Edge-centric Efficient Regression Analytics

Natascha Harth
School of Computing Science, University of Glasgow
n.harth.1@research.gla.ac.uk

Christos Anagnostopoulos
School of Computing Science, University of Glasgow
christos.anagnostopoulos@glasgow.ac.uk

***Abstract*—We introduce an edge-centric parametric predictive analytics methodology, which contributes to real-time regression model caching and selective forwarding in the network edge where communication overhead is significantly reduced as only model's parameters and sufficient statistics are disseminated instead of raw data obtaining high analytics quality. Moreover, sophisticated model selection algorithms are introduced to combine diverse local models for predictive modeling without transferring and processing data at edge gateways. We provide mathematical modeling, performance and comparative assessment over real data showing its benefits in edge computing environments.**

***Index Terms*—On-line regression analytics, quality of analytics, communication efficiency, model selection, vector quantization.**

I. Introduction

Regression (predictive) Analytics (RA) provides statistical models (e.g., multivariate linear & quartile regression) and patterns discovered in data, uses such models to predict new/unseen data and investigates how unseen data fit such models [1]. Real-time RA [3], [4] are materialized *after* contextual data are transferred from sensing devices to the Cloud aiming to build global, on-line, models over *all* data. Then, analysts/applications issue *regression queries* over such models for real-time data exploration, on-line prediction, and adaptive knowledge extraction [5], [2]. However, major challenges arise adopting this *baseline* RA approach. Massive raw data transfer is needed for building and updating such models. Since this is prohibitive for IoT environments due to constraints like limited network bandwidth, computational power, latency and energy, *edge computing* comes into play [10], [8], [6]. Such paradigm can be adopted to cope with this challenge by *pushing* as much intelligent computing logic for analytics as possible close to computing & sensing Edge Devices (EDs) [1], [7]. It is desirable then for EDs to transmit *only* data summaries, e.g., sufficient statistics & regression coefficients to the Cloud for RA.

Motivations & Goals: We envisage an edge-centric RA paradigm, where EDs are employed as first-class RA platforms [15], [9]. Our motivation is based on RA materialized at the edge including e.g., physical sensors (sensing contextual information), mobile EDs for participatory sensing, and Edge Gateways (EGs) interacting with EDs and sensors/actuators, as shown in Figure 1. Moving real-time data from EDs to remote data centers incurs network latency, which is undesirable for interactive, real-time data exploration and inferential analytics applications; e.g., urban surveillance applications generate humongous volumes of data (speed cameras; environmental time-series; earthquake monitoring) that are bandwidth prohibitive to completely move to the Cloud in real-time [9]. The network connectivity is intermittent causing loss of functionality if Cloud connectivity gets lost.

Cloud should not be the panacea RA paradigm shift. We advocate edge-centric RA by pushing the analytics frontiers from centralized nodes to the network periphery. The *pushed* RA intelligence is distributed among EDs and EGs. This triggers the idea that EDs locally build on-line regression models, which are *maintained* and *selectively* forwarded to EGs for efficient *model selection and sophisticated aggregation*, instead of sending raw data from EDs to EGs and/or to Cloud. Based on this models-only communication between EDs and EGs, we desire to obtain the same RA quality/accuracy compared with the baseline RA centralized approach by being communication efficient. We stress that our edge-centric approach retains the core advantages of using Cloud as a support infrastructure but puts back the RA processing to the edge given that computing capacity of EDs still increases [9]. Our edge-centric approach enjoys local model building and efficient model updating, reacting timely to incoming information, thus preventing concentration of raw data to central locations and respecting privacy of sensitive information. EGs are then equipped with novel model selection strategies to determine the most appropriate local models to engage per issued regression query.

Challenges & Desiderata: Multidimensional contextual data have special features such as *bursty nature* and *statistical transiency*, i.e., values expire in short time while statistical dependencies among attributes change over time [14], [4], [3]. Hence, the challenges for edge-centric RA are: (i) on-line, local model learning on EDs requiring real-time model updating and selective model forwarding to EGs in light of minimizing communication overhead, (ii) *best* models selection on EGs per regression query, and (iii) model caching techniques that achieve as high analytics quality/accuracy as the centralized approach. The desiderata of our approach are: (1) By introducing selective model forwarding from EDs to EGs and model caching at EGs, the communication overhead is significantly reduced as only model's parameters and sufficient statistics are disseminated instead of raw data. This meets the desired latency and energy efficiency, and reduces the closed-loop latency to analyze contextual data in real-time. (2) Model selection at EGs allows for combining diverse local models per regression query w.r.t. sufficient statistics coming from EDs without transferring and processing data at EGs.

II. Rationale & Problem Fundamentals

We focus on parametric regression analytics, e.g., [2], [16], [14] in a $(d+1)$-dimensional data space $(\mathbf{x}, y) \in \mathbb{R}^{d+1}$, where we seek to learn the dependency between input $\mathbf{x}$ and output y estimated by the *unknown* global data function $y = f(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^d$. Input $\mathbf{x} = [x_1, x_2]$ can refer e.g., to attributes temperature x_1 and CO_2 emission x_2, while y is humidity. A regression query is represented as the point $\mathbf{q} \in \mathbb{R}^d$ such we locally explore the behavior of $f(\mathbf{x})$ around $\mathbf{q}$ and are provided the prediction $\hat{y} = f(\mathbf{q})$ with prediction error $e(\mathbf{q}) = y - f(\mathbf{q})$; e.g., predict humidity y given $\mathbf{q} = [q_1, q_2]$; temperature q_1 and CO_2 q_2.

Edge-centric RA learns on-line the unknown regression model $y = f_i(\mathbf{x})$ over input-output pairs $\{(\mathbf{x}, y)_i\} \in \mathbb{R}^{d+1}$ measured *locally* at ED i. However, due to the diverse nature/contextual surroundings of each ED, e.g., environmental urban monitoring sensors in a smart city experience different and/or overlapping data ranges of temperature, CO_2 emission, UV radiation, and humidity in different city regions [14], a global model f_G fitting *all* data and interpreting *all* statistical dependencies among attributes cannot capture the very specific characteristics of data subspaces in each ED i. This raises the necessity of estimating local models f_i per ED i representing their specific local data $\{(\mathbf{x}, y)_i\}$, as discussed later.

We should efficiently and effectively combine such diverse local models f_i built over different data into an EG, thus, EG being able to interpret the diverse statistical dependencies and provide accurate predictions to queries in real-time. The rationale behind the intelligence on EDs is that they selectively forward their local models f_i to the EG, where EG caches the models, notated as f_i^o, to provide analytics. Evidently, cache model replacement is part of the EDs' intelligence trading-off analytics quality at the EG for communication overhead.

EG supports RA given an *ensemble* of cached local models introducing a sophisticated model selection over n *cached* local models $\mathcal{F} = \{f_1^o, \ldots, f_n^o\}$ delivered by n EDs. The final fused model should perform as accurately as if one were told beforehand which local model(s) from $\mathcal{F}$ was the best for a specific query and which was the best global model f_G over all the collected data from all EDs. Obviously, given a query, the best possible subset of local models to be engaged for prediction cannot be known in advance on the EG. Moreover, due to the above-mentioned constraints, we cannot build the global f_G on EG or Cloud over all the data; EDs do not transfer raw data for efficiency.

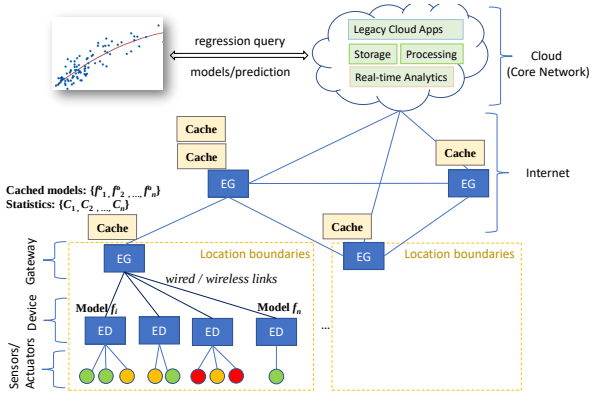

Fig. 1. Physical world is divided in geographical units where IoT devices are deployed. Edge-centric RA involve Edge Gateways, Edge Devices, and Edge Sensors and Actuators delivering cached models & sufficient statistics.

As an alternative, model selection can be simply averaging all local models: average model $f_{AVG}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x})$. However, as shown in Figures 2, 3 and Theorem 1, f_{AVG} induces unnecessarily large variability in prediction resulting in degraded accuracy. Specifically, consider the prediction error difference $\Delta e_i^G = e_{AVG} - e_i$ and $\Delta e_i^{AVG} = e_G - e_i$ of a local model f_i out of the global and average models, respectively, given a regression query. Figure 2 (left) shows the error differences (asc. order) for $n = 25$ local models derived from n EDs (Section V), where we *knew* beforehand that regression queries $\mathbf{q}$ followed the input data space distribution of each local model f_i, i.e., $\mathbf{q} \sim \mathcal{X}_i \equiv \{\mathbf{x}\}_i$. In this ideal case, we observe that all differences are positive, i.e., each local f_i provides better prediction than the average f_{AVG} (12% of local models perform similar/slightly better than f_{AVG} while 88% are significantly more accurate than f_{AVG}) and the global f_G (28% of local models are as accurate as f_G, while 72% are more accurate than f_G). Hence, given a query $\mathbf{q} \sim \mathcal{X}_i$ and *knowing* which model f_i to engage for prediction will provide accurate prediction.

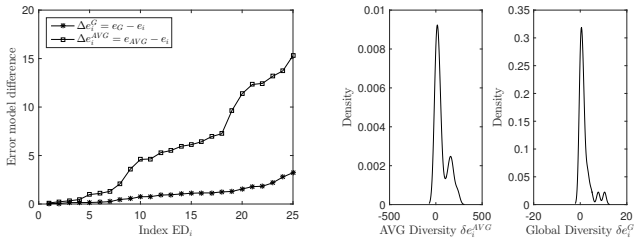

Fig. 2. (Left) Sorted error difference (asc) of each local model f_i with global f_G and average f_{AVG} models; (right) Probability density of local-average models diversity (δe^{AVG}) and local-global models diversity (δe^{G}).

Nonetheless, in reality, such information is not provided, thus, one has to *predict* which are the best model(s) to engage for prediction given an arbitrary regression query at EG. As shown, simple model averaging does not provide accurate predictions for more than 90% of the cases. The reason is the high *diversity* of the local models reflecting the very specific characteristics of the EDs' surroundings. The local-average models diversity $\delta e^{AVG} = \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n}(f_i(\mathbf{x}) - f_{AVG}(\mathbf{x}))^2]$ is defined as the variance of predictions derived from the local and average models. Figure 2 (right) shows the density distribution of δe^{AVG} for $n = 25$ models indicating highly diverse predictions among local models and their differentiation from f_{AVG}, thus, averaging eliminates such rich knowledge resulting to inaccurate predictions (see also Figure 3). Even if one is not communication aware by transferring all data from

EDs to Cloud and building/maintaining local models centrally, then, model averaging is not a reasonable solution. In our case, we avoid data transfer for local models building/maintenance, and require to obtain the highest possible quality of analytics being communication efficient and appreciate the diversities of local modeling. Similarly, local-global models diversity $\delta e^G = \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n}(f_i(\mathbf{x}) - f_G(\mathbf{x}))^2]$ (Figure 2(right)) indicates the ability of global f_G to provide more accurate predictions compared to f_{AVG} but no better than individual local models f_i when we know $\mathbf{q} \sim \mathcal{X}_i$ (Figure 2(left)). However, this comes at the expense of significant communication overhead for *entire* data transfer/model maintenance and lack of knowledge derived from local modeling.

The rationale behind the EG intelligence is to selectively engage some of the cached local models given a query by appropriate weighting than averaging while the case of global f_G modeling over all data in the EG is not feasible; no data are transfered from EDs to EG. Given a query $\mathbf{q}$, our challenge is to *predict* the most appropriate local models subset $\mathcal{F}' \subseteq \mathcal{F}$ in EG to engage in prediction by being as accurate compared to f_G as possible given (i) communication constraints, (ii) cached model replacement, and (iii) without knowing the distribution of the queries over input data $\{\mathcal{X}_i\}_{i=1}^{n}$.

Theorem 1. *Let e_G, e_{AVG}, and e_i be the prediction error of global f_G, average f_{AVG}, and local f_i models, respectively. It is not always true that $e_G < e_i$ and $e_{AVG} < e_i$.*

Proof. To prove Theorem 1, suppose its converse were true. Then it suffices to show counterexamples. Consider the multivariate linear regression $y \approx f(\mathbf{x}) = \mathbf{b}^\top \mathbf{x}$; $\mathbf{b} \in \mathbb{R}^d$ are coefficients. Given the data collection $\mathcal{X} = \mathcal{X}_i \cup \mathcal{X}_j$, where $\mathcal{X}_i$ and $\mathcal{X}_j$ are measured by ED i and ED j, respectively, we build global f_G over $\mathcal{X}$, f_i and f_j over $\mathcal{X}_i$ and $\mathcal{X}_j$ and the average $f_{AVG} = \frac{1}{2}(f_i + f_j)$. Given a query $\mathbf{q}$, then we would have been obtained better prediction using $f_i(\mathbf{q})$ than $f_G(\mathbf{q})$ and $f_{AVG}(\mathbf{q})$, as shown in Figure 3 (for query point $q_1 = 20$, f_2 provides more accurate prediction $f_2(q_1)$ than that of f_1, f_G, and f_{AVG} models since $q_1 \sim \mathcal{X}_2$), if we were told that $\mathbf{q}$ is drawn from the data distribution of $\mathcal{X}_i$, indicating that we should have engaged only f_i, thus yielding $e_i < e_G$ and $e_i < e_{AVG}$ avoiding averaging both local models. □

A. Problem Formulation

Our challenge is to predict the most appropriate $\mathcal{F}'$ per query that achieves almost the same or, hopefully, better accuracy than f_G and f_{AVG} without having to send all data to EGs.

Problem 1. *Given a local model f_i at ED i, whose image f_i^o is cached to the EG, define a communication efficient selective model update & delivery mechanism on ED i to replace the cached model at EG maximizing the analytics quality.*

Problem 2. *Given an ensemble $\mathcal{F} = \{f_1^o, \ldots, f_n^o\}$ of cached local models on EG, seek a model selection scheme to approximate the best $\mathcal{F}' \subseteq \mathcal{F}$ being as accurate as the global f_G had been built over all collected data disseminating only local models w.r.t. the update mechanism from Problem 1.*

Problem 3. *Given a local model f_i at ED i, define the sufficient statistics ED i should deliver to EG to guide the model selection in Problem 2.*

Regression includes parametric and non-parametric approaches [16], [20], [17]. Non-parametric approaches use stored data $\mathcal{X}$ for predictions, which in our context, are not computationally efficient in terms of data storage, calculations, and on-line updates/adaptations to incoming data [17]. Parametric regression seeks the optimal model parameters $\mathbf{b}$ from $\mathcal{X}$ that minimize the expected prediction error. Parametric models have the advantages of better interpretability of the data function, high prediction efficiency using only the parameters and not the data, and parameters adaptability [20]. This work focuses on parametric regression analytics.

Remark 1. *Our approach is generic in terms of the parametric regression models. Our algorithms extract knowledge only from the input space and prediction error being independent on the nature of the regression models/parameters on the EDs and their statistical expressiveness, which is application-specific/data-analysts decision on which parametric regression models to adopt for regression analytics.*

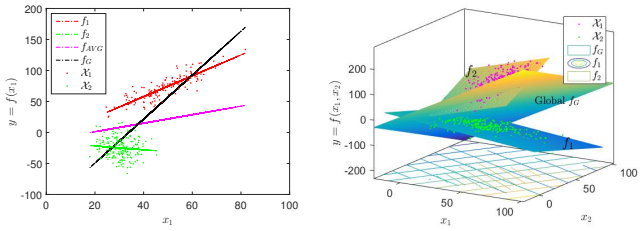

Fig. 3. (Left) Local linear models f_1 and f_2 built over data $\mathcal{X}_1$ and $\mathcal{X}_2$ from ED_1 and ED_2, respectively, global model f_G build over $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$, and average model $f_{AVG} = \frac{f_1 + f_2}{2}$; (right) Regression planes for local f_1, f_2 and global f_G models (f_{AVG} is not shown for readability).

B. Related Work & Contribution

Related Work: In centralized approaches [14], [17] all collected data are transferred centrally for analysis, thus, centralized regression modeling and maintenance suffer from heavy burden of massive data transfer and expensive fusion centers. In some cases, network nodes might not be willing to share their original data due to privacy issues. Our approach pushes regression analysis to the edge coping with the aforementioned constraints. In contrast, distributed approaches for parametric regression [16], [20], [3] focus explicitly on the distributed estimation of global model parameters over nodes, where the goal is to achieve the same prediction performance as the corresponding centralized one given that gathering all data centrally is expensive and/or impossible. Distributed regression (i) does not exploit data subspace locality and local models diversity (which are the key components in ensemble-based RA as evidenced above), (ii) focuses on training a pre-defined global

regression algorithm, where all involved nodes have agreed in advance, and (iii) requires extra techniques for parameters update and synchronization especially in real-time/adaptive RA. Such approach enforces nodes to adopt the same regression algorithm, which is not required in our approach providing the flexibility of hiring different regression models in EDs; our approach relies on the prediction performance of local models independently of the adopted regression algorithms on EDs. Recently, approaches for pushing analytics to the edge are proposed [12] either reduced to distributed parametric regression [16] (whose limitations are discussed above) or to selective data forwarding [15], [11], [13]. Specifically, [15] deals with time-optimized data forwarding among EDs and EGs in light of maximizing the quality of RA. Such approach reduces data communication in the network edge, however, data processing and model training are still built on EGs. This requires careful data transfer to control model maintenance & adaptation (see Figure 1). Our work further pushes model building, update and maintenance to the network periphery (EDs) thus avoiding completely data transfer (coping also with data privacy issues), while only parameters & sufficient statistics are *conditionally* disseminated for models adaptation and selection. The methods in [11] and [15] deal with data suppression based on local forecasting models on sensors in light of re-constructing data at the sink. However, they do not focus on regression/statistical dependencies learning at EDs (sensors) but only on reducing data communication via data suppression using forecasting models, also discussed in [13]. These models selectively disseminate data and univariate reconstruction models used at the sink, thus, actual regression modeling is achieved at the EG/sink with no guarantee on the analytics quality/prediction performance. Moreover, regression modeling does not scale since the EG lacks of model selection and caching mechanisms for selecting and maintaining the best models for RA, other than simple model averaging, whose limitations were discussed above and shown in Section V.

Contribution: To the best of our knowledge, this is the first edge-centric, on-line parametric regression analytics approach, which contributes to: (1) a novel input-error association statistical learning methodology and its mathematical analysis for extracting on-line sufficient statistics for delivery (Problem 3); (2) a communication efficient scheme that transmits only model parameters & sufficient statistics in the edge network for cached model updates in EGs (Problem 1); (3) novel model selection algorithms at EGs exploiting model statistics delivered by EDs (Problem 2); (4) comprehensive comparative assessment against current approaches of global and averaging regression and the methods in [15], [11] using real data.

III. Local Edge Device Intelligence

Local Decision Making. The ED i in Figure 1 locally learns a parametric regression model $f_i(\mathbf{x})$, e.g., $f_i(\mathbf{x}) = \mathbf{b}_i^\top(\mathbf{x})$, based on the recent local data in a sliding window $\mathcal{N}_i = \{(\mathbf{x}, y)_{t-N+1}, \ldots, (\mathbf{x}, y)_t\}$: $\mathcal{N}_i$ consists of the most recent N observed input-output pairs $(\mathbf{x}, y)$. Denote $\mathbf{b}_i$ the parameters of the current local model f_i and $\mathbf{b}_i^o$ the parameters of the cached local model f_i^o where ED i has already sent to EG at some time in the past. ED i is responsible for updating EG when there is a significant discrepancy of the prediction performance of the local f_i and cached f_i^o at EG. ED i keeps a copy of f_i^o locally to drive its decision making discussed later, where ED i sends only the parameters $\mathbf{b}_i$ if it is deemed necessary. This decision has to be taken in real-time by sequentially observing input-output pairs. Consider a discrete time domain $t \in \mathbb{T} = \{1, 2, \ldots\}$. ED i at time t captures the tth input-output pair $(\mathbf{x}, y)_t$ and, in real-time: **Case A:** Decides whether the pair $(\mathbf{x}, y)_t$ significantly changes the prediction performance of the current local f_i or not. In this case (A.I), ED i appends $(\mathbf{x}, y)_t$ to window $\mathcal{N}_i$ discarding the oldest pair and adjusts/re-train f_i accordingly based on the updated $\mathcal{N}_i$. Otherwise, (A.II), f_i is not adjusted/re-trained given $(\mathbf{x}, y)_t$. **Case B:** Decides whether the adjusted/re-trained local f_i (decided in case A.I) should be sent to EG or not. In this case (B.I), ED i updates EG with the up-to-date f_i provided that a significant prediction performance discrepancy is observed compared with the cached f_i^o. Otherwise, (B.II) no model update and delivery is performed between ED i and EG.

In Case A, ED i should be able to instantaneously determine whether the new pair is drawn from the input-output subspace defined by the pairs in $\mathcal{N}_i$ or not. In the former case, the new pair *interpolates* within the current input-output data subspace thus being considered as *familiar*. This familiarity indicates that the current model f_i is expected to provide a good prediction $\hat{y}_t = f_i(\mathbf{x}_t)$ given the tth input $\mathbf{x}_t$, i.e., $|y_t - \hat{y}_t| \leq \epsilon$ for some accuracy threshold $\epsilon > 0$. In this case, ED i does not need to adjust/re-train the current model f_i given that the tth pair is familiar (Case A.II), thus no communication with EG is needed.

If the tth pair is considered unfamiliar (*novelty*) w.r.t. the current input-output subspace, it renders a re-learning/adaptation of the current model f_i (case A.I). In general, a new local model f_i is derived after adaptation/re-training, thus, yielding ED i to examine the instantaneous model performance discrepancy between the new f_i and the cached model f_i^o (Case B). We quantify this discrepancy as the absolute difference of the prediction errors of f_i and f_i^o given the tth pair, i.e., $|e_i(\mathbf{x})_t - e_i^o(\mathbf{x}_t)|$. If this difference exceeds a discrepancy threshold $\theta > 0$, then ED i should update EG with the new model f_i and locally update the cached model $f_i^o = f_i$ (Case B.I). Otherwise, there is no need ED i to update EG, even if the cached and new models are not the *same*; *similarity* of these models is expressed by this discrepancy, i.e., f_i and f_i^o are considered similar if their prediction performance is the same w.r.t. θ. Hence, we enforce ED and EG to both have models that behave the same in terms of prediction.

Remark 2. *For parameters adaptation $\mathbf{b}_i$ upon a novel pair $(\mathbf{x}, y)$, one can employ either (A1) a sliding-window based batch re-training of f_i over window $\mathcal{N}_i$, or (A2) e.g., on-line/stochastic gradient descent (SGD) to incrementally update $\mathbf{b}_i$. For instance, in linear regression, in A1 case,*

$\mathbf{b}_i = (\sum_{l=1}^{N} \mathbf{x}_l^\top \mathbf{x}_l)^{-1}(\sum_{l=1}^{N} \mathbf{x}_l y_l)$ *if novelty pair* $(\mathbf{x}, y)$ *is inserted in* $\mathcal{N}_i$ *w.r.t. ordinary least squares optimization, while in A2 case,* $\mathbf{b}_i$ *is incrementally updated through SGD as* $\Delta\mathbf{b}_i = -\alpha(y - f_i(\mathbf{x}))\mathbf{x}$*;* $\alpha \in (0, 1)$*. The model update practice is beyond the scope of this paper; the reader could refer to [19] for efficient parametric regression adaptation.*

Local Model Delivery Mechanism. The challenge is to define an on-line method for assessing the novelty of a new pair, since based on this decision ED i can trigger a model updating process to EG. The novelty of an incoming $(\mathbf{x}, y)$ might trigger both: local model adaption and cache model update. The idea is to *incrementally* learn the k-th vector input subspace and *simultaneously* to associate the model prediction performance with that input subspace. To achieve this association, we need to on-line quantize (partition) the input space into K *unknown* subspaces, each one represented by an *input prototype* $\mathbf{w}_k \in \mathbb{R}^d, k \in [K]$ and then associate the prediction error $e(\mathbf{x}) = y - f_i(\mathbf{x})$ over input $\mathbf{x}$ lying around prototype $\mathbf{w}_k$ with an *error prototype* $u_k \in \mathbb{R}$; $k \in [K]$ is a compacted notation for $k = 1, \ldots, K$. That is, a new input $\mathbf{x}$ is firstly mapped to the closest $\mathbf{w}_k$ and then the corresponding error $e(\mathbf{x}) = y - f_i(\mathbf{x}) : k = \arg\min_{k \in K} \|\mathbf{x} - \mathbf{w}_k\|$ is summarized by the error prototype u_k. The rationale of this association is that we associate the *local* performance of f_i in the input subspace (represented by $\mathbf{w}_k$) with the *local* prediction error (represented by u_k). Prototype u_k provides local knowledge on the model accuracy in the subspace around $\mathbf{w}_k$, which will be further exploited to guide EG for model selection given a query, as will be shown later.

We propose a novel, fast and incremental input-error space quantization at ED i with unknown number of prototypes K. The objective joint optimization function in our case minimizes the combined (i) conditional Expected Quantization Error (EQE) in the input space, used for learning the best input prototypes representing novelty in input space, and (ii) conditional Expected Prediction Error (EPE) used for learning the best error prototypes capturing local model performance. The condition is based on the closest input prototype, i.e., we optimize the input/error prototypes $\mathcal{C}_i = \mathcal{W}_i \cup \mathcal{U}_i$, with $\mathcal{W}_i = \{\mathbf{w}_k\}$ and $\mathcal{U}_i = \{u_k\}$ to minimize the joint EQE/EPE:

$$\mathcal{J}(\{\mathbf{w}_k, u_k\}) = \mathbb{E}\big[\lambda\|\mathbf{x} - \mathbf{w}_k\|^2 + (1 - \lambda)|e(\mathbf{x}) - u_k|\big|\mathcal{A}_k\big] \quad (1)$$

where $\mathcal{A}_k \equiv \{k = \arg\min_{l \in [K]} \|\mathbf{x} - \mathbf{w}_l\|^2\}$, $e(\mathbf{x}) = |y - f_i(x)|$ is the absolute prediction error, and $\lambda \in [0, 1]$ is a regularization factor for weighting the importance of the input-error space quantization; $\lambda = 1$ refers to the known EQE [18], $\lambda \to 0$ indicates pure prediction-error based quantization; the expectation is taken over input-error pairs $(\mathbf{x}, e(\mathbf{x})) \in \mathbb{R}^d \times \mathbb{R}$.

Obviously, the number of prototypes K is not known a-priori and ED i incrementally decides *when* to add a new input-error prototype based on the input novelty and model performance. Hence, we propose an evolving algorithm that minimizes (1) starting initially with one ($K = 1$) input/error prototype pair $(\mathbf{w}_1, u_1)$ corresponding to the first input $\mathbf{x}_1$ and absolute prediction error $u_1 = |f_i(\mathbf{x}_1) - y_1|$ given the first pair $(\mathbf{x}_1, y_1)$. Then, current prototypes and new ones conditionally adapted and created, respectively, w.r.t. incoming pairs materializing the concept of *familiarity* and *novelty*, respectively. Specifically, based on a familiarity threshold ρ_I between the new input $\mathbf{x}$ and its closest prototype $\mathbf{w}_k$ and the dynamically changing error tolerance ρ_O for the current error $y - f_i(\mathbf{x})$, the pair $(\mathbf{x}, y)$ is classified as novel or not with the so far observed pairs. If the new pair is considered familiar w.r.t. recent history, the closest input prototype and corresponding error prototype are adapted to the familiar pair. However, if the current prediction error over the closest input subspace is not tolerated, i.e., greater than ρ_O, then this tolerance ρ_O decreases denoting less tolerance in the error space for future inputs. If input $\mathbf{x}$ is relatively far from its closest $\mathbf{w}_k$ w.r.t. ρ_I then a new input-error prototype is created. If the current prediction error is not tolerated, i.e., greater than ρ_O, then this pair is considered novel, which immediately renders the model re-learning/adaptation. Otherwise, this pair is familiar since the current error is tolerated, thus, avoiding model adaptation/re-training. Nonetheless, ρ_O decreases denoting less tolerance in the error space for future novel inputs.

The evolving Algorithm 1 minimizes (1) by incrementally adapting the input and error prototypes as stated in Theorem 2. Note, $\mathbf{w}_k$ and u_k converge to the centroid (mean vector) of the inputs $\mathbf{x}$ and to the median of the absolute prediction error in the k-th input-error subspace, as stated in Theorem 3. These (converged) prototypes are the sufficient statistics $\mathcal{C}_i$ (Problem 3), which will be exploited by EG for determining the most appropriate models given a query to EG (Figure 1).

Theorem 2 (Adaptation)**.** *The prototypes* $(\mathbf{w}_k, u_k) \in \mathcal{C}_i$ *minimize (1) iff given a pair* $(\mathbf{x}_t, y_t)$ *they are updated as:*

$$\Delta\mathbf{w}_k = \alpha_t\lambda(\mathbf{x}_t - \mathbf{w}_k) \quad \Delta u_k = \alpha_t(1 - \lambda)sgn(e_t - u_k), \quad (2)$$

$\alpha_t \in (0, 1)$ *is a learning rate:* $\sum_{t=1}^{\infty} \alpha_t = \infty$ *and* $\sum_{t=1}^{\infty} \alpha_t^2 < \infty$*,* $e_t = |y_t - f_i(\mathbf{x}_t)|$*, and* $\mathrm{sgn}(\cdot)$ *is the signum function.*

Proof. Proof is omitted due to space limitations. □

Theorem 3 (Convergence)**.** *The prototypes* $(\mathbf{w}_k, u_k) \in \mathcal{C}_i$ *converge to the centroid of input vectors and median of prediction error, respectively, of the* k*-th input-error subspace.*

Proof. Proof is omitted due to space limitations. □

The Algorithm 1 on ED i (i) optimally quantizes the input-error space by minimizing (1), (ii) on-line decides whether $(\mathbf{x}, y)$ is novel or not used for triggering model adaptation and/or cache model update, and (iii) incrementally evolves by identifying new prototypes in $\mathcal{C}_i$. It returns the updated statistics $\mathcal{C}_i$ and a classification of $(\mathbf{x}, y)$ as familiar or novelty. ED i decides on a cache model update given that $(\mathbf{x}, y)$ is novel. Since novelty might trigger a possible model modification, ED i is expected to obtain a new local model and assesses the performance difference with the cached model $|e_i(\mathbf{x}) - e_i^o(\mathbf{x})|$ given $(\mathbf{x}, y)$. Based on this difference ED i decides on sending to EG the new model for updating its cache. The ED i's local

decision making is shown in Algorithm 2; ED i has all the available knowledge for its input-error space encoded in $\mathcal{C}_i$.

Algorithm 1 On-line Local Algorithm at ED i

Input: new pair $(\mathbf{x}, y)$
Output: familiarity; updated prototypes $\mathcal{C}_i$

familiarity $\leftarrow$ FALSE
closest input prototype $k = \arg\min_{\ell \in [K]} \|\mathbf{x} - \mathbf{w}_\ell\|$
model prediction: $\hat{y} = f_i(\mathbf{x})$; absolute error $e = |y - \hat{y}|$
if ($\|\mathbf{x} - \mathbf{w}_k\| \leq \rho_I$) **then**
  prototype adaptation: $\Delta \mathbf{w}_k = \alpha \lambda (\mathbf{x} - \mathbf{w}_k)$
  prototype adaptation: $\Delta u_k = \alpha (1 - \lambda) sgn(e - u_k)$
  if $e > \rho_O$ **then**
    $\rho_O = \max(\frac{1}{2}\rho_O, \rho_O^*)$; adapt model f_i w.r.t. $(\mathbf{x}, y)$
  else
    familiarity $\leftarrow$ TRUE
  end if
else
  novelty (new prototype): $K = K + 1$, $\mathbf{w}_k = \mathbf{x}$, $e_K = e$
  if $e \leq \rho_O$ **then**
    $\rho_O = \max(\frac{1}{2}\rho_O, \rho_O^*)$; familiarity $\leftarrow$ TRUE
  else
    adapt model f_i w.r.t. $(\mathbf{x}, y)$
  end if
end if

Algorithm 2 Local Decision Making at ED i

Input: input-output observed pair $(\mathbf{x}, y)$

get pair $(\mathbf{x}, y)$ familiarity from Algorithm 1
if $(\mathbf{x}, y)$ is novel (not familiar) **then**
  append $(\mathbf{x}, y)$ in window $\mathcal{N}_i$; adapt/re-train model f_i
  model prediction error: $e_i(\mathbf{x}) = |y - f_i(\mathbf{x})|$
  cached model prediction error: $e_i^o(\mathbf{x}) = |y - f_i^o(\mathbf{x})|$
  if $|e_i(\mathbf{x}) - e_i^o(\mathbf{x})| > \theta$ **then**
    update EG with the new model f_i
    update cache model $f_i^o \leftarrow f_i$
  end if
end if

IV. Edge Gateway Intelligence

Up to this point, we have elaborated on how to learn the input-error space for ED i obtaining instantaneous feedback from local model f_i and generating the sufficient statistics (optimized parameters) $\mathcal{C}_i$. Such statistics are received by EG as a guiding light to select the most appropriate models per query. Our desideratum is that RA must be achieved in real-time with low communication overhead and be highly accurate. Communication overhead refers to the delivery of $\mathcal{C}_i$ and f_i from all ED i to EG and high accuracy refers to low prediction error given a regression query. EG caches all local models $\mathcal{F} = \{f_1^o, \ldots, f_n^o\}$ received from each ED i. Based on Algorithm 2, each ED i autonomously decides to update EG with the up-to-date local model f_i independently of the other EDs. Partial updates of the statistics $\mathcal{C}_i$ are sent to EG to significantly drive model selection. Note: EDs disseminate only *knowledge* (models and sufficient statistics) within the edge network and *not* actual data for RA tasks.

Assume that analysts/applications issue a query stream $\mathbf{q} \in \mathbb{R}^d$ to Cloud, which is directed to EG; see Figure 1. EG should return (i) accurate prediction $\hat{y}$ from the fused predictive model, and/or (ii) an inferential representation of the current local models around the input space defined by query point $\mathbf{q}$. And, these outcomes should be highly accurate and delivered in real-time without any further communication with the EDs. Hence, given a query $\mathbf{q}$, the challenge for EG is to (i) efficiently select the *most appropriate* subset of models $\mathcal{F}' \subseteq \mathcal{F}$ providing an ensemble prediction $\hat{y}$, whose prediction error is as close to the global f_G as possible and (ii) deliver the most representative models in $\mathcal{F}'$ that better explain the input-output dependency. We introduce model selection methodologies exploiting all knowledge coming from EDs. We model the ensemble prediction $\hat{y}$ as the weighted sum of the individual predictions $\hat{y}_i = f_i^o(\mathbf{q})$ from the cached models:

$$\hat{y} \quad = \quad \sum_{i=1}^{n} f_i^o(\mathbf{q}) \beta_i(\mathbf{q}). \tag{3}$$

The weight $\beta_i(\mathbf{q})$ in (3) is a function of $\mathbf{q}$ that interprets the importance of the performance of local model f_i in the local *familiar* input subspace around query $\mathbf{q}$ derived by the sufficient statistics $\mathcal{C}_i$. The $\beta_i(\mathbf{q})$ value drives the definition of $\mathcal{F}' \subseteq \mathcal{F}$ where EG engages *only* the models in $\mathcal{F}'$ for RA. We propose the following model selection methodologies:

Simple Model Aggregation (SMA): SMA does not exploit the statistics $\mathcal{C}_i$ received from EDs in the ensemble outcome. EG simply aggregates the individual predictions $\hat{y}_i = f_i^o(\mathbf{q})$ for deriving the final one thus setting $\beta_i(\mathbf{q}) = 1/n$: $\hat{y} = f_{AVG}(\mathbf{q}) = \frac{1}{n}\sum_{i=1}^{n} \hat{y}_i$. EG is *only* updated independently by a ED i upon a cache model f_i update, while no reception of $\mathcal{C}_i$ is required by any ED. The ensemble subset $\mathcal{F}' \equiv \mathcal{F}$, i.e., no model selectivity, where prediction accuracy is not favored compared to global f_G; see evaluation Section V.

Input-space Aware top-$\mathcal{K}$ Model (IAM): We first present the top-1 (best) model selection scheme ($\mathcal{K} = 1$). EG selects only one (best) model $f^* \in \mathcal{F}$ to engage RA, i.e., $\mathcal{F}' = \{f^*\}$ given query $\mathbf{q}$. The model selection is achieved by using prototypes $\{\mathbf{w}_{i,k}\}$ of the sufficient statistics $\mathcal{C}_i$ received at EG. IAM selects the model f^* whose the ℓ-th input prototype $\mathbf{w}_\ell^*$ is the closest to query $\mathbf{q}$ compared to *all* input prototypes in $\mathcal{W} = \{\{\mathbf{w}_{1,k}\}_{k=1}^{k_1} \cup \cdots \cup \{\mathbf{w}_{n,k}\}_{k=1}^{k_n}\}$ from *all* n models: $\mathbf{w}_\ell^* = \arg\min_{\mathbf{w} \in \mathcal{W}} \|\mathbf{q} - \mathbf{w}\|$. EG selects f^* whose input subspace (represented by $\mathbf{w}_\ell^*$) is the most familiar (closest) with query point $\mathbf{q}$, thus, the associated predictive model f^* can provide the best prediction. Without having obtained all input prototypes $\mathcal{W}_i$ from each f_i, EG could not discriminate which model's input subspace is the most familiar with the given query point. The weight function in IAM indicates the closest distance of $\mathbf{q}$ to the selected $\mathbf{w}_\ell^*$: $\beta_i(\mathbf{q}) = 1$ if $\exists \mathbf{w}_{i,k} \in \mathcal{W}_i : \mathbf{w}_{i,k} = \mathbf{w}_\ell^*$; 0 otherwise. EG engages only the f^* associated with the closest prototype for prediction, i.e.,

$\hat{y} = f^*(\mathbf{q})$. For $\mathcal{K} > 1$, EG ranks all prototypes $\mathbf{w} \in \mathcal{W}$ w.r.t. their distance from query $\mathbf{q}$ and selects those models $f_1^*, \ldots, f_{\mathcal{K}}^* \in \mathcal{F}' \subset \mathcal{F}$ whose closest input prototypes are ranked in the top-$\mathcal{K}$ closest distances. The ensemble prediction is then: $\hat{y} = \sum_{i=1}^{\mathcal{K}} f_i^*(\mathbf{q})\beta_i^*(\mathbf{q})$, where $\beta_i^*(\mathbf{q})$ is normalized to [0,1] w.r.t. the top-$\mathcal{K}$ inverse distances:

$$\beta_i^*(\mathbf{q}) = \frac{e^{-\|\mathbf{q}-\mathbf{w}_{i,\ell}^*\|^2}}{\sum_{l=1}^{\mathcal{K}} e^{-\|\mathbf{q}-\mathbf{w}_{l,\ell}^*\|^2}}. \quad (4)$$

The influence of the distance $\|\mathbf{q}-\mathbf{w}\|$, i.e., the closer to $\mathbf{q}$ the higher the weight importance, is achieved by the exponential inverse squared distance weighting $e^{-\|\mathbf{q}-\mathbf{w}\|^2}$.

Input/Error-space Aware top-$\mathcal{K}$ Model (IEAM): EG exploits all the knowledge from $\mathcal{C}_i, \forall i$ combining the familiarity of the input subspace of a model w.r.t. query $\mathbf{q}$ through the closest input prototype $\mathbf{w}_{i,\ell}$ and the associated performance reflected by the error prototype $u_{i,\ell}$. IEAM selects the best or the top-$\mathcal{K}$ best models from $\mathcal{F}$, which are not only familiar w.r.t. the queried input but *also* effective for providing accurate predictions based on their local prediction performance over the familiar subspace represented by the closest input prototypes to the query point. The combination of the two directions, input space familiarity and associated prediction performance, renders EG to proceed with more sophisticated model selection. The weight $\beta_i(\mathbf{q})$ represents a *degree of model closeness* to an issued query taking into consideration the (inverse) closest input distance $\mathbf{w}_{i,\ell} \in \mathcal{W}_i$ and the associated median of the absolute prediction error $u_{i,\ell}$ around this subspace. Specifically, $\beta_i(\mathbf{q})$ interprets the relative closeness of model f_i to query $\mathbf{q}$:

$$\beta_i(\mathbf{q}) = \frac{e^{-\|\mathbf{q}-\mathbf{w}_{i,\ell}\|^2}(1-\bar{u}_{i,\ell})}{\sum_{l=1}^{\mathcal{K}} e^{-\|\mathbf{q}-\mathbf{w}_{l,\ell}\|^2}(1-\bar{u}_{l,\ell})}, \quad (5)$$

where $\bar{u}_{i,k} = \frac{u_{i,k}}{\sum_{u\in\mathcal{U}} u}$ is the normalized median of the prediction error of model f_i over the k-th input/error subspace among all error medians $\mathcal{U} = \{\{u_{1,k}\}_{k=1}^{k_1} \cup \cdots \cup \{u_{n,k}\}_{k=1}^{k_n}\}$ from *all* n models. The prediction outcome is achieved by selecting $\mathcal{K} \geq 1$ models from $\mathcal{F}$ with the top-$\mathcal{K}$ high degrees of closeness of the $\mathcal{K}$ models ranked by $\beta_i(\mathbf{q})$, i.e., $\hat{y} = \sum_{i=1}^{\mathcal{K}} f_i(\mathbf{q})\beta_i(\mathbf{q})$ where $\beta_i(\mathbf{q})$ is provided in (5).

V. Performance & Comparative Assessment

Experimental Setup & Metrics. We evaluate and compare the performance of SMA (f_{AVG}), IAM, and IEAM with the Global (f_G) (centralized method) and the models in [11] and [15] over real data from EDs/sensors in Intel Berkeley Research lab[1]. We use two EGs with $n = 25$ EDs each; each ED senses 3-dim. vectors of temperature, humidity and light (2.3 million values for each in 36 days) every 31s. Each ED i learns a linear regression model $y = f_i(\mathbf{x}) = \mathbf{b}_i\mathbf{x}^\top$ over a sliding window of $N = 120$ vectors (1 hour history) with $d = 2$-dim. input $\mathbf{x} = [x_1, x_2]$ (x_1=humidity, x_2=light) and predicts output y=temperature adapted in Algorithm 1 and generating statistics $\mathcal{C}_i$. The learning rate $\alpha = 0.1$ [17] and regularization $\lambda = 0.5$ in (1) for putting equal importance of EQE and EPE. The familiarity threshold ρ_I is normalized in the input domain $[0,1]^d$, i.e., $\rho_I/\sqrt{d} \in (0,1)$; a value close to 1 refers to coarse vector quantization, thus a few prototypes K, while close to 0 refers to fine-grained quantization, thus many prototypes K. For updating the model parameters, in each ED i the discrepancy threshold $\theta_i = \gamma MED_i$ where factor $\gamma \in (0,3]$ and MED_i is the median of the error differences $|e_i(\mathbf{x}) - e_i^o(\mathbf{x})|$ in Algorithm 2 to control the expected communication between ED and EG. Based on θ_i, the initial error tolerance $\rho_O = \theta_i$ with minimum $\rho_O^* = \frac{\theta_i}{20}$. The performance metrics are: (i) percentage of the expected communication of all the models compared to the Global model: baseline solution of sending all raw data towards the EGs to construct a global model f_G, and (ii) prediction accuracy of RA per query measured by: Root Mean Squared Error $RMSE = [\frac{1}{M}\sum_{m=1}^{M}(\hat{y}_m - y_m)^2]^{1/2}$ and Mean Absolute Error $MAE = \frac{1}{M}\sum_{m=1}^{M}|\hat{y}_m - y_m|$. We examine the communication savings vs. Global model by just sending model parameters & statistics towards EGs instead of raw data, while measuring RMSE and MAE to assess the accuracy of SMA, IAM, IEAM, and DBP [11], HOVF [15] over the same $M = 3000$ regression queries. HOVF and DBP use linear forecasting models to predict ED's data, one model per attribute independently, and compare the predicted values with the current ones. If the difference less than a value tolerance then: DBP remains idle while HOVF decides whether to send only data to EG or not. Otherwise, DBP builds a new forecasting model per attribute and transmits the new models and data to EGs. In HOVF and DBP, f_i regression models are built in EGs. The window size and value tolerance for DBP and HOVF is $N = 120$ and as θ_i, respectively, for the sake of comparison.

Performance & Comparative Assessment. We assess our initial hypothesis where knowing the best local model f_i to involve per query $\mathbf{q}$ is unknown since we cannot know if $\mathbf{q} \sim \mathcal{X}_i$, while the Global model performs second best after the known local model with the SMA being the least accurate (Figure 2(left)). Fig.4(left) shows the MAE differences of IAM, IEAM, SMA and Global models compared to the known best local model f_i: Using IEAM, 72% of the cases obtain the same accuracy with the Global, IAM achieves same accuracy in 52% of the cases, while SMA obtains 16% of the cases with the same accuracy as Global. This indicates the capability of IEAM and IAM to identify the most appropriate local models *per query* in EGs without raw data transfer to EGs thus being communication efficient and generating as accurate predictions as the Global. Figure 4(right) shows the familiarity ratio $\rho_I/\sqrt{d}$ against number of prototypes K per ED; increasing the ratio towards 1 decreases K being negative exponential indicating the minimum storage requirement on EDs retaining prototypes for achieving accurate predictions as Global without transferring data. We set $\rho_I/\sqrt{d} = (0.05, 0.1)$ obtaining on average $K = (32, 18)$ per ED. We examine the impact of discrepancy threshold θ (via factor γ) on reducing

[1] http://db.csail.mit.edu/labdata/labdata.html

the communication between EDs and EGs and on RMSE. Fig.5(left) shows the robustness of IEAM ($\mathcal{K} = 1$,$\rho_I = 0.1$) and IAM ($\mathcal{K} = 1$,$\rho_I = 0.1$; $\mathcal{K} = 2$,$\rho_I = 0.05$) compared to SMA by increasing γ, which indicates less communication for model updates thus higher RMSE compared to Global. IAM and IEAM achieve significant lower RMSE towards Global for $\gamma < 1.5$, while SMA obtains, for all γ, high RMSE. To better illustrate the *efficiency* of IAM and IEAM trading-off RMSE with communication, Fig.5(right) shows a significant 80% decrease in communication for IEAM, which achieves RMSE slightly higher than Global. Note: the increase of RMSE and communication reduction in IAM and IEAM are not highly correlated, as the error with high communication results is nearly the same error than with nearly no communication. This indicates that EG identifies the best possible models for prediction based on the statistics $\mathcal{C}_i$ and only a few communication updates from EDs. The importance of statistics for finding the best models rather than simply averaging them is reflected by SMA, which cannot achieve low/comparable RMSE with the other models, even if it increases significantly the communication. Fig.6 shows the comparison of our models with HOVF and DBP in terms of accuracy and communication. IEAM is the most efficient achieving high accuracy with least communication exploiting the EDs' statistics $\mathcal{C}_i$; DBP ad HOVF are communication efficient but they do not account for the dependencies among attributes apart from selective data transfer, which has negative impact on RMSE. Finally, SMA achieves higher accuracy with less communication than Global, but does not consider the error behavior of the local model thus less accurate than IEAM.

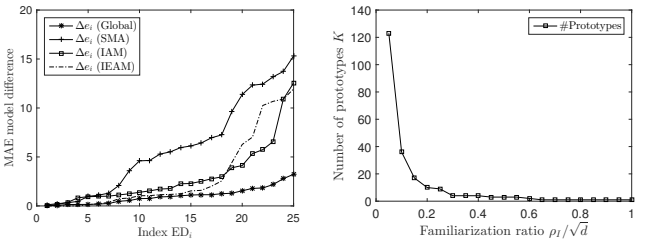

Fig. 4. (Left) MAE model differences over EDs; (right) Number of prototypes K vs. familiarization ratio $\rho_I/\sqrt{d}$.

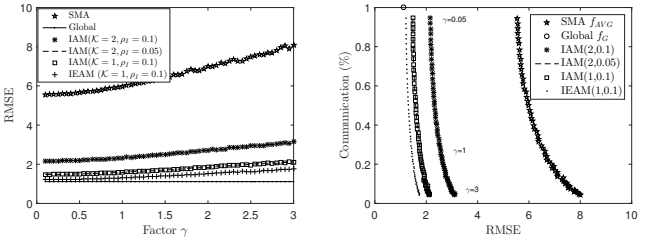

Fig. 5. (Left) RMSE vs. median factor γ; (right) communication vs. RMSE.

VI. Conclusions

A novel, edge-centric regression analytics methodology is introduced for on-line regression model caching and forwarding in the network edge being communication efficient. This is achieved by disseminating only model parameters and sufficient statistics instead of raw data, while the methodology introduces knowledge-driven model selection algorithms thus obtaining high analytics quality. Performance and comparative assessment with baseline models and models in the literature over real data evidenced its benefits in edge computing.

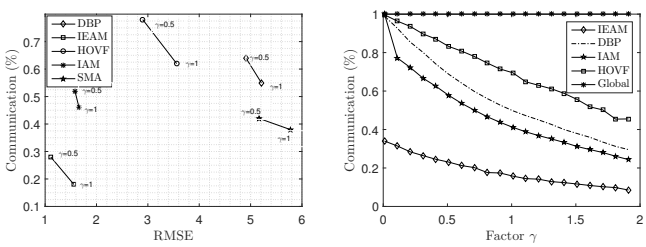

Fig. 6. (Left) Communication vs RMSE;(right) communication vs factor γ.

Acknowledgment

This research is funded by the EU/H2020/GNFUV Project (Grant#645220).

References

[1] E. G. Renart, J. Diaz-Montes, M. Parashar, 'Data-Driven Stream Processing at the Edge', IEEE ICFEC 2017, Madrid, pp. 31–40.

[2] P. M. Ferreira, A. E. Ruano, 'Online Sliding-Window Methods for Process Model Adaptation', IEEE TIM, 58(9):3012–3020, Sept. 2009.

[3] A. Lazerson, D. Keren, A. Schuster, 'Lightweight Monitoring of Distributed Streams'. ACM KDD 2016, NY, pp. 1685–1694.

[4] G. Cormode, 'The continuous distributed monitoring model', ACM SIGMOD Rec. 42(1):5–14, May 2013.

[5] R. Jain, S. Tata, 'Cloud to Edge: Distributed Deployment of Process-Aware IoT Applications', IEEE EDGE 2017, pp. 182–189.

[6] N. K. Giang et al, '"Developing IoT applications in the Fog: A Distributed Dataflow approach', IEEE IOT 2015, Seoul, pp. 155–162.

[7] X. Wei et al., 'MVR: An Architecture for Computation Offloading in Mobile Edge Computing', IEEE EDGE 2017, Honolulu, pp. 232–235.

[8] N. Anand, A. Chintalapally, C. Puri, T. Tung, 'Practical Edge Analytics: Architectural Approach and Use Cases', IEEE EDGE 2017, pp. 236–239.

[9] S. K. Sharma, X. Wang, 'Live Data Analytics With Collaborative Edge and Cloud Processing in Wireless IoT Networks', IEEE Access, 5:4621–4635, 2017.

[10] P. Garcia Lopez et al. 'Edge–centric Computing: Vision and Challenges', ACM SIGCOMM Comput. Commun. Rev. 45(5):37–42, 2015.

[11] U. Raza, A. Camerra, A. L. Murphy, T. Palpanas, G. P. Picco, 'Practical Data Prediction for Real-World Wireless Sensor Networks', IEEE TKDE, 27(8):2231–2244, Aug. 1 2015.

[12] G. Kamath et al. 'Pushing Analytics to the Edge', IEEE GLOBECOM, USA, 2016, pp. 1–6.

[13] N. Harth, C. Anagnostopoulos, D. Pezaros, 'Predictive intelligence to the edge: impact on edge analytics', Evolving Systems, 8, pp.1–24, 2017.

[14] Y. Kaneda, H. Mineno, 'Sliding window-based support vector regression for predicting micrometeorological data' ESWA J, 59:217–225, 2016.

[15] N. Harth, C. Anagnostopoulos, 'Quality-aware Aggregation & Predictive Analytics at the Edge'. IEEE Big Data 2017, Boston.

[16] H. Wang, C. Li, 'Distributed Quantile Regression over Sensor Networks', IEEE Trans. Signal Inf. Process. Netw., no. 99, 2016.

[17] L. Bottou, O. Bousquet, 'The tradeoffs of large scale learning', NIPS'07, 2007, pp.161—168.

[18] F. Shen, O. Hasegawa, 'An adaptive incremental LBG for vector quantization', Neural Networks, 19(5):694—704, June 2006.

[19] Y. Engel, S. Mannor, R. Meir, 'The kernel recursive least-squares algorithm', IEEE Trans. Sig. Proc. 52(8):2275–2285, Aug 2004.

[20] M. Gabel, D. Keren, A. Schuster, 'Monitoring Least Squares Models of Distributed Streams', ACM KDD 2015, NY, pp.319–328.