A Learning Augmented approach to Cardinality Estimation

Mărcuș Alexandru Marian

December 3, 2025

1 Schita cuprins

Abstract
Introduction
Related Work
Theoretical Foundation
Methodology
Implementation
Experiments
Conclusions

2 Theoretical Foundation

consistency, robustness, competitiveness
loss function: log loss/ cross entropy $L = (\log(E) - \log(N))^2$

3 Plan pentru partea aplicativa a lucrarii

Implementarea standard a algoritmelor HLL si HLL++.
Generare de date cu diverse distributii ce ar putea fi regasite in date reale (ex. uniform, clustered etc).
Feature extraction din HLL sketch (ex. max, min, media, devia standard, % din registre sunt goale, histograma a valorilor din registrii).
Antrenarea unui model mic (ex. regresie, un neural network cu putine layere etc).
Evaluarea modelului fata de standarde ca si acuratete, runtime si memorie.
Analiza rezultatelor si concluzii.

4 Methodology

4.1 Experimental Setup

To measure the performance of this method, we generated 200000 different HyperLogLog sketches, each sketch having a cardinality in the range of $[10, 10^8]$, with a log-uniform distribution. Assuming a uniform hash function, we can generate uniformly distributed 64 bit integers, simulating the hash values of random elements. For real-world datasets we are using

4.2 Model matematic

Algorithm 1 Learned HyperLogLog

Require: Let $h : \mathcal{D} \to \{0,1\}^{64}$ hash data from domain $\mathcal{D}$. Let $m = 2^p$ with $p \in [4..16]$.

Phase 0: Initialization.

Initialize m registers $M[0]$ to $M[m-1]$ to 0.

Phase 1: Aggregation.

for all $v \in S$ **do**

  $x := h(v)$

  $id := (x_{63}, \ldots, x_{64-p})_2$ ▷ First p bits of x

  $w := (x_{63-p}, \ldots, x_0)_2$

  $M[id] := \max(M[id], \rho(w))$

end for

Phase 2: Result computation.

return Model Prediction

Instead of using the classic formula $E := \alpha_m m^2 \left(\sum_{j=0}^{m-1} 2^{-M[j]}\right)^{-1}$ [1] for extracting the result from our sketch, we propose training a neural network to predict the cardinality. This should have several advantages, such as reduced bias and better results for real world distributions. As a bonus, we also eliminate the need for magic numbers and special cases of the original algorithm. This approach should not have much overhead, in terms of memory or speed.

For the features of the model we are using As the loss function, we chose the Mean Squared Logarithmic Error $L = \frac{1}{n}\sum_{i=0}^{n}\left(\log(1+\hat{N}) - \log(1+N)\right)$

4.3 Results Validation

For validating our results we are plotting the Relative Error $error := \frac{|\hat{N}-N|}{N}$. In the literature, the "raw" HLL estimate shows different performances for different ranges of cardinality, that are often empirically determined and corrected. We compare our results to the classic HLL [1], HyperLogLog++ [2] and other learned methods [15] [16]. The ideal result would be an error close to the theoretical limit $1.04/sqrt(m)$

References

[1] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. *Discrete Mathematics & Theoretical Computer Science*, DMTCS Proceedings vol. AH, 2007 Conference on Analysis of Algorithms (AofA 07), Jan 2007.

[2] Stefan Heule, Marc Nunkesser, and Alexander Hall. Hyperloglog in practice: algorithmic engineering of a state of the art cardinality estimation algorithm. In *Proceedings of the 16th International Conference on Extending Database Technology*, EDBT '13, page 683–692, New York, NY, USA, 2013. Association for Computing Machinery.

[3] Aiyou Chen, Jin Cao, Larry Shepp, and Tuan Nguyen. Distinct counting with a self-learning bitmap. *Journal of American Statistical Association*, 106:879–890, 2011.

[4] Renzhi Wu, Bolin Ding, Xu Chu, Zhewei Wei, Xiening Dai, Tao Guan, and Jingren Zhou. Learning to be a statistician: learned estimator for number of distinct values. *Proceedings of the VLDB Endowment*, 15(2):272–284, October 2021.

[5] Xiaoying Wang, Changbo Qu, Weiyuan Wu, Jiannan Wang, and Qingqing Zhou. Are we ready for learned cardinality estimation? *CoRR*, abs/2012.06743, 2020.

[6] Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. *CoRR*, abs/1712.01208, 2017.

[7] Zhenwei Dai and Anshumali Shrivastava. Adaptive learned bloom filter (ada-bf): Efficient utilization of the classifier. *CoRR*, abs/1910.09131, 2019.

[8] Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with predictions. *CoRR*, abs/2006.09123, 2020.

[9] Michael Mitzenmacher. A model for learned bloom filters and related structures. *CoRR*, abs/1802.00884, 2018.

[10] Kenneth G. Paterson and Mathilde Raynal. HyperLogLog: Exponentially bad in adversarial settings. Cryptology ePrint Archive, Paper 2021/1139, 2021.

[11] Qingzhi Ma, Ali Mohammadi Shanghooshabad, Mehrdad Almasi, Meghdad Kurmanji, and P. Triantafillou. Learned approximate query processing: Make it light, accurate and fast. In *Conference on Innovative Data Systems Research*, 2021.

[12] Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. *CoRR*, abs/1802.05399, 2018.

[13] Keerti Anand, Rong Ge, and Debmalya Panigrahi. Customizing ml predictions for online algorithms, 2022.

[14] Otmar Ertl. New cardinality estimation algorithms for hyperloglog sketches. *CoRR*, abs/1702.01284, 2017.

[15] Kangfei Zhao, {Jeffrey Xu} Yu, Hao Zhang, Qiyan Li, and Yu Rong. A learned sketch for subgraph counting. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 2142–2155, 2021. Publisher Copyright: © 2021 ACM.; 2021 International Conference on Management of Data, SIGMOD 2021 ; Conference date: 20-06-2021 Through 25-06-2021.

[16] Brian Tsan, Asoke Datta, Yesdaulet Izenov, and Florin Rusu. Approximate sketches. *Proc. ACM Manag. Data*, 2(1), March 2024.