SketchBoost: Fast Gradient Boosted Decision Tree for Multioutput Problems

Leonid Iosipoi,
Sber AI Lab and HSE University, Moscow, Russia

Anton Vakhrushev,
Sber AI Lab, Moscow, Russia

NeurIPS 2022

Background on GBDT

Gradient Boosted Decision Tree (GBDT) is one of the most powerful methods for solving prediction problems in both classification and regression domains.

It is a dominant tool today in applications where tabular data is abundant, for example, in e-commerce, financial, and retail industries.

It has contributed countless top solutions in Kaggle competitions.

Motivation

Our main focus is the scalability of GBDT to multioutput problems:

- multiclass classification
- multilabel classification
- multioutput regression

These problems arise in various areas such as Finance, Multivariate Time Series Forecasting, Recommender Systems, and others.

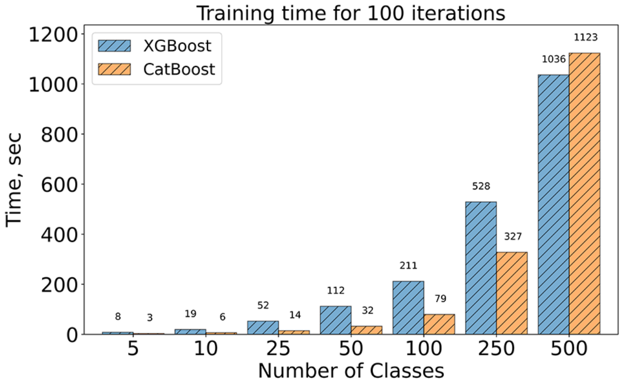
Motivation

There are several extremely efficient, open-source, and production-ready implementations of GBDT such as

- XGBoost
- LightGBM
- CatBoost

And even for them, learning a GBDT model for moderately large datasets with high-dimensional output can require much time.

Motivation



XGBoost and CatBoost training time for 100 iterations on a synthetic dataset (2000k instances, 100 features) for multiclass classification.

Fast Split Scoring

In this work, we propose a general methodology and novel methods to accelerate the training process of GBDT in the multioutput scenario.

Key idea: To compress the data associated with output dimensions during the split search (the most time-consuming step in GBDT) while keeping other boosting steps without change.

This is achieved by approximate computation of a scoring function used to find the best split of decision trees.

Numerical Results

To evaluate the performance of proposed methods, we made a simple and fast GBDT library called Py-Boost[1]

- It is written in Python
- It is easily customizable
- It has a similar to the scikit-learn interface
- It works on GPU and uses Python GPU libraries (e.g., CuPy and Numba)

[1] `https://github.com/sb-ai-lab/Py-Boost`

Numerical Results

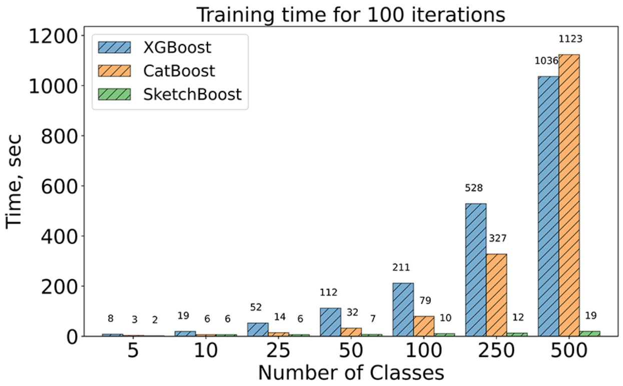
SketchBoost[2] is a part of Py-Boost library which implements the proposed "sketching" methods.

In our numerical study, we compare SketchBoost to

- existing state-of-art boosting toolkits (XGBoost and CatBoost)
- deep learning baseline (TabNet)

[2] `https://github.com/sb-ai-lab/SketchBoost-paper`

Numerical Results



SketchBoost, XGBoost, and CatBoost training time on a synthetic dataset $(2000$k instances, 100 features) for multiclass classification.

Numerical Results

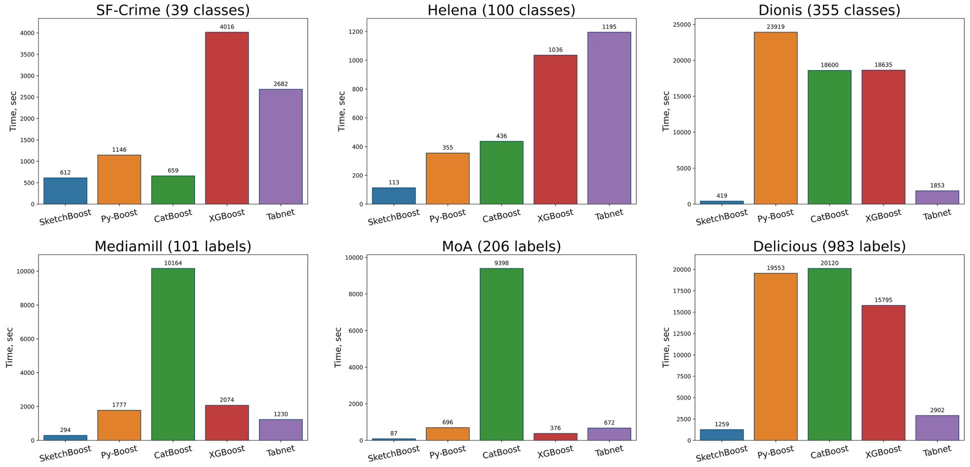
The experiments are conducted on datasets from Kaggle, OpenML, and Mulan website for multiclass and multilabel classification.

Dataset	**Task**	**Rows**	**Features**	**Classes/Labels**
SF-Crime	multiclass	878 049	10	39
Helena	multiclass	65 196	27	100
Dionis	multiclass	416 188	60	355
Mediamill	multilabel	43 910	120	101
MoA	multilabel	23 814	876	206
Delicious	multilabel	16 110	500	983

* a more extensive study is presented in the paper

Numerical Results

Training time per fold in seconds.



* CatBoost can be trained on GPU only for multiclass classification tasks.

Numerical Results

Cross-entropy loss on test set.

	Our Algorithms		Baselines		
Dataset	**SketchBoost**	**Py-Boost**	**CatBoost**	**XGBoost**	**TabNet**
Multiclass classification					
SF-Crime (39 classes)	2.2038	2.2067	**2.2036**	2.2208	2.4819
Helena (100 classes)	**2.5673**	2.5865	2.5698	2.5889	2.7197
Dionis (355 classes)	**0.2848**	0.3114	0.3085	0.3502	0.4753
Multilabel classification					
Mediamill (101 labels)	**0.0743**	0.0747	0.0754	0.0758	0.0859
MoA (206 labels)	**0.0160**	0.0160	0.0161	0.0166	0.0193
Delicious (983 labels)	0.0620	0.0619	**0.0614**	0.0620	0.0664

Conclusions

- We introduce “sketching” methods that are generic and can be easily integrated into any GBDT realization.
- Our empirical study shows that these methods achieve comparable and sometimes even better results than the existing state-of-the-art GBDT implementations but in remarkably less time.
- These methods are implemented in SketchBoost which itself is a part of our Python-based implementation of GBDT called Py-Boost.

Thank you for listening!

Py-Boost: `https://github.com/sb-ai-lab/Py-Boost`

Other projects of sb-ai-lab: `https://github.com/sb-ai-lab`