Paper Report "MacroBase: Prioritizing Attention in Fast Data" Big Data Storage and Processing Infrastructures

Fayán Leonardo Pardo Ladino

March 2019

1 Introduction

In this report I am going to summarize and explain the paper "*MacroBase*: Prioritizing Attention in Fast Data". I am going to explain the main concept and description of *MacroBase*, the design behind it, the tests the researchers did on the solution, comment their results and finally I will link it with the IBD course and expose my conclusions gathered after reading the paper.

2 Paper Introduction

As stated in the paper we are nowadays gathering and dealing with huge amounts of data, companies like *Twitter* gather more than 12M events per second. These numbers have been increasing since then (2017) and it is still not going to stop increasing. This has made difficult to process data with human abilities, not only because of the amount of data, but also because in most of the cases this information must be analysed fast. Although there are industrial existing solutions to deal with these problems, such as *Apache Flink* which enables stream processing, these still leave to the application developer the implementation of scalable analysis that prioritize attention. This development should take into account that fast data analysis should determine few results to send them to end users, execute quickly to keep up with huge data volumes and adapt changes within the data stream itself. The paper also states that the high-ends industrial deployments rely a lot on static rules and thresholds which are computationally efficient but fragile and error-prone.
With all these problems in mind, researchers from Stanford University developed *MacroBase*. *MacroBase* is an analytics engine which provides operators to classify and explain fast data volume to users using extensible streaming dataflow pipelines. It permits the users to tune the queries by adding specific feature transformations to their pipelines, providing supervised classification rules to complement or replace unsupervised classifiers and creating custom streaming transformation, classification and explanation operators.
The paper also states in this section that *MacroBase* has been successful in several analysis implementations, for example, to find unusual and previously unknown behaviours in their products. In order to explain better in the paper how *MacroBase* works, they focused in three industrial use cases they seek to support with this solution: mobile applications, datacenter operation and industrial monitoring.

3 MacroBase Architecture

As stated before, *MacroBase* prioritizes attention and to do so it uses two classes of operators: Classification operators, which examine data points and labels them according to user-specified classes, and Explanation operators which will group and aggregate multiple data points. Developing these operators is the core concept behind *MacroBase*, relational analytics have well defined re-usable operators but not for classification and explanation. With this in mind, the paper explained *MacroBase* structure and architecture the following way.
MacroBase runs **Query Pipelines** of dataflow operators over input data streams. These pipelines follow two principles: First, and as I just mentioned, the operators work over data streams; second, it uses a compiler system to enforce interoperability, making each operator to implement one of several type signatures. By this way the compiler enforces that all pipelines will follow a common structure:
1. Ingestion. *MacroBase* ingests data streams from external data sources. In the paper they show as example JDBC to read from SQL queries. Following this example, *MacroBase* will cre-

ate data points for each row ingested and each data point will have a set of *metrics* used for key measurements (e.g. trip time) and *attributes* that correspond to associated metadata (e.g user ID). The metrics will be used to detect unusual events and the attributes to explain behaviours.
2. Feature Transformation. *MacroBase* executes data transformations over the stream. These transformations can be statistical, datatype specific or time series specific operations. This feature placed at the beginning of the pipeline lets users to prepare or encode their data for the following steps without modifying later stages. Also given that the base type is unchanged in this feature (*Point→Point*), it can be chained several times to apply different transformations.
3. Classification. *MacroBase* labels each point according to its metrics. The training and the evaluating classifiers of the metrics happen in this stage. For this, *MacroBase* supports several models. The paper describes later, for example, a default unsupervised model offered by *MacroBase* but users can also use their supervised and pre-trained model. This operator will receive a stream of data points (*Point*) and will create a stream of a labelled points (*Label, Point*).
4. Explanation. Instead of returning the result of labelled points, *MacroBase* will aggregate them and create explanations. By default, *MacroBase* will return explanations in the form of attribute-value combinations common among outlier points and uncommon among inlier points. This operator will receive a stream of labelled points (*Label, Point*) and will generate explanations (*Explanation*). The *Explanation* subclass provide additional information like statistics or representative sequences of points. *MacroBase* will continuously execute explanation operators to summarize the stream, but the emission of explanations will be done by user demand to avoid waste of resources.
5. Presentation. Given that the number of generated explanations may still be large, *MacroBase* pipelines rank explanations by statistics specific to the explanations and sorts them by their degree of outlier. By default, *MacroBase* will render a report and present it via REST API or GUI.
Operating Modes. *MacroBase* supports three operating modes: *MacroBase* graphical front-end to allow users to interactively explore their data and configure inputs, metrics and attributes; One-shot queries executions that can be run programmatically over the data; Streaming queries that can be run programmatically over a potentially infinite stream of data, if desired, with triggers to alert customers or users.

4 MacroBase Default Pipeline Classification

Although *MacroBase* lets users to configure their own operators, they also offer default classification operators. In the paper they explain how the estimation procedure and the streaming execution is designed.
1. Robust Estimation Procedure. *MacroBase* uses unsupervised density-based classification to identify points that abnormal relative to the population, but as the paper states some anomalous points can impact on density estimation with calculations such as *Z-Score*. To solve this problem *MacroBase* Default Pipeline (MDP) uses the following robust estimation calculations: the Median Absolute Deviation (MAD) for univariate data since the median is resistant to outliers and the Minimum Covariance Determinant (MCD) for multivariate data that finds the tightest points that represent the sample and summarizes them with their location and scatter. These unsupervised models allow MDP to score points without requiring labels or rules from users and in the paper they clearly show these procedures are way more robust than the *Z-Score*.
2. MDP Streaming Execution. The procedures explained before, although robust, are not prepared for an streaming context in which they must be retrained over a continuous stream of data. For this problem, *MacroBase* uses the Adaptable Damped Reservoir (ADP) which maintains

a sample of input data that is exponentially weighted towards more recent points and also operates over window sizes enabling flexibility. The probability of inserting a point in this window is an exponentially weighting function of the number of points observed so far, while the decay of points may depend on the users application and/or workload. For this reason, *MacroBase* supports two decay policies: time-based decay which decays at a specified rate measured according to real time and batch-based decay which decays at a specified rate measured by arbitrarily sized batches. With this implementation *MacroBase* solves the problem of maintaining training inputs and maintaining percentile thresholds.

5 MacroBase Default Pipeline Explanation

As explained above, *MacroBase* uses explanations to contextualize and differentiate outliers and inliers according to their attributes. In this paper they explain how to achieve this, what is the explanation strategy and how to implement it in streaming.
1. Risk Ratio or relative risk ratio, is an epidemiology metric used in the MDP to identify combinations of attribute values that are relatively common in outliers. This metric quantifies how much more likely a data point is to be an outlier if it is of a specific attribute combination, as opposed to the general population. MDP finds combinations with high support or occurrence and, for this purpose, it accepts a minimum risk ratio and level of outlier support as input parameter.
2. Basic Explanation Strategy. As the paper states, the first solution to apply risk ratio for several attribute sets is to search twice but this is a waste of time and effort. To solve this, MDP exploits the cardinality imbalance that reduces the attributes to explore and also takes advantage of individual item ratios computing them first as these are cheap or inexpensive and then the combinations. The final strategy is as follows: MDP calculates the attribute values with minimum risk ratio, then computes the supported outlier attribute combinations and finally computes the risk ratio for each combination based on their support in the inliers.
3. Streaming Explanation. They state in the paper that individual attributes have enough characteristics to support counts. To count individual attributes frequent items the MDP uses a heavy-hitters sketch called Amortized Maintenance Counter (AMC). However, MDP also needs to track combinations of attributes and, for that, they developed a combination of data structures: AMC for the frequent attributes and an adaptation of the CPS-Tree data structure to store frequent attributes. To sum up the whole explanation streaming process, when a data point arrives at this summarization operator, *MacroBase* inserts each point's attributes into an AMC sketch, then inserts a subset of point's attributes into a tree that maintains a frequency descending order and when the window elapses, *MacroBase* decays the count of the items and the counts in each node of the prefix tree, removes the attributes that are no longer above the support threshold and rearranges the prefix tree order.

6 Evaluation

In this part of the paper, the researches set a prototype server for *MacroBase* and tested the solution and several characteristics of its design, comparing the results with other solutions if possible.
1. Result Quality. In order to test the result quality of *MacroBase* they performed several tests with different inputs conditions. With these tests they confirmed high quality results: It is able to identify correlated causes of outlying data for noise of 20% or more, a magnitude which is likely rare in practice; it accurately identifies systematic abnormalities in real world data; the ADR approach makes MDP resilient to variable arrival dates; and they even tested *MacroBase* in production with

CMT, obtaining undiscovered results.
2. End-to-End Performance. After testing performance of different *MacroBase* phases on different data sets, arriving to the conclusion that the overhead of each component is data -and query- independent.
3. Microbenchmarks and comparison. They tested the cardinality aware explanation efficiency compared to FP-Growth and they found out that while having same results, *Micro Base* strategy was faster. They also evaluated AMC and compared it to other heavy-hitter sketchers. Specifically compared with SpaceSaving, AMC outperfomanced having more space consumption cost which may be a fact to take into account in case you must save memory space.
4. Case studies and extensibility. Finally, in the tests explained in the paper they expose how they tested *MacroBase* over supervised, time-based and video surveillance data, showing different application and successful results.

7 Conclusions

To sum up and as we have seen in the paper, *MacroBase* is an analytics engine that prioritizes attention in fast data streams. To achieve this, they focused the solution development in providing a flexible architecture that combines streaming classification and data explanation. This focus provided *MacroBase* an increase in performance and result quality which have been useful in some environments that have been shown in the paper.
In my opinion, I find the solution exposed in the paper pretty interesting. We have seen in IBD class solutions to analyse data and create monitoring pipelines, such as *Apache Spark*, *Apache Flink*, *Apache Beam* etc. but all of these solutions had the same problem: after passing the data through the pipelines, there is still a human analysis of the resulting data to know where to focus attention. With *MacroBase* this bottleneck can be avoided and we can get accurate conclusions over data way faster. The design approach done with *MacroBase* makes it unique in the market, if the users want to get anything close to this solution, they must implement them themselves with other monitoring pipelines technologies like the ones mentioned above, with the handicap that these were not designed with the same focus, thus they will not be as accurate and fast as *MacroBase*.
I personally find the paper quality really good. They explained in depth the design of the solution with the detailed explanation about the choices they made when implementing it. They also explained the tests to prove the improvement in performance and accuracy and they explained the tools and steps done to verify them. The only negative point I can say about the paper/solution is that they do not talk how to deal with scalability in the main sections. Even with the high performance results achieved, *MacroBase* should also support parallelization to deal with the still increasing data loads. This concern is mentioned in Appendix D, and they even tested parallelization with *MacroBase*, obtaining less accurate results. However, they mentioned in this appendix that is at the moment subject of ongoing work, meaning that this concern is identified and there may be a solution in the near future.