# Heterogeneous Ensemble for Feature Drifts in Data Streams

Sebastian Döhler  Jan Kohlhase  Noah Kornelius Wöhler

Gottfried Wilhelm Leibniz Universität Hannover

## Introduction

Data stream classification needs to be real-time, efficient and be able to cope with high-dimensional, continuously arriving data. This problem occurs in many real-life applications like online stores or stock markets. A lot of modern technologies produce massive amounts of data that need to be processed to gain information from it. The discovered patterns from this data change over time. Two different kinds of changes exist that are commonly referred to as concept drifts. There are gradual drifts with small changes over time and sudden drifts that incur drastic changes. To deal with this, HEFT proposes a method to incorporate feature selection into ensemble learning to reduce computational complexity and adapt to the different types of concept drifts. For the feature selection, a modified version of the FCBF algorithm is used. A heterogeneous ensemble is created to diversify the classification. This is done because different classifiers can perform differently on distinct chunks of the data.

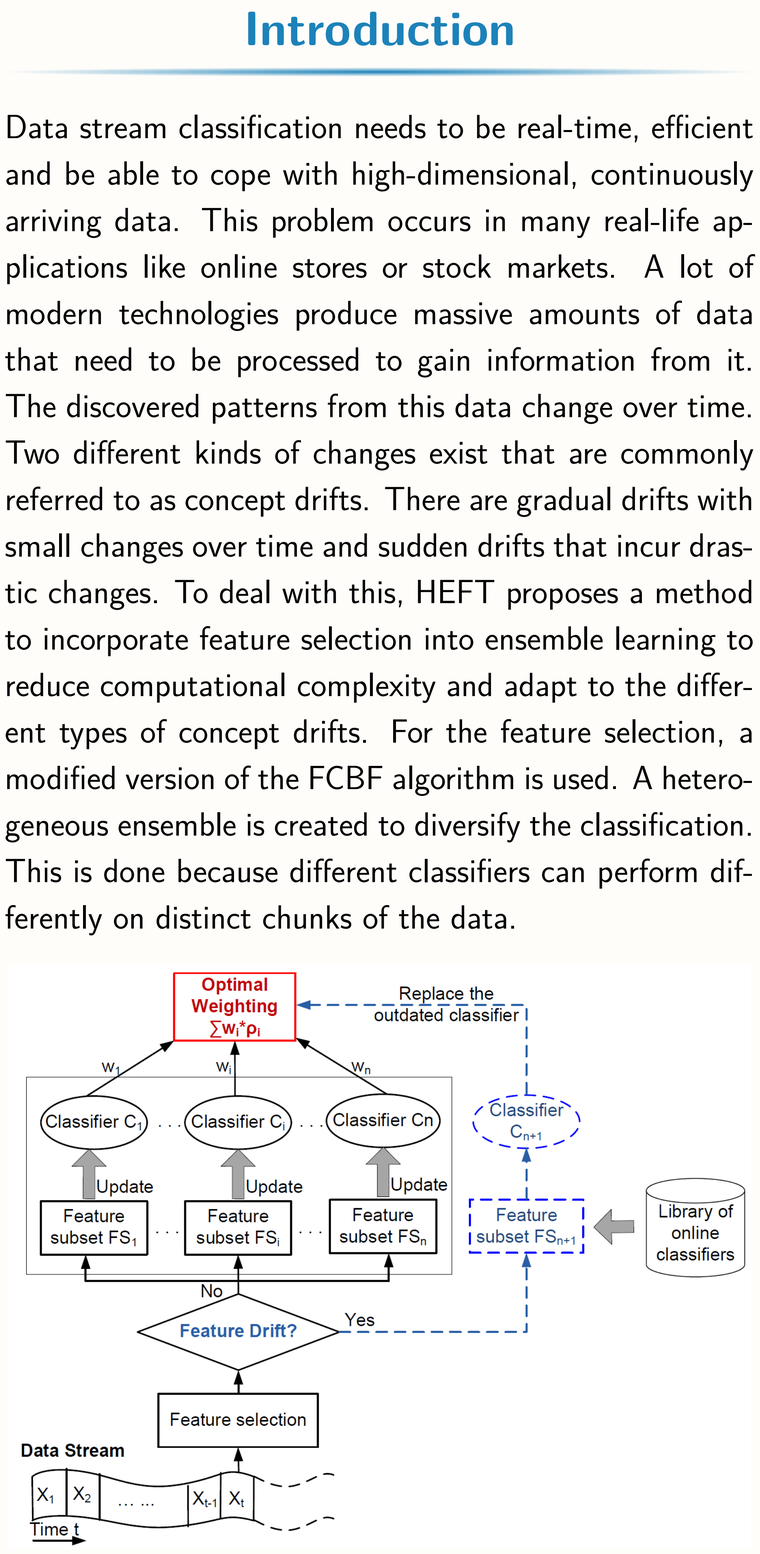


Figure 1: Overview of the algorithm.

## Method

The method first uses a feature selector to reduce the data to a feature space of lower dimensionality. By comparing the feature subspaces at two consecutive points in time, the algorithm detects feature drifts. If the current feature subspace differs from the previous, then there is a change in the underlying distribution. It then applies an ensemble block which is an easy and efficient method to improve the classification results. A small heterogeneous ensemble is used instead of a homogeneous one. The optimal weighting method assigns each classifier a weight with respect to its classification error. According to this weight, the final prediction is a weighted sum of all the classifiers' predictions. If there was a feature drift, then a newly trained classifier of the best performing type is added to the ensemble. If no feature drift occurred, then all the classifiers in the ensemble are partially fitted on the current chunk of data. If the ensemble is full and a new classifier is added to the ensemble, the worst performing classifier is removed.

### Feature selection

There are two different types of drifts that we distinguish. First, there is a *concept drift* which describes changes in the probability distribution of the classes at two consecutive time points. Second, there is a *feature drift* which is a change in the most discriminative feature subset at two consecutive time points.

- Selection of relevant features: Maximize the correlation with the class label and minimize the correlation with other features
- Dimensionality reduction: not all features in high-dimensional datasets are critical for training a classifier

**Fast Correlation Based Filter (FCBF)** selects features that correlate strongly with the class label, but have little correlation with other features. The correlation measure used is called Symmetrical Uncertainty.

**Conditional Mutual Information (CMIM)** is similar to FCBF. It uses a cost function that focuses on both correlation with the class label and independence from other features and does not require tuning of a threshold hyperparameter.

## Experiments and Results

### *Datasets*

#### Synthetic

**LED** *24* binary features, *10* classes
digits displayed on a seven-segment LED display, each attribute has a 10 % chance of being inverted, 17 of the features are irrelevant for the classification

**SEA** *3* numeric features, *2* classes
two relevant features are summed and compared to a threshold

#### Real-World

**weather** *8* numeric features, *2* classes
daily aggregated weather information like temperature, humidity and pressure binary labeled by whether or not it rained that day

**KDD** *41* features, *23* classes
network data with a distinction between *good* and *bad* connection types or attacks

### *Experiments*

#### Setup

The proposed HEFT algorithm was implemented in Python using the *scikit-multiflow* framework. We tested the influence that different parts of the algorithm have on runtime and mean classification accuracy. The altered hyperparameters were the chosen base learners, different feature selection methods and the method of choosing which classifier is added to the ensemble after a feature drift.

#### Base Learners

The base learners that may be part of the ensemble initially are: *Online Naive Bayes*, *Extremely Fast Decision Tree (HATT)* and *Multilayer Perceptron*.

#### Choosing new base learners

In the original HEFT algorithm, only the best performing type of classifier currently in the ensemble is considered when training a new one. This causes the problem of the ensemble converging to a homogeneous steady state. Therefore we want to evaluate the influence of randomly choosing a base learner that is to be added to the ensemble, which should prevent this limitation.

All experiments were conducted by training on 25,000 samples.

*Experimental results on synthetic and real-world datasets.*

| | HEFT | | AWE(NB) | | AEE(NB) | |
|---|---|---|---|---|---|---|
| | Acc | Time[s] | Acc | Time[s] | Acc | Time[s] |
| LED | **0.733**[1] | 99.2[1] | 0.729 | 463.5 | **0.733** | 579.3 |
| SEA | 0.825[2] | 44.2[2] | 0.830 | 31.3 | **0.881** | 42.3 |
| KDD | 0.843[3] | 636.2[3] | 0.742 | 152.4 | **0.995** | 596.9 |
| weather | 0.737[4] | 173.3[4] | **0.738** | 20.1 | 0.694 | 27.2 |

[1] Base learners: NB, MLP; non-random choice; CMIM feature selection
[2] Base learners: NB, MLP; non-random choice; CMIM feature selection
[3] Base learners: NB, HATT; non-random choice; FCBF feature selection
[4] Base learners: NB, HATT, MLP; non-random choice; CMIM feature selection

*Base learner choice in HEFT.*

| | standard | | random | |
|---|---|---|---|---|
| | Acc | Time[s] | Acc | Time[s] |
| LED | 0.624 | 1578.7 | **0.680** | 1731.0 |
| SEA | **0.714** | 152.1 | 0.662 | 205.0 |
| KDD | **0.736** | 343.2 | **0.736** | 404.3 |
| weather | 0.696 | 71.8 | **0.705** | 87.4 |

*Feature Selection in HEFT.*

| | FCBF | | CMIM | |
|---|---|---|---|---|
| | Acc | Time[s] | Acc | Time[s] |
| LED | 0.583 | 217.0 | **0.721** | 3092.7 |
| SEA | 0.660 | 152.9 | **0.716** | 204.1 |
| KDD | **0.843** | 676.3 | 0.630 | 71.2 |
| weather | 0.687 | 23.2 | **0.714** | 136.1 |

## Conclusion

According to our results from the first table, our implementation of the HEFT algorithm does not match that of the paper's. Accuracy-wise it has a performance that is close to the state-of-the-art stream learning algorithms. Time-wise it does not perform nearly as well as one would have expected. We were not able to reproduce the gains in time and performance that [1] could achieve.

A random choice of which base learner to add when the ensemble is full has a negative impact on accuracy and training time when compared to the standard method on artificial datasets, as seen in the second table. On real world data, the accuracy is at least as high as when employing the standard method, the runtime however is about 10 – 20 % worse.

The random choice method also significantly reduces the chance of converging to a homogeneous ensemble. The third table shows that HEFT in conjunction with *CMIM* performs better on numeric feature datasets, such as LED, SEA and weather, than *FCBF*, while *FCBF* provides a much higher accuracy on nominal feature datasets such as KDD. Moreover, *FCBF* is generally faster to compute.

### *References*

[1] [Nguyen, 2012] Hai-Long Nguyen, Yew-Kwong Woon, Wee-Keong Ng, Li Wan (2012)
Heterogeneous ensemble for feature drifts in data streams
*Pacific-Asia conference on knowledge discovery and data mining*, 1 – 12.

[2] [Jacob Montiel, 2018] Jacob Montiel, Jesse Read, Albert Bifet, Talel Abdessalem (2018)
Scikit-Multiflow: A Multi-output Streaming Framework
*Journal of Machine Learning Research*, Vol 19, no 72, 1 – 5

[3] [François Fleuret, 2004] François Fleuret (2004)
Fast binary feature selection with conditional mutual information
*Journal of Machine Learning Research*, Vol 5, 1531 – 1555

**Data Mining Lab - Winter Semester 2019/2020 - Code: https://kohlhasej.github.io/heftstream/**