A Boosting Tree Based AutoML System with Concept Drift Adaptation

Meta_Learners:

Zheng Xiong, Jiyan Jiang, Wenpeng Zhang

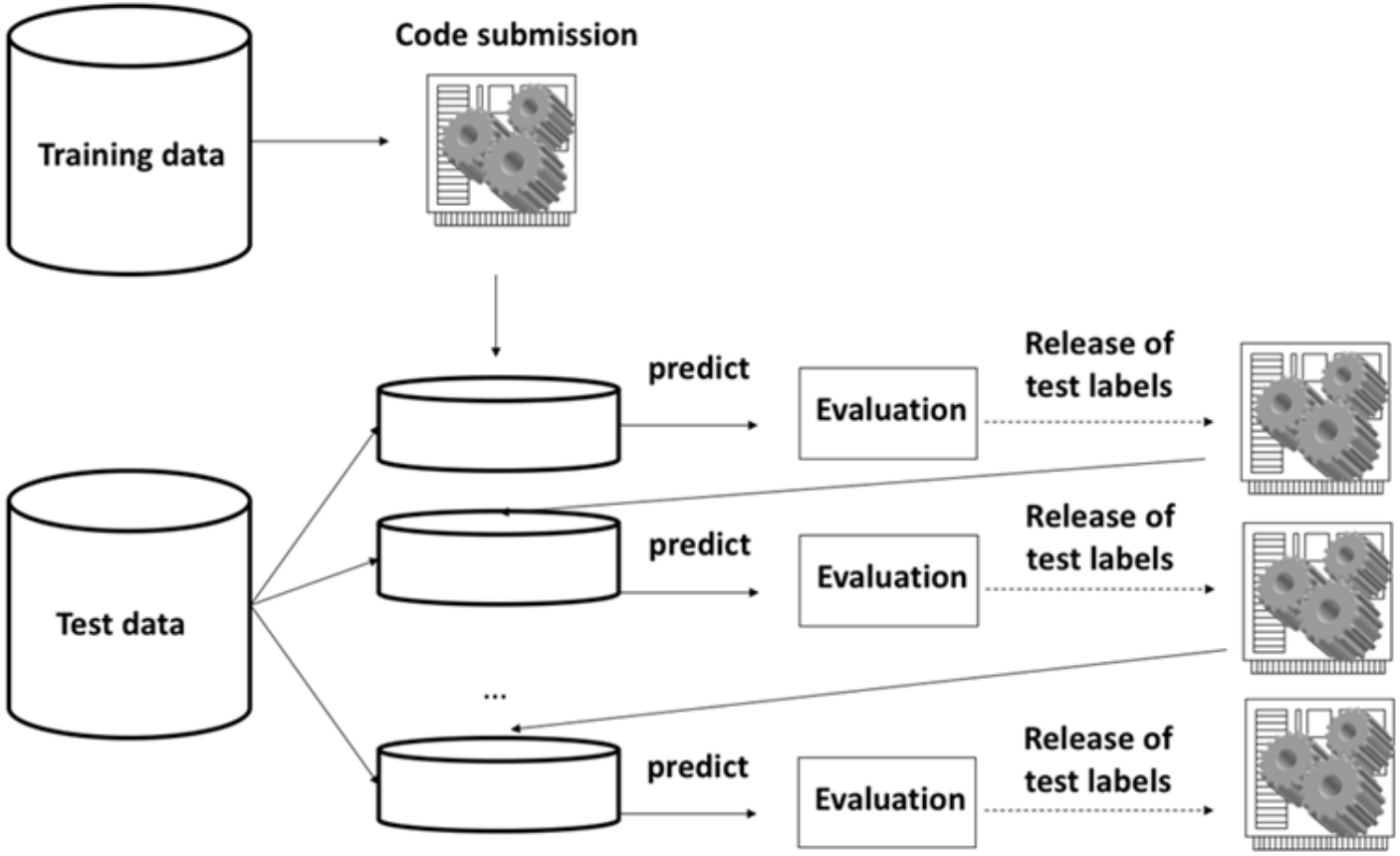
Advisor: Prof. Wenwu Zhu

Department of Computer Science, Tsinghua University, Beijing

Outline

- Problem Statement
- System Framework
- Automated Feature Engineering
- Concept Drift Adaptation
- Conclusion

Problem Statement



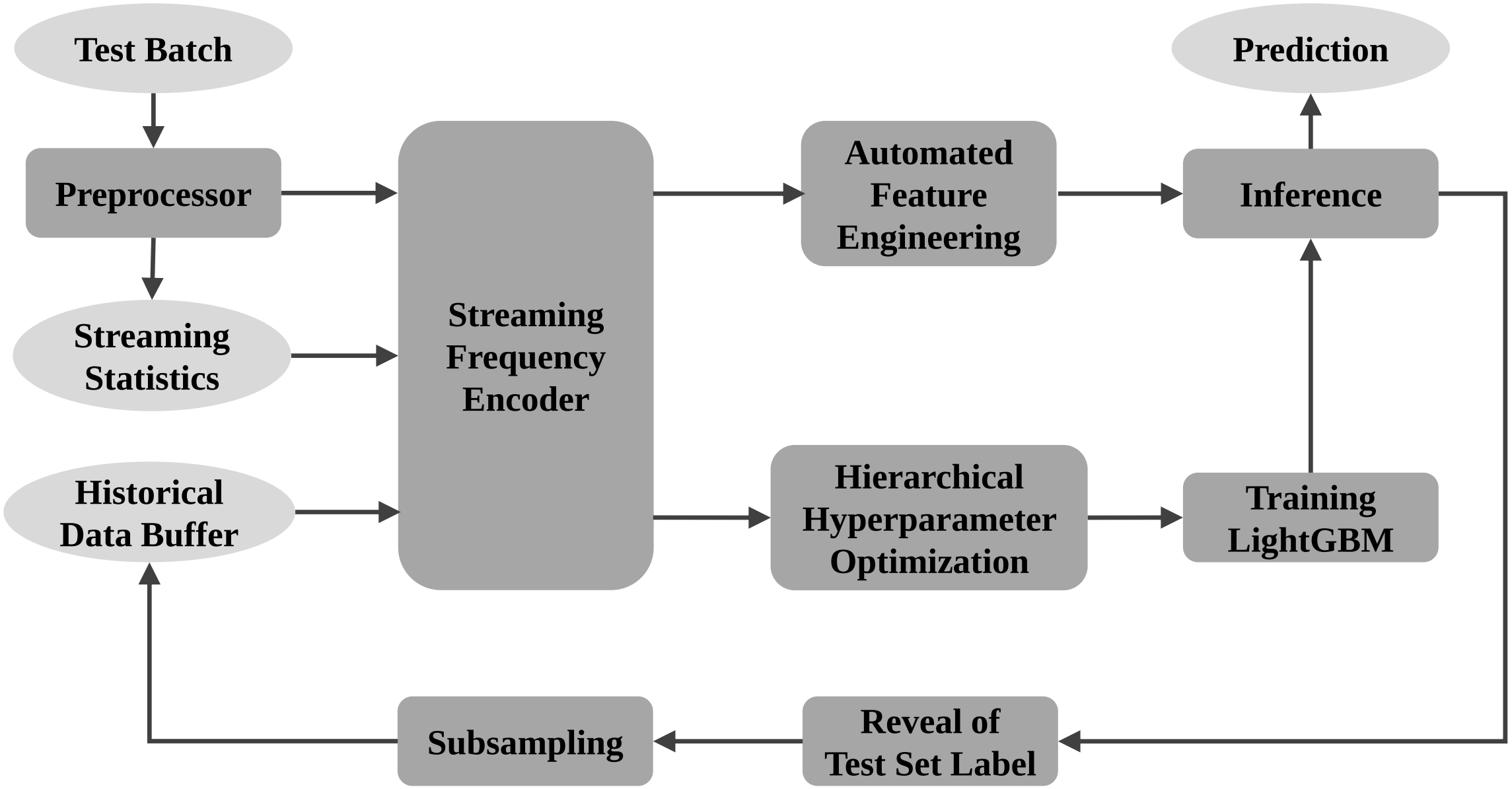
AutoML: the final submission of the feedback phase is blindly tested on 5 unseen new datasets without human intervention

Concept drift: data comes in stream with data distribution changing between batches

System Framework

- Use gradient boosting tree (GBT) as the classifier across different datasets
- Perform automated feature engineering to improve model performance
- Tackle concept drift by retraining and streaming co-encoding
- Optimize hyperparameters hierarchically to improve the efficiency and robustness of the system

System Framework



Automated Feature Engineering

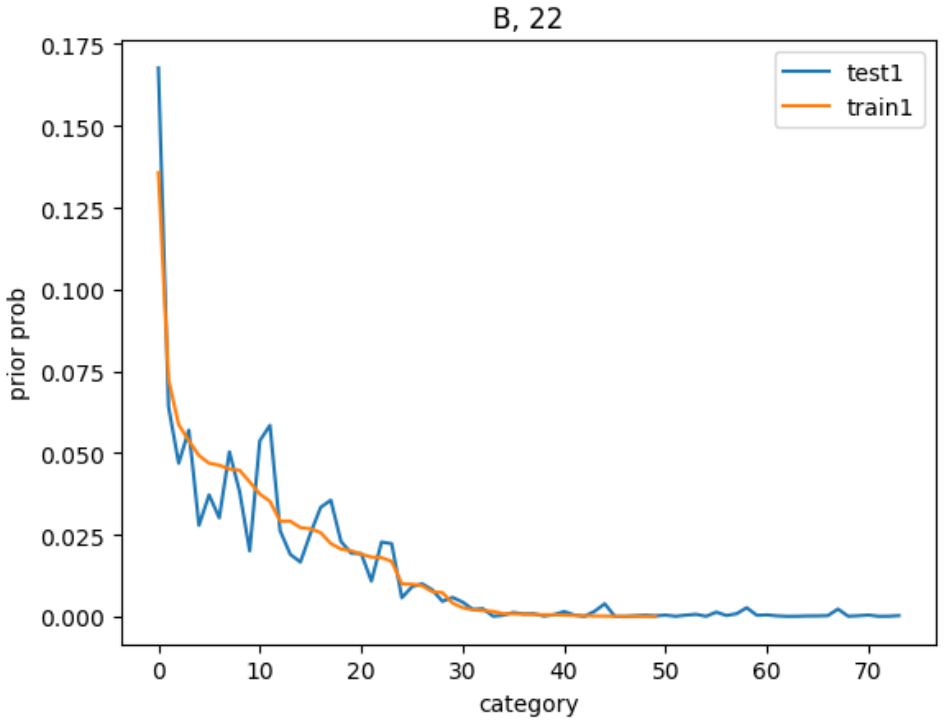
- **First-order feature engineering**
  - Frequency encoding of categorical features

- **High-order feature engineering**
  - Predefine a set of binary transformations based on prior knowledge
  - Apply each type of transformation on the original feature set to generate new features in an expansion-reduction fashion

Automated Feature Engineering

- **High-order feature engineering**
  - Predefined binary transformations:
    - Numerical-numerical: $+, -, \times, \div$
    - Categorical-numerical: num_mean_groupby_cat
    - Categorical-categorical: cat_cat_combine, cat_nunique_groupby_cat
    - Categorical-temporal: time_difference_groupby_cat
  - Key steps in the expansion-reduction strategy:
    - **Pre-selection**: select features used for feature generation based on prior knowledge
    - **Feature generation**: generate new feature with all feasible pairs of the pre-selected features
    - **Post-selection**: select generated features based on the performance and feature importance of a coarsely trained GBT model

Concept Drift Adaptation



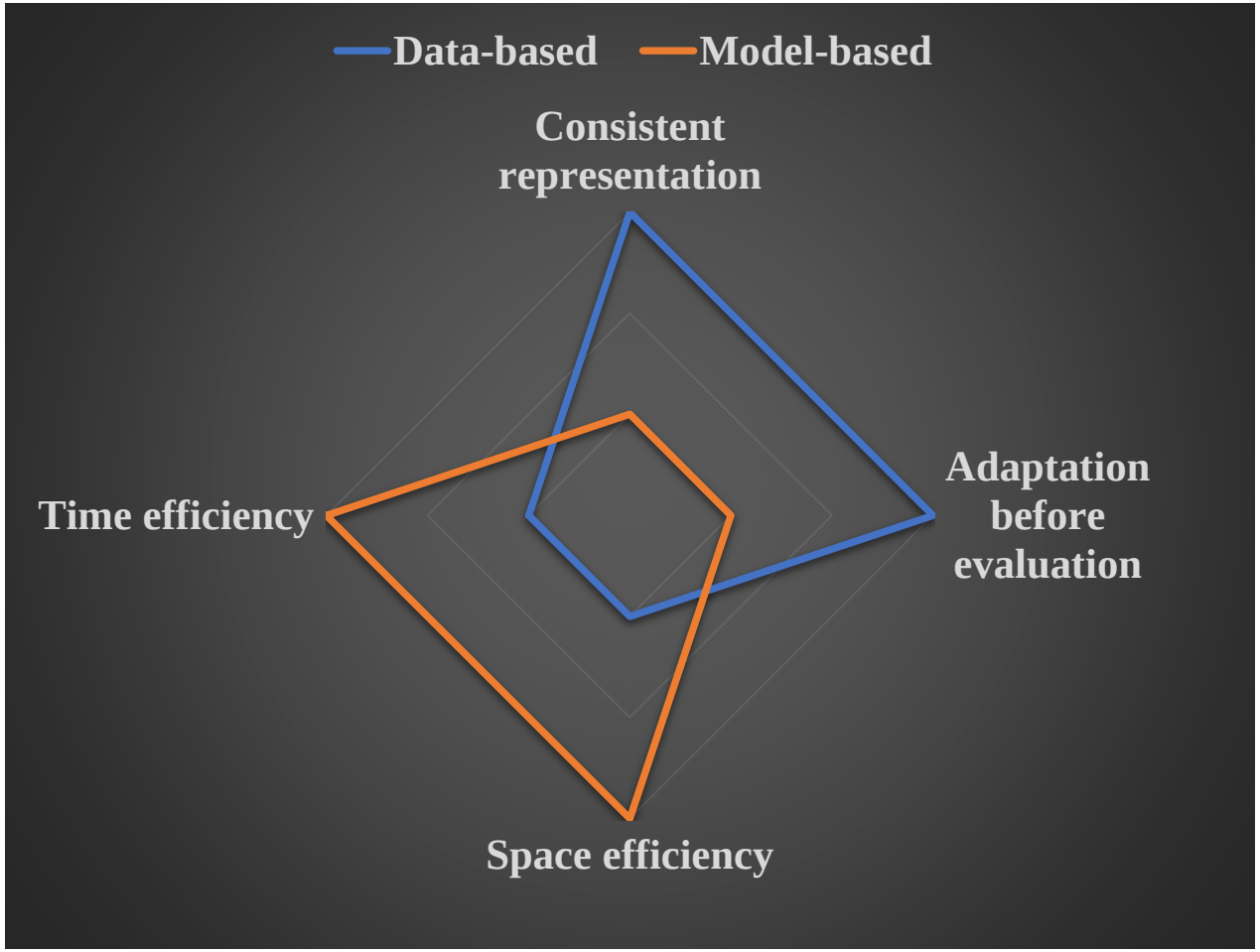
Concept drift in categorical features

- Many unseen new categories may appear in the test batch
- The frequency of existing categories may change significantly between batches

Concept Drift Adaptation

Different strategies for concept drift adaptation

- Data-based: retrain a new model with historical data
- Model-based: create an ensemble with models trained on previous batches



Concept Drift Adaptation

Our solution:

- Retrain a GBT model with the last N batches for each test batch
- Apply streaming co-encoding to achieve a consistent representation between training set and test set
- Strategies to improve space and time efficiency:
  - Data subsampling
  - Streaming statistics

Conclusion

- Propose a boosting tree based AutoML system with concept drift adaptation
- Design an efficient automated feature engineering strategy which significantly improves model performance
- Adapt to concept drift by retraining and streaming co-encoding
- Future work:
  - Explicit concept drift detection and adaptation
  - Generalization and scalability of automated feature engineering

Thank You!