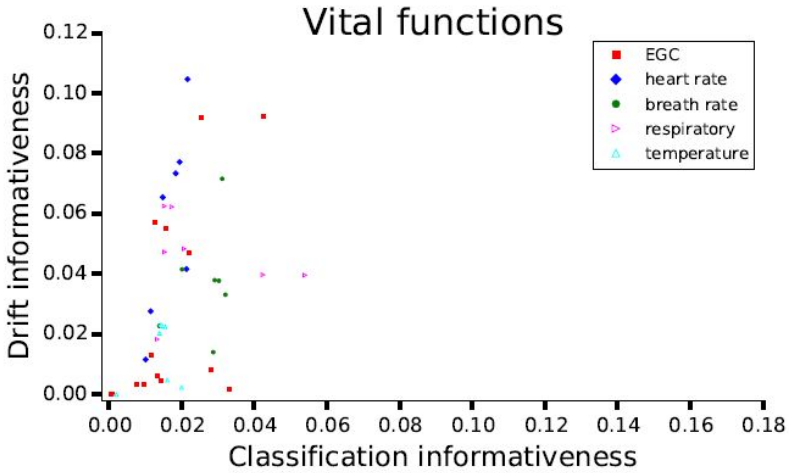
- Keep an eye on the distribution of your data.
  - Comparisons of basic data statistics from different periods.
    - Time window-based data monitoring.
    - Cross-correlation checks of the monitored features.
  - Anomaly detection algorithms are very useful.
    - Periodic reports about the number of detected anomalies.
  - Monitoring of the target variable distribution.
- Keep an eye on the performance of your prediction model.
  - Periodic assessments of the quality of predictions.
  - Comparisons with the performance of previous model versions.

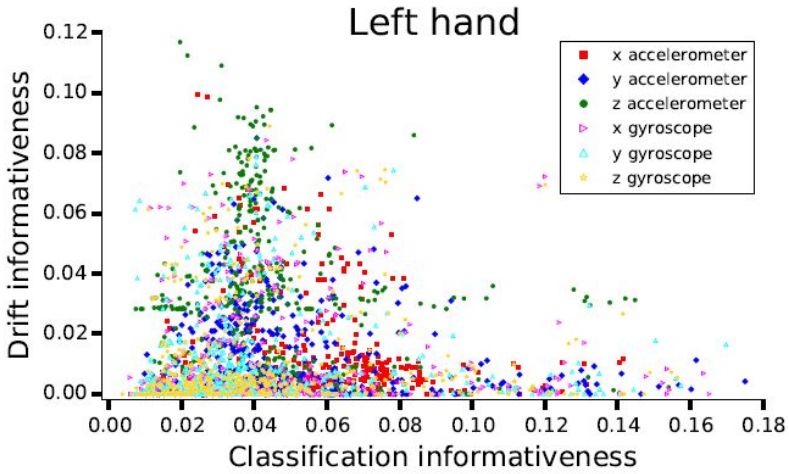
Triggering model updates

- Periodic (pre-scheduled) model updates.
  - Predictable costs and allocated resources.
  - Some updates might be unnecessary.
- Updates are triggered by the data monitoring mechanisms.
  - A proactive approach - may allow to avoid the drop in performance.
  - Updates are done only if they are clearly needed.
  - May not detect changes in concept “meaning”.
- Updates are triggered by the performance monitoring mechanisms.
  - The model is updated after “damage was done”.

Finding robust data representation

- Finding a good data representation can be the key to dealing with concept drift.
- Using features that are less prone to data drift will increase the robustness of the model.
- The robustness to data drift can be used as an additional criterion for the feature selection.





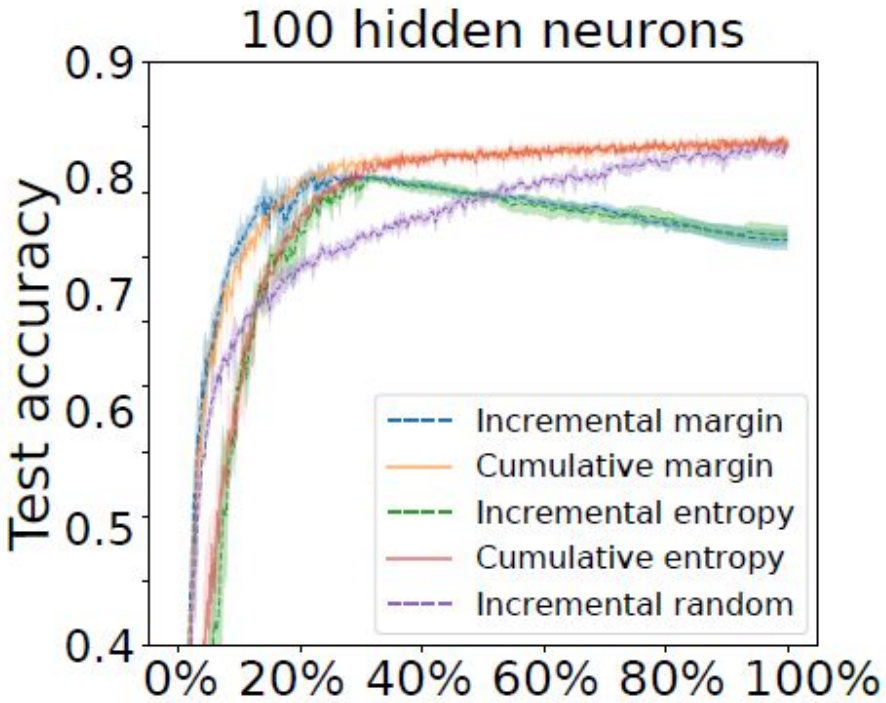
source: M. Boulle, Tagging Fireworkers Activities from Body Sensors under Distribution Drift

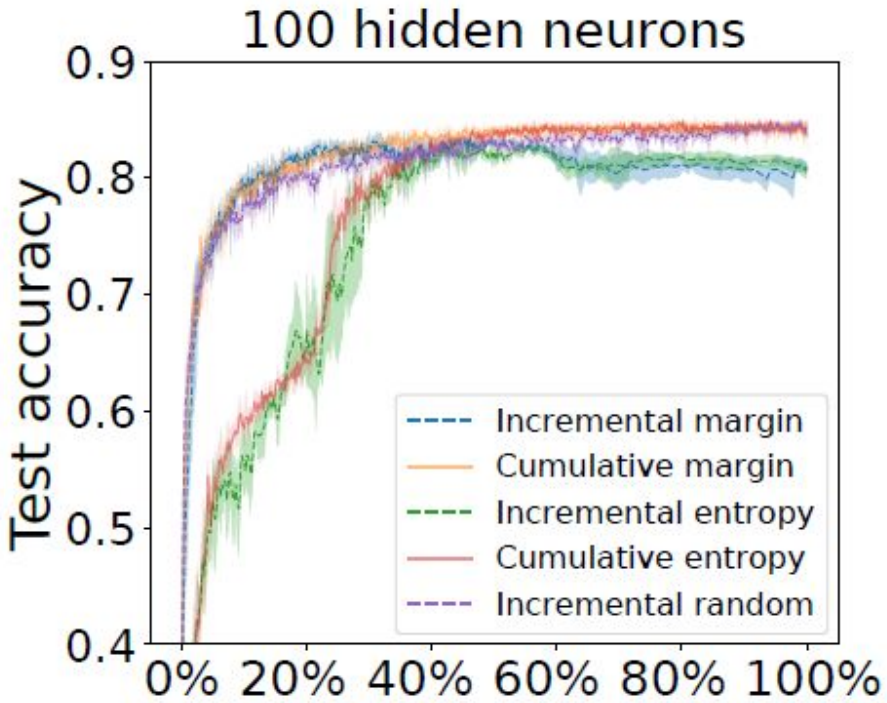
UNIVERSITY OF WARSAW

Active and incremental learning

- Incremental learning can be applied in the active learning cycle.
  - It significantly reduces the training time.
  - Might be necessary for some algorithms - like the expected error reduction (EER) method.
  - However, often the training time is not a big issue…
- Re-training models in each iteration of the AL cycle is often a better strategy if the data pool is fixed.
  - Incremental learning is the best choice when the concept drift is expected.
  - Incremental learning can deal with new data classes.

Experiment: MNIST and Fashion-MNIST data



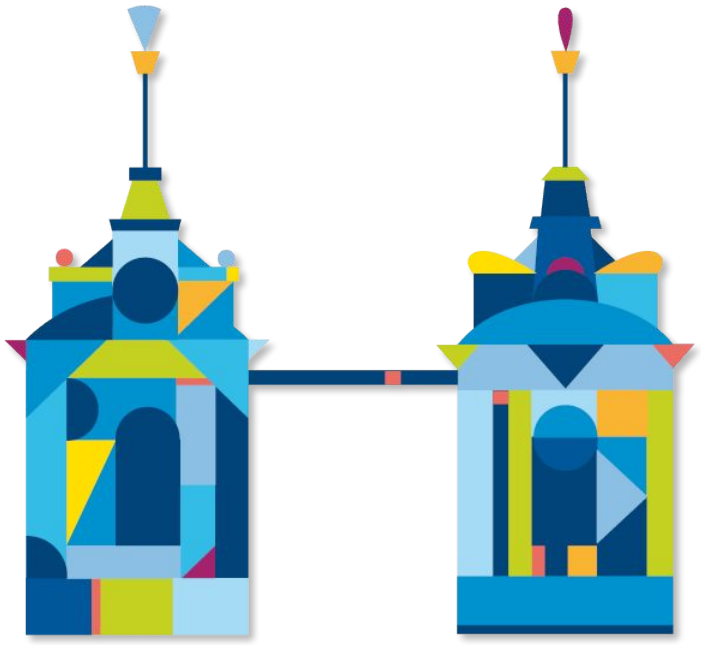


Summary

- We discussed advantages and potential issues related to incremental learning.
- We talked about “incremental versions” of several popular machine learning algorithms.
- We considered the problem of learning under the concept drift.
- We discussed standard drift detection methods.
- We summarised the incremental model updates in the context of active learning.

Literature:

1. C. Salperwyck, M. Boullé, V. Lemaire: Concept drift detection using supervised bivariate grids. In 2015 International Joint Conference on Neural Networks, pp. 1-9, (2015).
2. R. Polikar, L. Upda, S.S. Upda and V. Honavar. Learn++: an incremental learning algorithm for supervised neural networks. In IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 31, no. 4, pp. 497-508, (2001).
3. Y. Cai, W. Xu, F. Zhang: ikd-Tree: An Incremental K-D Tree for Robotic Applications. CoRR abs/2102.10808 (2021).
4. S.S. Sarwar, A. Ankit, K. Roy: Incremental Learning in Deep Convolutional Neural Networks Using Partial Network Sharing. In IEEE Access, vol. 8, pp. 4615-4628, (2020).
5. J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, G. Zhang: Learning under Concept Drift: A Review. In IEEE Transactions on Knowledge and Data Engineering, vol. 31, no. 12, pp. 2346-2363, (2019).
6. J.D. Bossér, E. Sörstadius, M.H. Chehreghani: Model-Centric and Data-Centric Aspects of Active Learning for Deep Neural Networks. IEEE BigData 2021: 5053-5062. (2021).
7. Z. Zhou, J.Y. Shin, S.R. Gurudu, M.B. Gotway, J. Liang: Active, continual fine tuning of convolutional neural networks for reducing annotation efforts. Medical Image Anal. 71: 101997 (2021).



QUESTIONS OR COMMENTS?

a.janusz@mimuw.edu.pl

or

d.kaluza@mimuw.edu.pl