December 14, 2018

SYSTEMS AND METHODS FOR PREDICTING ONLINE VIDEO METRICS

Aaron Shon

Charles Johan Larsson Tibell

Axel Erik Olov Henriksson

SYSTEMS AND METHODS FOR PREDICTING ONLINE VIDEO METRICS

In big data architecture, input data from a database or from real-time systems is fed to an analytics engine that runs a predictive analytics model. Client events, such as actions of users indicating consumption or interaction with certain online content at client devices (e.g., clicking to watch a video online), generate discrete time-stamped records of events, known as logs, that can be stored in a database. Processing and analyzing these logs can lead to gaining actionable insights about usage pattern that may be useful to determine usage trend including predicting usage pattern in near future.

Typically, accuracy of prediction by an analytics engine increases with the amount of data that is fed to the engine, i.e. the size of the database. Often, event data is accumulated in a log over a significant amount of time, e.g., over a few hours, or even over an entire day. Therefore, analytics engines whose only source of input data is an aggregated event log, are associated with high latency (in the range of hours or days), even if the analytics engines themselves are capable of producing results within a few seconds once the data is fed. Additional factors that negatively affect latency in log-based analytics include, but are not limited to, delays in log processing, network delay, delay due to spam marking etc.

Existing low-latency data ingestion and processing systems may collect real-time event data (as well as event logs) as sources of data to feed into the analytics engine. However, the prediction accuracy may fluctuate depending on which data source (or which combination of data sources) is being used by the analytics engine. For example, unreliable and lossy real-time data negatively impacts prediction accuracy. Additionally, often existing predictive analytics models cannot efficiently handle probabilistic interpretation, making it difficult for a user to know when to trust the metric values that the models predict.

Finally, irrespective of the latency of the data processing and analytics, existing analytics systems are vulnerable to abuse by malicious actors, such as hackers and other unauthorized users, if the abusers can access the direct output of the analytics engine.

The trend prediction system described here addresses all of the above shortcomings by implementing a low-latency reporting mechanism that runs as a layer on top of an analytics engine. The output of the reporting layer is based on, but not identical to, the direct output of the analytics engine, and hence, less amenable to reverse-engineering by the abusers. The mechanism described here relies on a low-latency data ingestion and processing system to receive real-time event data. Simultaneously, a high-latency and high-accuracy data ingestion and processing system receives log data and generates canonical data based on the log data. A training module receives these two sets of data, and uses machine learning techniques to train an artificial intelligence network that predicts metrics based on incoming data analysis. The artificial intelligence network may determine inferences that are communicated to a predictive analytics system indicating the confidence level in the prediction, and the predictive analytics systems analyzes the received data and determines whether to present the resulting metrics in a user interface (UI) or not.

The metrics generated by predictive analytics may be useful for online content creators/distributors to gauge current interest level in the content, as well as forecast future interest in that type of content. The metrics may also be helpful for capacity planning for hosts of online content. The online content may be online video data suitable for video analytics predicting video metrics (such as view counts), though the systems described here can easily be modified to generate metrics in domains outside of video analytics. In short, the trend prediction system proposed here combines diverse technologies (including machine learning) to give

predictions with sub-second latency and with confidence bounds to inform when the prediction should be trusted. The analytics UI can decide whether or not to display the prediction, based on a predefined threshold confidence level.

One unique aspect of the proposed trend prediction system is continuous learning. The predictions are generated by a predictive model that learns via continuous serving systems (such as TensorFlow), i.e., the model can be continuously trained with new incoming real-time data. Model training uses "mini-batches" of data typically containing a few thousand data points at a time. The system can thereby accommodate changes faster than conventional machine learning systems trained on larger, less frequent data batches.

Another unique aspect of the proposed trend prediction system is generation of online inference with confidence intervals. A key ability of the system is the probabilistic interpretation of the model output. The system calculates the confidence intervals and a maximum likelihood value of the metrics generated by the analytics model. The system further determines whether to display or not to display the predicted value based on whether the confidence interval is greater than or less than a predetermined threshold value. Following are two example methods that can be used to build confidence intervals:

1. *Confidence interval width based on inference likelihood*: Gradient-descent based training methods can output, an inference log likelihood as an indication of how well the model fits the training data. The inference log likelihood can be used as a proxy for confidence interval width by applying a scaling factor to compute confidence interval width. That is, given a set of input points X_0, ..., X_N from the autoregressive time series data received from the data ingestion and processing system, the model generates a

certain negative log likelihood '*-L'*. The confidence interval is set to be 'α/L' where α is a scaling constant independent of the model parameters.

2. *Explicit noise model*: A generative graphical model (such as a deep belief neural network) can be adapted to learn the confidence intervals of each individual metrics.

An additional important aspect of the proposed trend prediction system is quantization to address potential abuse and privacy concerns. A probabilistic technique (e.g., dithering) may be used to quantize the output data of the analytics model. This helps to keep privacy in the data because there is an additional non-deterministic layer which makes attacks by the abusers more difficult. Additionally, models are protected from being reverse-engineered by not directly exposing the model outputs, but exposing the model inferences instead. Since inference labels are probabilistic, it is much more difficult to accurately know exact outputs of the underlying analytics system even if the inference labels are known.

Furthermore, the proposed trend prediction system is scalable to generate a very high number of inferences, e.g. billions or even trillions of inferences per day. The system can work with existing serving infrastructure (such as TensorFlow) for machine learning models that support tens of millions to hundreds of millions of queries per second (QPS), enabling the inferences to scale massively as traffic demands increase.

Figure 1shows a data flow diagram for the proposed trend prediction system 110. As described above, the trend prediction system 110 has a training module 112 which trains an artificial intelligence network 114 (e.g., a deep belief neural network) capable of doing predictive analysis. The training module receives data from both a high-latency data ingestion and processing system 106 as well as a low-latency data ingestion and processing system 108. The systems 106 and 108 receive data from an event log 104. Additionally, when events 102 occur at

client devices, the systems 106 and 108 receive real-time online data. The low-latency data ingestion and processing system 108 also communicates inference features 118 to the training module 112 at serving time. Training module determines model parameters 122 based on the autoregressive real-time time series data set and communicates the model parameters to the artificial intelligence network 114 in the analytics engine. The analytics engine runs the model and predicts metrics as model output. Additionally, the analytics engine determines inferences 120 based on the inference features 118 and incoming data from the low-latency system 108. The inferences 120 are communicated back to the training module to increase the efficacy of model training. Confidence intervals may be associated with the predictions, as described above. Further, the analytics module decides whether to display predicted metrics with associated confidence interval 124 in the UI 116 of the analytics module.

The trend prediction system 110 can be used to detect anomalous traffic, which may be caused by abusers. Videos can be prioritized for flagging if their predicted traffic greatly varies from their actual traffic. The system 110 can also be used to build optimization models to help content creators predictively tune their channels and video content to maximally engage audiences. Another use of the system 110 is improving freshness of smart headlines based on content created by various content creators. For livestream traffic, search and discovery could use predictions from system 110 as an additional signal to promote live traffic that seems to be going viral.

The algorithms used by the proposed trend prediction system 110 can be used not only for video metrics, but across broader data sets of real-time data used by any analytics platform, such as datacenter analytics to predict latency or resource consumption in real time based on intermittent monitoring of datacenter resource use and availability. In general, not observing any

latency spike, i.e. not observing any irregularities in the display of data-based metrics indicates that a prediction system is helping to "smooth out' any underlying latency in the real-time analytics system.

ABSTRACT

A low-latency reporting mechanism that runs as a layer on top of an analytics engine is described. The analytics engine runs a predictive model to generate one or more metrics related to online content usage in near future. The predictive model is based on artificial intelligence and machine learning. The predictive model can be trained with low-latency real-time event data as well as canonical data obtained from historical event logs.

Keywords: Predictive Analytics, Low-Latency Data, Confidence Interval, Autoregressive Model, Abuse Prevention, Online Video Data

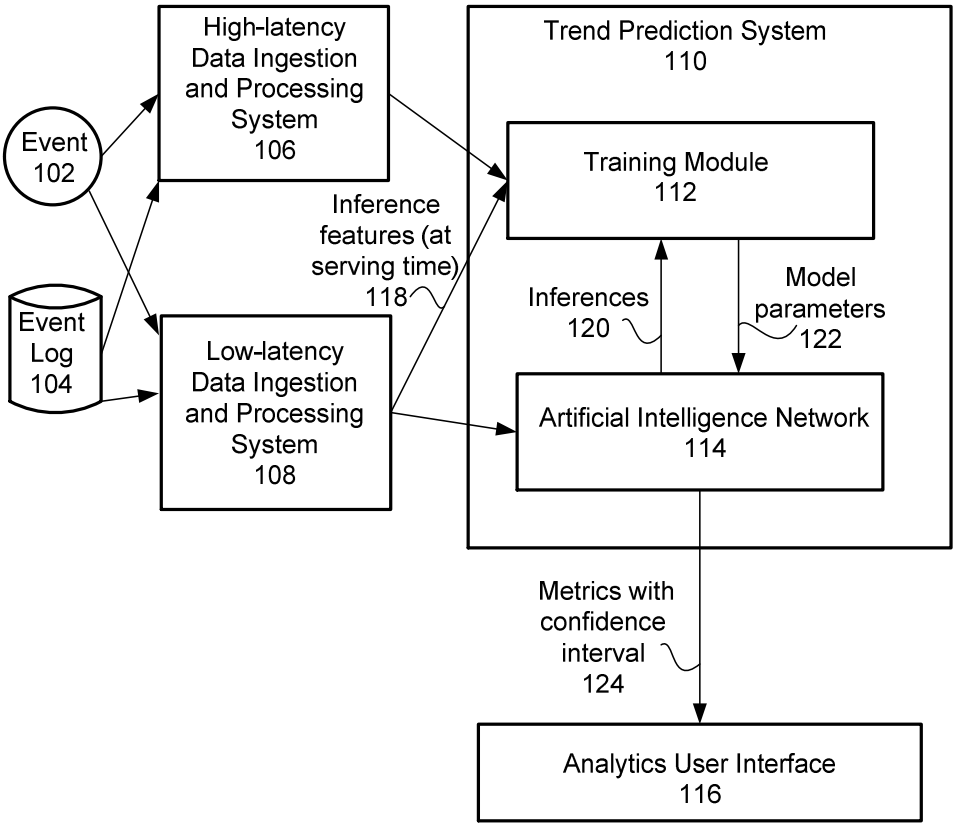
High-latency Data Ingestion and Processing System 106
Trend Prediction System 110
Event 102
Training Module 112
Inference features (at serving time) 118
Event Log 104
Inferences 120
Model parameters 122
Low-latency Data Ingestion and Processing System 108
Artificial Intelligence Network 114
Metrics with confidence interval 124
Analytics User Interface 116

FIG. 1