BibTeX:

Distributed Traffic Flow Prediction with Label Proportions: From in-Network towards High Performance Computation with MPI

Thomas Liebig THOMAS.LIEBIG@TU-DORTMUND.DE
University of Dortmund, 44221 Dortmund, Germany

Marco Stolpe MARCO.STOLPE@TU-DORTMUND.DE
University of Dortmund, 44221 Dortmund, Germany

Katharina Morik KATHARINA.MORIK@TU-DORTMUND.DE
University of Dortmund, 44221 Dortmund, Germany

Abstract

Modern traffic management should benefit from the diverse sensors, smart phones, and social networks data that offer the potential of enhanced services. In disaster scenarios, it is no longer guaranteed that a central server and reliable communication is always available. This motivates a distributed computing setting with restricted communication. Also in distributed High Performance Computing communication costs have to be reduced to the minimum and costly broadcast to all compute nodes hould be avoided. We want to learn local models with high communication efficiency. They still require the exchange of label information in a setting of supervised learning. The transmission of all labels among the nodes can be as costly as communicating all observations. Sophisticated methods are required to trade-off prediction performance against communication costs.

We hereby present an in-network algorithm based on local models that only sends label counts to neighboring nodes. Therefore the method is a novel approach that transfers no data about individual observations, but just aggregated label information. We outline its MPI implementation. And evaluate our approach on real world data in a traffic monitoring scenario. Tests reveal that in comparison to sending all labels, the algorithm is scalable.

Proceedings of the 2^{nd} International Workshop on Mining Urban Data, Lille, France, 2015.

1. Introduction

Traffic flow prediction is an important task for traffic managers. It allows performance assessment of major traffic infrastructure, like roads and junctions. Individual mobility benefits from predictions, as they provide necessary data for proactive, smart decisions on individual travel plans, e.g. predictive situation-aware trip planning by avoidance of likely traffic hazards (Niu et al., 2015; Liebig et al., 2014). Traffic flow models are based on sensor observations of current traffic gained by a mesh of (mostly static) presence sensors. While existing learning methods centralize and process measurements on a dedicated traffic management server, they have some major drawbacks, particularly in cases of disaster: The need for a reliable communication infrastructure reduces sustainability in case of natural hazards. First, the server-side collection causes high communication costs, decreasing the system's ability to process all sensor data, in time. Second, the area of traffic prediction systems is limited by the political area of homogeneous regulations for sending the data through the network. Third, increasing the network's density bares the risk of re-identification of individual persons and tracking them throughout the network. Existing systems are therefore limited by communication bandwidths, processing capabilities and political regulations.

We tackle these limitations by a distributed spatio-temporal in-network learning algorithm, where sensors compute local models and efficiently communicate label counts with their topological neighbors. Our approach sends space-time aggregated values that, by design, provide k-anonymity. Hence, our method is privacy preserving and can be applied for large-scale traffic management scenarios. Our particular focus is on the prediction of future traffic flow at junctions throughout the region of interest (e.g. a city, a state or even areas at European scale). Possible

applications comprise, for instance,

- distributed car-to-car scenarios where cars or trucks communicate at junctions the number of observed vehicles at the road to estimate traffic flow and alter their individual transportation plans based on predicted traffic conditions, or,
- large scale traffic flow prediction that processes massive local observations on a high performance computer.

Scalable in-network algorithms belong to the field of distributed data mining. Existing work mostly focuses on horizontally partitioned data. There, full observations, i.e. all features and labels, are stored on different nodes in a network. However, network states representing the current traffic flow are *vertically partitioned*. Here, only partial information about observations is stored on different nodes. Learning and prediction therefore either require the transmission of observations or labels to other nodes. Previous work (Das et al., 2011; Lee et al., 2012; Stolpe et al., 2013) has focused on sending less information about observations to a central coordinator. Here, we deal with reducing the amount of labels sent to neighboring peer nodes. Communication-efficient algorithms for vertical distributed learning are not just relevant for traffic flow prediction, but for applications as diverse as intrusion detection, monitoring production processes or smart grid management. The main contributions of our work are the following:

1. We introduce a privacy-preserving approach for the distributed learning of spatio-temporal prediction models which transfers only aggregated label information, but *no* data about individual observations.
2. A connection is drawn between the task of learning from label proportions and reducing communication costs in distributed environments, and it is evaluated on real-world data.
3. We introduce a fast search strategy for the LLP algorithm (Stolpe & Morik, 2011) and demonstrate its prediction performance in the context of traffic flow prediction.

The next section reviews related work. Section 3 details our problem setting and introduces a novel approach for the in-network training of local models. Section 4 discusses learning from aggregated label information, discusses its implementation in using message passing interface (MPI), analyses its communication cost and aspects of privacy. Evaluations of our approach can be found in Sect. 5. We finish with conclusions and outlook on future work.

2. Related Work

Many distributed data mining algorithm learn from horizontally partitioned data, whereas our data is vertically partitioned. In this context, privacy-preserving SVMs like (Yunhong et al., 2009) are not scalable, since they send quadratic kernel matrices to a central server. Distributed optimization algorithms (Bellet et al., 2014) exchange predictions for each observation per iteration, potentially sending more than the entire dataset. So does a co-regularized least squares regression in (Brefeld et al., 2006). Communication-efficient anomaly detection algorithms (Das et al., 2011; Stolpe et al., 2013) combine local and global models, but are 1-class algorithms reducing data sent about observations, not labels. In (Lee et al., 2012), local support vector machine (SVM) models are trained, but all labels are sent by a central server.

Also in traffic flow prediction, most literature describes processes on central servers. There are two major ways to model traffic: using a simulation (Raney & Nagel, 2006) or applying an imputation model, trained on previous sensor measurements. Models are required for the estimation of traffic flow at locations not being observed at all. Such imputation is not the focus of our study, but the prediction of traffic flow at sensor locations. We point the interested reader to methods of simulation (e.g. cellular automaton (Raney & Nagel, 2006)) and model-based imputation (e.g. (Liebig et al., 2012)). Most learning-based traffic flow prediction methods analyse time series, where a popular model is based on auto-regressive integrated moving average (ARIMA) (Ahmed et al., 1979). Recently, an application of a Gaussian Markov Model was proposed in (Schnitzler et al., 2014), and more advanced graphical models, namely Spatio-Temporal-Random-Fields (STRFs), were applied to traffic modeling in (Piatkowski et al., 2013).

Distributed approaches comprise an approach that applies kNN and Gaussian Process Regression (Chen et al., 2014), on-line distributed prediction of traffic flow in a large-scale road network (Wang et al., 2014), distributed traffic modeling in a MapReduce framework (Chen et al., 2013), Mapreduce parallel multivariate regression (Dai et al., 2014) and MPI (Message Passing Forum, 1994) based high performance computation based on SVM (Yang et al., 2014). Few distributed approaches combine sketches of neighbouring sensors to get probabilistic estimates of the number of vehicles co-occurring at different locations. Instead of counting and re-identifying individual vehicles, we use aggregated quantities.

The task of learning from aggregated label information was first introduced in (Kück & de Freitas, 2005). Theoretical bounds have only recently been proven in (Yu et al., 2014). (Musicant et al., 2007) propose variants of existing algorithms. The SVM optimization problem has been

adapted to the setting (Rüping, 2010; Yu et al., 2013). Mean Map (Quadrianto et al., 2009) estimates the mean operator solving a system of linear equations, while (Patrini et al., 2014) extend it with a manifold regularization, outperforming both SVMs and Mean Map on standard datasets. A modified Kernel k-Means algorithm (Chen et al., 2009) minimizes the distance to the given label proportions by matrix factorization. Recent work learns Bayesian network (Hernndez-Gonzlez et al., 2013) and generative (Fan et al., 2014) classifiers. The LLP algorithm proposed in (Stolpe & Morik, 2011) first determines clusters and then tries to label them. LLP only has linear running time, while its prediction performance competes with the approaches in (Quadrianto et al., 2009; Rüping, 2010) and(Chen et al., 2009).

3. Distributed Learning of Spatio-Temporal Local Models

Given are m distributed sensor nodes $P_1, \ldots, P_m$. Each sensor node P_i delivers an infinite series of real-valued measurements $\ldots, v_{t-1}^{(i)}, v_t^{(i)}, v_{t+1}^{(i)}, \ldots$ for different time points $\ldots, t-1, t, t+1, \ldots$. Time spans between two measurements are equidistant, given a constant sample rate. Let t denote the current time of measurement, while $t-a$ and $t+a$ are time points a steps in the past and future. Each sensor node also has a spatial location.

Many traffic flow management tasks require the prediction of traffic flow categories, that are achieved by a discretization of raw values into distinct intervals (e.g. risk level assignment or decision for emergency traffic signals).The task, given the current time point t, is therefore to predict a label y from a set $Y = \{Y_1, \ldots, Y_l\}$ of distinct categories at some arbitrary node P_i at future time point $t+r$, based on the current and previous (raw) sensor readings at all or a subset of nodes $P_1, \ldots, P_m$.

We assume that for learning, measurements and labels are somehow recorded (see below) over a fixed-length time period. For the supervised training of prediction models, each node P_i thus provides a sequence $V_i = \langle v_1^{(i)}, \ldots, v_n^{(i)} \rangle$ of measurements, $v_j^{(i)} \in \mathbb{R}$, and a sequence $L_i = \langle y_1^{(i)}, \ldots, y_n^{(i)} \rangle$ of labels $y_j^{(i)} \in Y$.

Distributed Learning of Local Models

Instead of centralizing all data, we propose that each P_i records and stores its own measurements and labels. For predicting future traffic flow categories at node P_i, we restrict learning to P_i itself and c topological neighboring nodes around P_i. For instance, to learn and predict the future type of traffic flow at some street junction, considered are only measurements and labels recorded at the junction itself and at c junctions closest to it.

Before training, each P_i preprocesses measurements V_i as follows. A window of size p is slided over the series V_i with step size 1, storing all thereby created windows $x_t^{(i)} = \{v_{t-p+1}^{(i)}, \ldots, v_t^{(i)}\}$, $t = p, \ldots, n$ as rows in a dataset D_i. Let $N^{(i)} = \{n_1^{(i)}, \ldots, n_c^{(i)}\}$ be the set of indices for the c neighboring nodes around P_i. Based on the datasets $D_i, D_{n_1^{(i)}}, \ldots, D_{n_c^{(i)}}$ and labels L_i, we want to learn a *local* function (model) $f^{(i)}$ that, given windows $x_t^{(i)}, x_t^{n_1^{(i)}}, \ldots, x_t^{n_c^{(i)}}$ of sensor readings from node P_i and its neighbors, predicts the label $y_{t+r}^{(i)}$ at node P_i with horizon r correctly.

Interpreting windows $x_t^{(i)}, x_t^{n_1^{(i)}}, \ldots, x_t^{n_c^{(i)}}$ as features of a single observation x that should be classified, the data is *vertically partitioned*, since each neighboring node of P_i only stores partial information about x, i.e. a subset of features.

An obvious choice for the training of $f^{(i)}$ at P_i is to ask for the recorded measurements at each neighboring node, concatenate their columns at P_i and join the labels stored at P_i to the new dataset. The approach is more scalable than centralizing all data, since the number c of neighbors is fixed, avoiding the bottleneck problem of limited bandwidth. However, each node still needs to transmit *all* measurements to each of its neighbors, consuming at least as much energy per node as sending all data to a single server.

Therefore, we propose to send only label information from node P_i to its neighbors and to train models $f_0^{(i)}$ at node P_i and $f_{n_1^{(i)}}^{(i)}, \ldots, f_{n_c^{(i)}}^{(i)}$ at its neighbors. As model $f^{(i)}$ at node P_i, we propose a majority vote over predictions from itself and its neighboring nodes. All models are *local*, since they only consider measurements and labels of a fixed number of close topological neighboring nodes around P_i. Moreover, the approach works fully *in-network* without a central coordinator, since each node only communicates with its neighboring peer nodes. As learners at each node, one may consider supervised learners, like kNN, Decision Trees or SVMs. Considering the limited computational resources of sensor nodes, however, our evaluation in Sect. 5 is solely based on kNN.

Since the number of bits to encode all labels is often less than an encoding of all measurements, communication is saved by sending labels *from* P_i instead of measurements *to* P_i. However, supervised learning still requires individual labels for all observations. The question is if communication can be reduced even further, by sending fewer labels or aggregated label information to each neighboring node.

Semi-supervised (Chapelle et al., 2006) and active learning (Balcan et al., 2010) show that training on fewer labels may achieve a similar performance as training on all labels.

However, such methods do not preserve the privacy of the data, since they need individual labels of observations (see Sect. 4). Instead, we propose to send only *aggregated* label information, i.e. label counts, to neighboring nodes for learning.

4. Aggregation of Label Information

Before sending label information to each of its neighboring nodes, P_i divides its time-related sequence L_i of labels into consecutive batches $C_1^{(i)}, \ldots, C_h^{(i)}$ of a fixed size b (see Fig. **??**). It respects the prediction horizon r, such that each $C_j^{(i)}$ consists of labels from time point $t+(j-1)b+r$ to $t+jb+r$ and align correctly with time points of observations (i.e. windows of measurements) at other nodes. Let n be from here on the size of datasets D_i, i.e. the number of windows stored. Then, h is $\lceil n/bs \rceil$. For each batch j, labels $y \in Y$ are aggregated by counting how often they occur, and stored in a $h \times l$ matrix of label counts $Q^{(i)} = (q_{jd}^{(i)})$, where $q_{jd}^{(i)} = |\{y \in C_j | y = Y_d\}|$.

Let $P_{n_e^{(i)}}$ be a neighboring node receiving label counts from P_i. $P_{n_e^{(i)}}$ transforms $Q^{(i)}$ into a label proportion matrix $\Pi^{(i)} = (\pi_{jd}^{(i)}) = q_{jd}^{(i)}/b$, i.e. the counts of labels are divided by batch size b. Since every node knows b and r, $P_{n_e^{(i)}}$ can partition its own windows $x_1^{n_e^{(i)}}, \ldots, x_n^{n_e^{(i)}}$ of measurements into batches $B_1^{n_e^{(i)}}, \ldots, B_h^{n_e^{(i)}}$. Since the sender respects r, the time spans used for aggregating the labels align correctly with the windows of measurements stored at $P_{n_e^{(i)}}$.

The learning task at node $P_{n_e^{(i)}}$ now consists of learning a model $f_{n_e^{(i)}}^{(i)}$, only based on its batches of (unlabeled) measurements and the label information from node P_i, stored in the label proportion matrix $\Pi^{(i)}$, such that the expected prediction error over individual observations is minimized. This task is also known as *learning from label proportions*.

Several methods have been developed to solve the task (see Sect. 2). Considering the limited computational resources of sensor nodes, the LLP algorithm (Stolpe & Morik, 2011) looked most promising for our evaluation in Sect. 5, since LLP has a linear running time and its centroid model a small memory footprint. Moreover, it can handle multiclass classification problems as they arise in traffic monitoring. However, we found that it still needs to be improved for scalability issues and performance. The next section describes LLP shortly, while Sect. 4 introduces a new local search method.

The LLP Algorithm

LLP learns from label proportions by first clustering all observations and then assigning labels to each cluster. The task of cluster analysis consists of partitioning a set of observations into a set $\mathcal{C}$ of k disjunct groups (clusters) $C_1, \ldots, C_k$, such that the similarity of observations in each cluster is minimized. LLP relies on the idea that observations having the same class also share similar features, i.e. that clusters somehow correspond to classes. LLP allows for several clusters per class and assumes that the majority of elements of a cluster belongs to the same class. Once given a clustering the only remaining problem is to assign correct labels to each cluster.

More formally, let $\mu : X \to \mathcal{C}$ be a mapping that assigns an arbitrary observation $x \in X$ to a cluster $C \in \mathcal{C}$. For centroids $c_1, \ldots, c_k$ found with k-Means, $\mu(x)$ would be defined as $\mu(x) = \text{argmin}_{C_k \in \mathcal{C}} ||x - c_k||^2$.

Further, let $\ell : \mathcal{C} \to Y$ be a mapping which assigns a label $\lambda \in Y$ to each cluster $C \in \mathcal{C}$. For ease of notation, let f denote model $f_{n_e^{(i)}}^{(i)}$ to be learned at node $P_{n_e^{(i)}}$, B_i denote the batch $B_i^{n_e^{(i)}}$ and Π denote matrix $\Pi^{(i)}$. f is the composition of mappings ℓ and μ, i.e. $f = \ell \circ \mu$.

With prediction model f, entries γ_{jd} of a model-based proportion matrix $\Gamma_f = (\gamma_{jd})$ can be calculated as

$$\gamma_{jd} = \frac{1}{|B_j|} \sum_{x \in B_j} I(f(x), Y_d), \quad I = \begin{cases} 1 & : \quad f(x) = Y_d \\ 0 & : \quad f(x) \neq Y_d \end{cases} . \tag{1}$$

The LLP algorithm now minimizes the mean squared error

$$\text{MSE}(\Pi, \Gamma_f) = \frac{1}{hl} \sum_{j=1}^{h} \sum_{d=1}^{l} (\pi_{jd} - \gamma_{jd})^2 , \tag{2}$$

between the given label proportion matrix Π and the model-based proportion matrix Γ_f by trying different label mappings ℓ.

A Local Search Strategy with Multistarts

The LLP algorithm as introduced in (Stolpe & Morik, 2011) can work with different cluster algorithms and labeling strategies. LLP with an exhaustive labeling strategy, called LLP_{exh} in the following, tries all possible labelings of the clusters. We found it too time-consuming for the evaluations done in Sect. 5. The greedy strategy proposed in (Stolpe & Morik, 2011) didn't achieve sufficient accuracies for traffic prediction. Hence, a better search strategy is demanded.

We propose a local search that is started multiple times with different random combinations of labels. LLP with this search strategy will be called LLP_{lsm} in the following. The local search greedily improves on the current labeling of clusters by trying all possible labels at each component of a labeling vector $\boldsymbol{\lambda}$. Fitness measures how well the model-

based label proportion matrix Γ_f, as calculated from the current labeling, matches the given label proportions. If the fitness improves, the search starts from the first component of the labeling vector $\boldsymbol{\lambda}$, again. Otherwise, it resets the label at the current position $kpos$ to the label of the best (local) solution found so far. Returned is the best labeling found over all starts of the different greedy searches.

In each iteration, the greedy search runs until no further improvement is possible. Moreover, at each step of the algorithm, the fitness either improves or is staying the same (which is a stopping criterion). Therefore, each search finds a local minimum. Since the number of searches is finite, the returned labeling vector is also locally minimal. In comparison to LLP_{exh}, it cannot be guaranteed that a globally optimal solution is found. However, with regard to the prediction results presented in Sect. 5, we found that a local search performed sufficient enough, despite a much lower running time.

LLP as introduced in (Stolpe & Morik, 2011) combines the MSE with two other error measures. However, we found that the use of these additional measures decreases the accuracy in the traffic monitoring scenario. Hence, all experiments in Sect. 5 are based on the MSE, only. Similarly, we abstain from the evolutionary feature weighting presented in (Stolpe & Morik, 2011), since it would heavily increase the algorithm's running time.

MPI IMPLEMENTATION

We explicitly focus on the implementation with the Message Passing Interface (MPI) (Message Passing Forum, 1994) as message passing in the top super computers in the top500 list[1] base on the MPI standard. MPI implements the single program multiple data paradigm (Darema, 2001) whereas every node of a distributed system executes the same program but uses different data. To coordinate this architecture a MPI program consists of 1 master node and multiple slave nodes that perform computations, the results are collected at the master. The art of MPI programming is to divide the problem into multiple tasks that are transferred to the slaves and processed thereby. Usually, the number of tasks exceeds the number of slave nodes and the distribution of the tasks has to be organized by the master.

The work in (Nupairoj & Ni, 1994) analyses the performance of such MPI systems and reveals that communication is major bottleneck in MPI programs. A succeeding publication (Piernas et al., 1997) provides empirical estimates for computation of communication costs depending on message lengths. Based on this publications two major conclusions ban be made: (1) The shorter the messages the lower the communication cost, and(2) broadcast messages should be avoided. Our algorithm respects both findings and therefore seems suitable for an MPI implementation. For ease of development we decided for the Cran-R package Rmpi (Yu, 2002) which provides basic MPI functionalities in Cran-R, thus matrix operators can be applied to the data. Our program comprises the following generic steps:

1. Load Rmpi, and spawn slaves
2. Definition of the functions for the master
3. Definition of necessary functions for the LLP algorithm at the slave
4. Initialization of the data
5. Send required data and functions to the slaves
6. Tell slaves to execute their function
7. Communicate with the slaves to perform computation
8. Collect the results
9. Close slaves and quit

For the learning from label proportions, our implementation presumes a shared network file system and initially processes the data at the master such that the sliding windows of the measurements are stored as Robjects on the file system. Every task gets its pointer to the corresponding slice of data and the label proportions of neighbouring nodes. The LLP algorithm is executed in every task and the trained models (cluster centers and their labels) are again stored physically for later re-use. This also allows the deployment of the parallel learned models in embedded devices or the future application of the label proportion models in high performance computation settings. Next subsection analyses the communication cost of LLP in comparison to kNN algorithm.

ANALYSIS OF COMMUNICATION COSTS

Each node P_i transmits a matrix Q to each of its neighboring nodes, consisting of counts for each label $Y_d \in Y$ and batch. Such counts may be assumed to be integers. The maximum value of each integer is b, which means we need to reserve at most $\lceil \log_2 b \rceil$ bits for each label. The number of batches, given n observations, is $\lceil n/b \rceil$. The total number of bits z_{AGG} for encoding matrix Q is therefore

$$z_{\text{AGG}} = \left\lceil \frac{n}{b} \right\rceil \lceil \log_2 b \rceil |Y| \quad . \tag{3}$$

In comparison, the number of bits z_{ALL} required to encode *all* labels of n observations, for $|Y|$ different labels, is at most

$$z_{\text{ALL}} = n \lceil \log_2 |Y| \rceil \quad . \tag{4}$$

[1] http://www.top500.org/lists/2014/11/, last accessed May, 1st

The total costs are then either z_{AGG} or z_{ALL}, multiplied by the number of nodes m. Here we assume that label information is broadcast to each neighboring node, which is not unrealistic for sensors in topologically close regions. All payloads reported in Sect. 5 base on this assumption.

Analysis of Privacy

The *vulnerable data* are the original sensor readings. These traffic flow measurements bare the risk of re-identification of individual vehicles. For example in a dense sensor network with sparse observations of vehicles, their occurrence may be tracked throughout the network. As mobility often is a regular behaviour and contains patterns this risk is even higher. In this section we show that our LLP_{lsm}-based algorithm transforms the data such that re-identification risk is at most $1/s$.

In our distributed setting, *adversaries* of a particular sensor node are malicious sensors that could use received measurements of neighboring sensors for deduction of individual mobility traces. The following *attack model* is possible: The adversary analyses differences among neighboring sensor readings and deduces individual movement. If the difference among two neighboring sensor readings is zero and both traffic flow counts are w, it is (depending on network topology) likely that w vehicles moved between the two sensors. In case of three neighboring sensors P_a, P_b, P_c their measurements v_a, v_b, v_c can be combined as follows: If $v_a - v_b = w = v_c$ it may be deduced that on the way from P_a to P_b w vehicles turned to P_c, in case $v_a - v_b = -w = -v_c$ w vehicles originated from the location P_a.

With our new LLP_{lsm}-based approach we process discretized traffic flow values and just communicate counts of these value ranges. We denote the minimal (nonzero) interval width by s. Thus, measurements may not be distinguished up to a granularity of s vehicles and w is bounded by s, $w \geq s$. In turn, the risk of re-identification with the hereby described attack model is at most $1/s$. Our approach therefore provides s-anonymity by design. The aggregation of label information reduces the remaining risk for disclosure of neighboring labels at a malicious sensor node. The solely transmission of label counts prevents doubtless reconstruction of the labels (Yu et al., 2014).

5. Experiments

We perform tests of the method on data of the city of Dublin. The Sydney Coordinated Adaptive Traffic System (SCATS) provides information on vehicular traffic at over 750 fixed sensor locations as spatio-temporal time series (McCann, 2014). The data we use[2] is a snapshot from 01/01/2013 till 14/05/2013, consisting of tuples (t, u, w), where u is the location of the observation and consists of an index for the junction, the arm and the lane number at which the sensor is located at. The metric w contains the aggregated vehicle count at sensor location since last measurement. The time stamp t denotes the recording time.

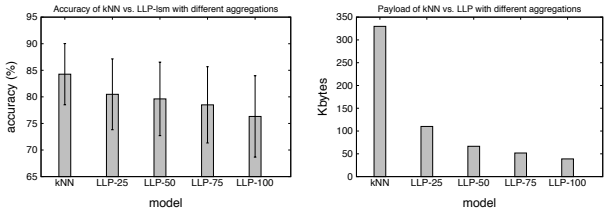

Figure 1. Trade-off between accuracy and payload sent for kNN and LLP_{lsm}

Local models are trained for each of the 296 sensor nodes and their nearest topological neighbors. As supervised base-line learner that receives all labels, we use kNN with $k = 15$. For learning from aggregated label counts, we cluster the observations at each node with k-Means ($k = 15$, 50 different random starting points, 500 iterations at maximum) and label the clusters with LLP_{lsm} (with 150 starts of the local greedy search) at each node for different batch sizes $b = 25, 50, 75$ and 100. The accuracy of each method is assessed by a 10-fold cross validation, i.e. all models are trained and evaluated for different hold-out sets 10 times. In total $296 \times 7 \times 10 = 20,720$ models for kNN need to be evaluated and $296 \times 7 \times 10 \times 4 = 82,880$ models trained and evaluated for LLP_{lsm}. The evaluation has been done offline in parallel on different machines (about 36 CPU cores).

Figure 1 shows the trade-off between accuracy and payload sent for kNN and LLP_{lsm} trained on differently sized batches of aggregated labels. Besides the average accuracy over all 10-fold cross-validations at each node, the bars in Fig. 1 (left) also depict the standard deviation of accuracy over all nodes.

In general, LLP_{lsm} performs slightly worse than kNN. Nevertheless, there are still many junctions for which the traffic flow is predicted quite well with LLP_{lsm}. Some locations have bad performance with both methods, a comparison to the map reveals that these are locations of parking areas e.g. inner-city parking houses and recreational areas where many vehicles stay for a long period of time.

6. Conclusions

The task of scalable traffic flow prediction involves a trade-off between the accuracy of models and the amount of communication between networked nodes. Especially in high

[2] Data is publicly available at http://dublinked.ie .

performance computation and embedded devices communication is costly.

In this paper we presented a novel approach for local models that trades-off communication costs to prediction accuracy which is suitable for in-network deployment and cluster computations.

Future work will focus on examining more sophisticated aggregation strategies for labels. We will study how to include dynamic distributed traffic flow prediction in state-of-the-art (multi-modal) route planning methods proposed by (Bast et al., 2014).

Acknowledgements

This research has received funding from the European Union's Seventh Framework Programme under grant agreement number FP7-318225, INSIGHT. Additionally, this work has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876, project B3. We thank Jan Czogalla for data preprocessing.

References

Ahmed, M.S., Cook, A.R., of Oklahoma. School of Civil Engineering, University, and Science, Environmental. *Analysis of Freeway Traffic Time Series Data Using Box and Jenkins Techniques*. 1979.

Balcan, M.-F., Hanneke, S., and Vaughan, J. W. The true sample complexity of active learning. *Machine Learning*, 80(2–3):111–139, 2010.

Bast, Hannah, Delling, Daniel, Goldberg, Andrew, Müller-Hannemann, Matthias, Pajor, Thomas, Sanders, Peter, Wagner, Dorothea, and Werneck, Renato. Route planning in transportation networks. Technical Report MSR-TR-2014-4, January 2014.

Bellet, A., Liang, Y., Garakani, A. B., Balcan, M.-F., and Sha, F. Distributed Frank-Wolfe algorithm: A unified framework for communication-efficient sparse learning. *CoRR*, abs/1404.2644, 2014.

Brefeld, U., Gärtner, T., Scheffer, T., and Wrobel, S. Efficient co-regularised least squares regression. In *Proc. of the 23rd Int. Conf. on Machine Learning (ICML)*, pp. 137–144, New York, NY, USA, 2006. ACM.

Chapelle, O., Schölkopf, B., and Zien, A. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

Chen, Cheng, Liu, Zhong, Lin, Wei-Hua, Li, Shuangshuang, and Wang, Kai. Distributed modeling in a mapreduce framework for data-driven traffic flow forecasting. *Intelligent Transportation Systems, IEEE Transactions on*, 14(1):22–33, 2013.

Chen, S., Liu, B., Qian, M., and Zhang, C. Kernel k-Means based framework for aggregate outputs classification. In *Proc. of the Int. Conf. on Data Mining Workshops (ICDMW)*, pp. 356–361, 2009.

Chen, Xing-Yu, Pao, Hsing-Kuo, and Lee, Yuh-Jye. Efficient traffic speed forecasting based on massive heterogenous historical data. In *Big Data (Big Data), 2014 IEEE International Conference on*, pp. 10–17, Oct 2014. doi: 10.1109/BigData.2014.7004425.

Dai, Liang, Qin, Wen, Xu, Hongke, Chen, Ting, and Qian, Chao. Urban traffic flow prediction: A mapreduce based parallel multivariate linear regression approach. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pp. 2823–2827, Oct 2014. doi: 10.1109/ITSC.2014.6958142.

Darema, Frederica. The spmd model: Past, present and future. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pp. 1–1. Springer, 2001.

Das, K., Bhaduri, K., and Votava, P. Distributed anomaly detection using 1-class SVM for vertically partitioned data. *Stat. Anal. Data Min.*, 4(4):393–406, 2011.

Fan, K., Zhang, H., Yan, S., Wang, L., Zhang, W., and Feng, J. Learning a generative classifier from label proportions. *Neurocomput.*, 139:47–55, 9 2014.

Hernndez-Gonzlez, J., Inza, I., and Lozano, J. A. Learning bayesian network classifiers from label proportions. *Pattern Recognition*, 46(12):3425–3440, 2013.

Kück, H. and de Freitas, N. Learning to classify individuals based on group statistics. In *Proc. of the 21th UAI*, pp. 332–339, 2005.

Lee, S., Stolpe, M., and Morik, K. Separable approximate optimization of support vector machines for distributed sensing. In *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *LNCS*, pp. 387–402, Berlin, Heidelberg, 2012. Springer-Verlag.

Liebig, Thomas, Xu, Zhao, May, Michael, and Wrobel, Stefan. Pedestrian quantity estimation with trajectory patterns. In *Machine Learning and Knowledge Discovery in Databases*, pp. 629–643. Springer Berlin Heidelberg, 2012.

Liebig, Thomas, Piatkowski, Nico, Bockermann, Christian, and Morik, Katharina. Predictive trip planning - smart routing in smart cities. In *Proceedings of the Workshops of the EDBT/ICDT 2014 Joint Conference (EDBT/ICDT 2014), Athens, Greece, March 28, 2014*, volume 1133, pp. 331–338. CEUR-WS.org, 2014.

McCann, Barry. A review of scats operation and deployment in dublin. In *Proceedings of the 19th JCT Traffic Signal Symposium & Exhibition*. JCT Consulting Ltd, 2014.

Message Passing Forum. Mpi: A message-passing interface standard. Technical report, Knoxville, TN, USA, 1994.

Musicant, D. R., Christensen, J. M., and Olson, J. F. Supervised learning by training on aggregate outputs. In *7th Int. Conf. on Data Mining (ICDM)*, pp. 252–261, 10 2007.

Niu, Xiaoguang, Zhu, Ying, Cao, Qingqing, Zhang, Xining, Xie, Wei, and Zheng, Kun. An online-traffic-prediction based route finding mechanism for smart city. *International Journal of Distributed Sensor Networks*, 501:970256, 2015.

Nupairoj, Natawut and Ni, Lionel M. Performance evaluation of some mpi implementations on workstation clusters. In *Scalable Parallel Libraries Conference, 1994., Proceedings of the 1994*, pp. 98–105. IEEE, 1994.

Patrini, G., Nock, R., Caetano, T., and Rivera, P. (almost) no label no cry. In *Advances in Neural Information Processing Systems 27*, pp. 190–198. Curran Associates, Inc., 2014.

Piatkowski, Nico, Lee, Sangkyun, and Morik, Katharina. Spatio-temporal random fields: compressible representation and distributed estimation. *Machine Learning*, 93 (1):115–139, 2013. ISSN 0885-6125.

Piernas, Juan, Flores, A, and García, José M. Analyzing the performance of mpi in a cluster of workstations based on fast ethernet. In *Recent advances in Parallel Virtual Machine and Message Passing Interface*, pp. 17–24. Springer, 1997.

Quadrianto, N., Smola, A. J., Caetano, T. S., and Le, Q. V. Estimating labels from label proportions. *J. Mach. Learn. Res.*, 10:2349–2374, 12 2009.

Raney, B. and Nagel, K. An improved framework for large-scale multi-agent simulations of travel behavior. *Towards better performing European Transportation Systems*, pp. 305–347, 2006.

Rüping, S. SVM classifier estimation from group probabilities. In *Proc. of the 27th Int. Conf. on Machine Learning (ICML)*, pp. 911–918, 2010.

Schnitzler, François, Liebig, Thomas, Mannor, Shie, and Morik, Katharina. Combining a gauss-markov model and gaussian process for traffic prediction in dublin city center. In *Proceedings of the Workshops of the EDBT/ICDT 2014 Joint Conference (EDBT/ICDT 2014), Athens, Greece, March 28, 2014*, volume 1133, pp. 373–374. CEUR-WS.org, 2014.

Stolpe, M., Bhaduri, K., Das, K., and Morik, K. Anomaly detection in vertically partitioned data by distributed core vector machines. In *European Conf. on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, pp. 321–336. Springer, 2013.

Stolpe, Marco and Morik, Katharina. Learning from label proportions by optimizing cluster model selection. In *Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part III*, ECML PKDD'11, pp. 349–364, Berlin, Heidelberg, 2011. Springer-Verlag.

Wang, Yubin, van Schuppen, Jan H., and Vrancken, Jos. On-line distributed prediction of traffic flow in a large-scale road network. *Simulation Modelling Practice and Theory*, 47(0):276 – 303, 2014. ISSN 1569-190X. doi: http://dx.doi.org/10.1016/j.simpat.2014.06.011.

Yang, Zhaosheng, Mei, Duo, Yang, Qingfang, Zhou, Huxing, and Li, Xiaowen. Traffic flow prediction model for large-scale road network based on cloud computing. *Mathematical Problems in Engineering*, 2014, 2014.

Yu, F. X., Kumar, S., Jebara, T., and Chang, Shih-Fu. On learning with label proportions. *CoRR*, abs/1402.5902, 2014.

Yu, F. X. Yu, Liu, D., Kumar, S., Jebara, T., and Chang, S. $\propto$SVM for learning with label proportions. In *Proc. of the 30th Int. Conf. on Machine Learning (ICML)*, pp. 504–512, 2013.

Yu, Hao. Rmpi: Parallel statistical computing in r. *R News*, 2(2):10–14, 2002. URL http://cran.r-project.org/doc/Rnews/Rnews_2002-2.pdf.

Yunhong, H., Liang, F., and Guoping, H. Privacy-preserving SVM classification on vertically partitioned data without secure multi-party computation. In *5th Int. Conf. on Natural Computation (ICNC)*, volume 1, pp. 543–546, 8 2009.