Accuracy vs. traffic trade-off of Learning IoT Data Patterns at the Edge with Hypothesis Transfer Learning

Lorenzo Valerio
Institute for Informatics and Telematics
National Research Council
Pisa, Italy
Email: lorenzo.valerio@iit.cnr.it

Andrea Passarella
Institute for Informatics and Telematics
National Research Council
Pisa, Italy
Email: andrea.passarella@iit.cnr.it

Marco Conti
Institute for Informatics and Telematics
National Research Council
Pisa, Italy
Email: marco.conti@iit.cnr.it

***Abstract*—Right now, the dominant paradigm to support knowledge extraction from raw IoT data is through global cloud platform, where data is collected from IoT devices, and analysed. However, with the ramping trend of the number of IoT devices spread in the physical environment, this approach might simply not scale. The *data gravity* concept, one of the basis of Fog and Mobile Edge Computing, points towards a decentralisation of computation for data analysis, whereby the latter is performed closer to where data is generated. Along this trend, in this paper we explore the accuracy vs. network traffic trade-off when using Hypothesis Transfer Learning (HTL) to learn patterns from data generated in a set of distributed physical locations. HTL is a standard machine learning technique used to train models on separate disjoint training sets, and then transfer the partial models (instead of the data) to reach a unique learning model. We have previously applied HTL to the problem of learning human activities when data are available in different physical locations (e.g., areas of a city). In our approach, data is not moved from where it is generated, while partial models are exchanged across sites. The HTL-based approach achieves lower (though acceptable) accuracy with respect to a conventional solution based on global cloud computing, but drastically cuts the network traffic. In this paper we explore the trade-off between accuracy and traffic, by assuming that data are moved to a variable number of data collectors where partial learning is performed. Centralised cloud and completely decentralised HTL are the two extremes of the spectrum. Our results show that there is no significant advantage in terms of accuracy, in using fewer collectors, and that therefore a distributed HTL solutions, along the lines of a fog computing approach, is the most promising one.**

I. Introduction

There is unanimous consensus in the research and industry communities about the fact that IoT applications will account for a quite significant (and increasing day-by-day) share of the Big Data that we will have to manage in the near future [1]. Many reference market analyses (e.g. [2]), show that IoT is one of the technologies (not only in the ICT domain) bound to have the biggest economic potential. As most of the value of IoT applications will come from the analysis of the data generated by IoT devices, the research area of IoT data analysis and management is a very challenging and exciting one.

The common trend in most of current architectures [3] is to transfer IoT data from the physical locations where they are generated, to some global cloud platform, where knowledge is extracted from raw data, and used to support IoT applications. This is the case, among others, of the ETSI M2M architecture [4]. However, there are concerns whether this approach will be sustainable in the long run. The projections of growth of the number of deployed IoT devices are exponential over the next years [2]. Together with data generated by personal users' devices, which are also likely to be part of IoT applications, this is likely to make the amount of data generated at the edge of the network huge, making it impractical or simply impossible to transfer them to a remote cloud platform. In addition, data might also have privacy and confidentiality constraints, which might make it impossible to transfer them to third parties such as global cloud platform operators. These trends push towards a decentralisation of cloud platforms towards the edge of the network, according to the Fog [5] and Mobile Edge Computing [6] paradigms.

In this paper we follow this approach, and study the behaviour of a distributed learning solution based on Hypothesis Transfer Learning (HTL). In general, in HTL, instead of training a model on the whole training set, multiple parallel models are trained on disjoint subsets, and then the partial models are combined to obtain a single final model. We have applied HTL to the case of distributed learning in IoT environments in [7], where we have presented an activity recognition solution. Specifically, in our use case data collected from users' personal devices are available in a number of disjoint physical locations, where partial learning models are trained. HTL is used to exchange and combine these partial models and obtain a unique model. In [7] we have shown that this solution is able to drastically cut the network traffic required to perform the learning task, with an affordable reduction of accuracy with respect to a conventional solution where data is transferred on a global cloud platform.

In this paper, we study the accuracy vs. network traffic trade-off of this solution, when a variable number of *Data Collectors* (DCs) is used. Specifically, we assume that disjoint

978-1-5090-1131-5/16/$31.00 ©2016 IEEE

parts of the complete dataset are collected at a number of DCs, where partial models are computed according to a HTL scheme. As a special case, when the number of DCs is equal to 1 we obtain the conventional cloud approach, while when the number of DCs equals the number of physical locations we obtain the completely decentralised HTL scheme defined in [7]. The rationale of this study is as follows. The fewer the DCs, the larger the dataset on which each partial model is trained. Therefore, we may expect that fewer DCs could result in more accurate partial models, and therefore on more accurate final models. On the other hand, the fewer the DCs, the higher the network traffic, as data would need to be moved further away from their initial locations.

To test our hypothesis, we consider both the same dataset used in [7], where data are initially available in 20 physical locations, and a new dataset with 27 physical locations to push our solution to its limits. For a given number of DCs, we define a network cost incurred in moving data from the physical locations, and run HTL on the DCs to compute the accuracy of the obtained data model. Comparison with the two extreme cases (completely decentralised HTL and centralised cloud solutions, respectively) show some interesting features. Specifically, we show that reducing the number of DCs only provides a marginal advantage in terms of accuracy over a completely decentralised HTL solution, sometimes hardly distinguishable from a statistical standpoint. On the other hand, the cost in terms of network traffic does increase as the number of DCs decreases. Therefore, the key findings from this study is that a decentralised data analysis solution, along the line of a fog computing paradigm, is the most suitable one, as it (i) drastically reduces the network traffic, and (ii) still guarantees a very high accuracy of the final data model.

II. Related work

The distributed learning problem can be solved in different ways. One typical approach is represented by the so-called "ensemble methods" such as bagging, boosting and mixtures of experts [8], [9], [10], [11]. Typically, these methods randomly split a dataset in smaller portions and allocate them to different classifiers that are independently trained. The individual classifiers are subsequently aggregated, sometimes after an additional training phase or with other techniques based on feedbacks coming from the training phase. Although these approaches are very powerful, they assume that the training set is available to a single coordinating processor.

Another very promising set of powerful solutions able to process huge amounts of data is represented by deep learning techniques [12], [13], [14]. These techniques are nowadays widely used to solve many complex tasks. Differently from our solution, these approaches do not target knowledge extraction where data have privacy constraints, or when network overhead should be minimised.

Other works propose fully distributed and decentralized learning algorithms. To the best of our knowledge, the more relevant with respect to our reference scenario are the following. In [15] authors present a distributed version of Support Vector Machine (SVM). Authors of [16] propose a distributed learning algorithm for arbitrary connected machines. Another similar solution presented in [17] propose two distributed learning algorithms for random functional link networks. Both are iterative solutions that in order to converge to a model have to repeatedly exchange their partial models' parameters. The purpose of all these solutions is to learn a model whose accuracy performance is close, or at least comparable, with a centralized algorithm with access to the complete dataset. Although the accuracy is a crucial evaluation term for such kind of solutions, also the network traffic triggered by those algorithm is fundamental, if we consider a network constrained scenario like the IoT. To the best of our knowledge, the only solution that takes into account both these aspects is [7].

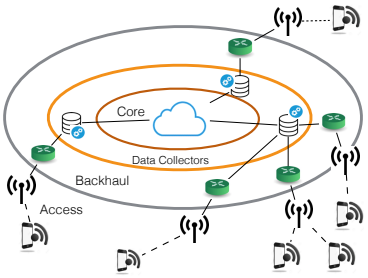


Fig. 1. Considered reference scenario.

In this paper, we extend the analysis initiated in [7] in order to study the accuracy vs. network traffic trade off. Precisely, we evaluate the accuracy of the solution based on Hypothesis Transfer Learning presented in [7] at different levels of data aggregation. Our aim is to evaluate to what extent the aggregation of data affects (positively or not) the accuracy of our solution and the cost in terms of network traffic associated to it.

III. Problem statement and system assumptions

In this section we present the reference scenario we consider in this paper. For the sake of clarity we will refer to Fig. 1. We assume an environment where the connectivity to all the smart devices is provided through wireless technology (e.g. WiFi, Bluetooth, etc.). In this scenario, smart devices equipped with a wide variety of sensors, are able to sense and collect data. In the considered scenario we assume the availability of a number of Data Collectors (DC) that are located in the core of the network. We assume the presence of many DCs responsible for different groups of locations where data are generated. Each DC receives and store data sensed by devices that are, at a give time, physically located in the area served by the DC. In our scenario, data is local, i.e. data collected in a specific location is stored on the closest DC and further data transfers are not allowed. Beyond storing data collected in a specific area, DCs perform data analysis tasks in order to build models from data. Since they build models on locally collected data only, in order to improve the accuracy of their own local models, DCs are allowed to share with each other the information about the partial models they built from data. Clearly, we assume that

978-1-5090-1131-5/16/$31.00 ©2016 IEEE

the data contained in each DC (from now on "data partitions") are homogeneous, e.g., for a sensing application, they must be related to a well-defined physical entity that is sensed across different locations. This is necessary to meaningfully use, for the model corresponding to one partition, features of the models trained on the rest of the partitions.

In this paper the learning task that all the DCs have to solve in a collaborative and distributed way is a classification problem. Specifically, we assume that each data partition located on a DC is a training set used to train a local classifier. The features of the local classifiers are exchanged with all the other DCs and used by each of the them to refine their own local classifier.

Our purpose is twofold. On the one hand we want that at the end of the distributed learning process, each DC has learnt a model from its own data, with a prediction accuracy equivalent to the one obtainable in a cloud settings, i.e. the case when the entire dataset is accessible from the cloud and the learning process is accomplished considering the union of all the local datasets. On the other hand we want to minimize the traffic load over the network (defined more precisely in the following). It is worth noting that from an architectural point of view, the scenario considered in this paper and shown in Fig. 1 represents an intermediate case among the typical cloud solution (CL) and a fully distributed one (DS). Precisely, in a cloud solution all the DCs ideally collapse to one DC located in the Core that holds the entire collected data. Conversely, in the fully distributed solution each DC corresponds to a device that senses and analyses its own data at the edge of the network. These two cases, from the network load point of view, correspond to the extreme limits of this architecture. In fact in the CL case, all the data sensed are transferred over the core network to be analysed, while in the DS case the data never leaves the smart device that collects it and, as in the case previously described, only partial models are sent over the network.

IV. Distributed Learning through Hypothesis Transfer Learning

In this section we briefly describe our distributed learning solution based on the Hypothesis Learning Framework (HTL).

HTL methods are machine learning algorithms that have the objective of finding a profitable way of exploiting the knowledge acquired by a model m_1 trained on a dataset D_1, in order to boost the accuracy of the learning process of a second model m_2 during its training phase on a different dataset D_2. In other words, the idea of HTL is to transfer the knowledge acquired by m_1 to the model m_2 in order to improve the performance of the latter. This kind of solution is typically used in situations in which the size of D_2 is largely smaller than the size of D_1. In fact in this situations, without a technique such as HTL, the accuracy of m_2 would be strongly affected by the small amount of data available to learn the model. In HTL, only models are exchanged and not data.

In this paper we exploit one of the reference solutions in the HTL literature [18] to design a distributed learning solution that fits the architecture presented in Section III. The HTL algorithm we use in this paper is GreedyTL [18]. Due to space reasons, we describe the main steps of our distributed learning solution, without providing all the mathematical details. A more detailed description of GreedyTL and how it is exploited to design the distributed learning solution evaluated in this paper, can be found in [18] and [7], respectively.

We assume that a dataset D is distributed over a number L of Data Collectors. Therefore each DC l holds a partition $D_l, l = 1, \ldots, L$ of D. Our distributed learning solution is composed by 5 steps that are performed by each DC.

Step 0: In the first step each DC trains a classifier on its local data. At this step there is not a unique choice for the learning algorithm to be used. In our paper we use a Support Vector Machine learning algorithm because of its good performance and simplicity for the data classification task that we target. After this first learning step, each DC l_i holds a model $m_{l_i}^{(0)}$.

Step 1: After Step 0, each DC l_i sends its model $m_{l_i}^{(0)}$ to all the other DCs. At the end of this phase each DC holds L SVM classifiers, each one trained on different data.

Step 2: Each DC performs a second learning phase on the same data, using the GreedyTL algorithm. In this step, the purpose of GreedyTL is to learn a classifier that includes the knowledge contained in the models $m^{(0)}$ sent by all the other DCs. Precisely, GreedyTL solves an optimisation problem in order to find the linear combination of models $m^{(0)}$ that produces a classifier with the best prediction accuracy. At the end of this step each DC has a new classifier $m^{(1)}$ that is the output of the GreedyTL algorithm.

Step 3: After the second learning phase, there is another synchronization phase where all the DCs exchange their newly learnt $m^{(1)}$ with each others.

Step 4: Once all the models have been received, each location aggregates all of them into a single model $m^{(2)}$. In this paper we adopted a consensus approach. Briefly, in the consensus approach, each model at each DC is computed as $h^{(2)} = \frac{1}{L}\sum_{l=1}^{L} \alpha_l m_l^{(1)}$, where α_l represents the weight of the $l-$th model computed as the accuracy on its training set. This represent a quite simple way to aggregate many models into one model taking into account how accurate is each model on its own local data. We used it because of its simplicity and its pretty good performance. Moreover, once computed the final average model, all the partial models computed at step 1 and held by each DC are no longer useful and can be removed in order to save storage.

V. Performance Evaluation

In this paper we take the Human Activity Recognition problem based on data collected through smart-phones' sensors as a representative example of a possible application of our distributed learning framework in an IoT environment. While activities to be recognised typically happen at each individual node, it is still useful to learn the models based on a larger training set with respect to what is available at each node. Specifically, we have shown in [7] that a model trained on the

978-1-5090-1131-5/16/$31.00 ©2016 IEEE

entire dataset is significantly more accurate than individual models trained locally on the sole datasets observed by each node. In addition, as we also show in this paper, through this approach it is possible to use the learned activity recognition model at new nodes without any prior training phase, obtaining very accurate results.

We used two real world datasets commonly used in the field of Human Activity Recognition (HAR). The first dataset (HAPT)[19] was built during experiments carried out with a group of 30 volunteers. Each volunteer performed a protocol of activities composed by six basic activities: standing, sitting, lying and three dynamic activities: walking, walking downstairs and walking upstairs. In the dataset are present also postural transitions that occurred between the static postures. Data have been recorded through smart-phones that participants were wearing on the waist during the experiment execution. The sensed data are from the smart-phones' embedded accelerometer and gyroscope. Each of these sensors produced raw measurements such as 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz. Raw sensor signals are pre-processed by applying noise filters and then sampled in fixed-width sliding windows. The dataset contains ~ 10000 records of 561 features for a total size of 48MB. The second dataset (WISDM) [20] we consider in this paper was built during experiments performed by twenty-nine volunteer subjects carrying a smart phone while performing a specific set of activities, such as walking, jogging, ascending stairs, descending stairs, sitting, and standing. Likewise in the HAPT dataset, raw signals ware collected by means of the smart-phone sensors' (GPS., accelerometers, etc) every 50ms. The collected raw data was divided into 10s segments in order to generate ~ 5000 activity records of 43 informative features. In this dataset the activities "Sitting" and "Standing" are not present for all the users therefore in our simulations these activities are not considered. It is worth noting that the amount of not considered data related to these two activities is so small (i.e. less that 10% of the entire dataset) that does not affect our results. Note that the size if this dataset is very small ($\sim 1.2MB$). However, as it will be clear in the following, we use it to show that there exists a limit beyond which exchanging models instead of aggregating data is no more beneficial.

In this paper we are interested in the analysis of the trade off between the accuracy obtained after the distributed learning process accomplished by the Data Collectors and the cost in terms of network load. Precisely, since the learning problem at hand is a classification problem, we measure the accuracy as *prediction loss*, which is defined as the mean number of prediction errors done by the classifier trained in the l-th Data Collector. More formally

$$\ell_l = \frac{1}{n}\sum_{i=1}^{n} I(y_i, \hat{y_i}) \tag{1}$$

with

$$I(y_i, \hat{y_i}) = \begin{cases} 1 & \text{if } y_i \neq \hat{y_i} \\ 0 & \text{otherwise} \end{cases}$$

where $\hat{y}$ is the predicted class (i.e. the kind of activity predicted by the model) of the i-th pattern x_i (i.e. the set of measurements sensed during the activity) and y_i is its true class (i.e. the real activity performed by a volunteer) and n is the cardinality of the test set. Accuracy performance are computed according the "leave one user out" procedure. Precisely, we used the data collected by 20 (HAPT) and 27 (WISDM) randomly selected users as training set and the data of one randomly selected user as test set. Reported results are average values computed on 10 runs. Moreover, we reported the confidence intervals at level 95% of confidence for the average values.

In addition to comparing the accuracy of HTL with a varying number of DCs, in the paper we also compared the accuracy results of our solution based on Hypothesis Transfer Learning with a similar approach that does not exploit the Hypothesis Transfer Learning. Precisely, the benchmark solution (hereafter called *no-HTL*) performs the Steps 0,1,4 of our distributed learning procedure skipping Step 2 and 3, i.e. the ones where all the base models are combined into one local model by the GreedyTL. Note that in order to have a fair comparison, in the benchmark approach we use the very same learning algorithm (SVM) used for our HTL solution.

The network overhead is the amount of traffic we generate to send data through the network. As it will be clear later on, HTL performs better than no-HTL in terms of accuracy in all the considered scenarios, therefore, here we focus on the network overhead generated only by HTL and we do not analyse the one generated by no-HTL. The network overhead is computed as follows:

$$OH_{tot} = OH_{data} + OH_{model} \tag{2}$$

where OH_{model} is the amount of traffic generated during the synchronization steps (Step 1 and 3) in which models are exchanged between DCs, and OH_{data} is the traffic generated to send the data from the smart-phones to the closest DC. Moreover, OH_{data} is defined as

$$OH_{data} = OH_{acc} + OH_{bkl}$$

where OH_{acc} is the cost for sending data over the access network and OH_{bkl} is the cost payed for sending data from Access Points at the edge of the network to DCs. We point out that the amount of traffic on the backhaul generated by sending the collected data to DCs is affected by a number of parameters. To keep the analysis simple, we assume that in case when a global cloud platform is used, this cost is comparable to the cost of transferring data over the access network. When the number of DCs increases, the cost to send the data to each DC linearly decreases (with the number of DCs), to take into account that DCs are closer to the locations where data are generated. More formally

$$OH_{bkl} = \rho OH_{acc}$$

where

$$\rho = 1 - \frac{L-1}{N-1}$$

978-1-5090-1131-5/16/$31.00 ©2016 IEEE

Here L is the number of DCs and N is the number of smart-phones that sense the data. Therefore, in case of a completely decentralised solution, when L equals N, OH_{bkl} is 0 because we do not generate any backhaul traffic related to the data. On the other hand, when we use a global cloud platform, i.e. $L = 1$, the backhaul traffic is considered the same as the access traffic.

In order to study to what extent the level of data aggregation expressed as the number of DCs affects the performance of our distributed learning solution, we performed the learning task varying the number of DCs and analysing the relation between the traffic generated and the corresponding accuracy obtained. Results for the HAPT dataset are reported in Table I.

TABLE I
HAPT DATASET. NETWORK OVERHEAD (MB) AND ACCURACY RESULTS.

L	OH_{acc}	OH_{bkl}	OH_{model}	OH_{tot}	Loss
20	0	0.	23.57	**23.57**	0.038
10	48	25.26	5.65	**75.91**	0.028
5	48	37.9	1.27	**87.17**	0.039
4	48	40.42	0.77	**89.19**	0.037
1	48	48	0	96	0.005

Note that the results on the first and the last row refer to the fully distributed and the cloud only scenarios, respectively. In fact we see that in the former case the network overhead is composed only by the models' exchange because the data remain on the smart-phones that collected it. Conversely, in the latter we see that the total amount of traffic is twice the size of the dataset. This is due to the fact that all the collected data are sent to the cloud in order to learn a model from data. The other rows show the traffic generated with different numbers of DCs . The first observation is that the traffic generated on the Access network is constant, while OH_{bkl} increases as the number of DCs decreases. This is quite straightforward if we consider that if we diminish the number of DCs, more data has to pass through the Core of the network. However, also OH_{model} is affected by the number of DCs, as with fewer DCs there are fewer models to be exchanged between them.

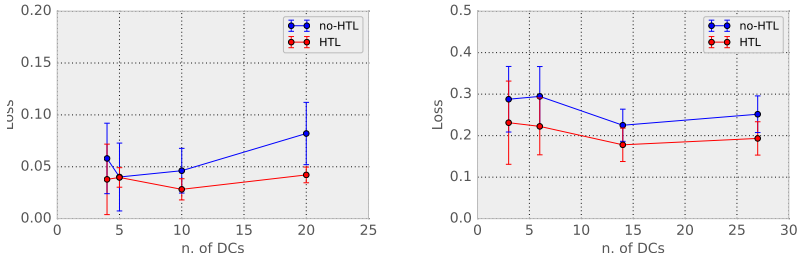
Regarding the accuracy reported in the last column of Table I we see that the performance of our distributed solution is not very sensitive with respect to the number of DCs. This suggests that it is possible to obtain very good learning performance also with the minimum network traffic, as also shown in Fig. 2a. Specifically, Fig. 2a compares the loss of HTL and no-HTL as functions of the number of DCs. Let us first focus on the HTL curve. We observe that the loss is statistically equivalent for any number of DCs (as confidence intervals overlap). This means that we can still obtain very good accuracy (equivalent to that obtained in a centralised cloud solution) also with the minimum network overhead, achieved for the maximum number of DCs. In other words, computing partial models over larger datasets (which is the effect of reducing the number of DCs) does not help. This might look surprising at first, as one would expect that the accuracy of the models when using fewer DCs might be higher, as models are trained over larger and larger datasets. However, overcoming the limitation of partial models trained over small datasets is exactly the main feature of HTL. The reason can be understood by comparing the HTL and the no-HTL curves. Specifically, note that in no-HTL the loss is in general higher, and, although confidence intervals still overlap, the loss with more DCs tends to increase. This is because in the HTL algorithm nodes exploit one more step of aggregation of the partial models, which allow them to better incorporate locally also knowledge extracted by the other partial models in the rest of the DCs. This overcomes the limitations of training the partial models on smaller datasets, ultimately resulting in high accuracy also with the maximum number of DCs.

TABLE II
WISDM DATASET. NETWORK OVERHEAD(MB) AND ACCURACY RESULTS.

L	OH_{acc}	OH_{bkl}	OH_{model}	OH_{tot}	Loss
27	0	0.	6.32	6.32	0.19
14	1.2	0.6	3.13	4.93	0.18
6	1.2	0.96	0.44	2.6	0.22
3	1.2	1.1	0.04	**2.34**	0.23
1	1.2	1.2	0	2.4	0.14

The analysis on the HAPT dataset suggests that (i) accuracy is basically independent on the number of DCs, and (ii) overhead is lower with more DCs. Thus, one might conclude that keeping data at original locations is always the best choice. To challenge this conclusion we used the second dataset, i.e. WISDM. We anticipate that this analysis confirms (i), but not (ii). As anticipated before, this dataset is quite different from HAPT. It contains fewer activities recorded, each one is described by fewer features but, the number of initial locations is higher(i.e. 27) than in the HAPT dataset. This means that in each location there are fewer activities observed. These characteristics make this dataset more challenging than HAPT because it represents a sort of worst case scenario with very few observations for each initial location. Results for the WISDM dataset are reported in Table II. As in Table I the first and the last rows represent the fully distributed and the centralized solutions, respectively. In terms of accuracy, results confirm what we have observed in the HAPT dataset. There is no significant difference in the loss obtained with different number of DCs, and thus even a completely decentralised solution yield similar accuracy with respect to a completely centralised one. Moreover, comparison between HTL and no-HTL in Fig. 2b shows that the advantage of the former is rather constant independently on the number of DCs. However, the situation in terms of overhad - and therfore the optimal operating point - changes in the case of the WISDM dataset. The figures reported in the first raw of Table II are very interesting. In fact if we compare the amount of traffic generated by the exchange of models in the fully distributed configuration with the centralized solution we notice that the latter is better both in terms of accuracy and traffic overhead. The reason is that, the size of the models used by HTL combined with the number of models' exchanges is larger than the dataset itself. This confirms (as theoretically predicted in [7] that there exists a

978-1-5090-1131-5/16/$31.00 ©2016 IEEE



(a) HAPT (b) WISDM

Fig. 2. HTL vs no-HTL Loss comparison varying the number of data collectors

trade-off between the size of the models, the number of DCs and the amount of data for each DC that imposes a limit below which moving data is more efficient than moving models. In fact, if we look at the fourth row of Table II we observe that moving models is more efficient than moving data with that number of DCs. Thus, as the accuracy is basically independent on the number of DCs, we can conclude that in the case of WISDM the optimal operating point is neither a completely centralised nor a completely decentralised solution. This is not because of an *accuracy* argument, but because of a *network overhead* argument.

VI. Conclusions

In this paper we performed an analysis of the trade off between accuracy and network traffic for the distributed learning solution based on the Hypothesis Transfer Learning framework presented in [7]. We compared the amount of traffic generated when analysing the data collected by smart-devices on a variable number of Data Collectors located in the core network and we related this quantity to the accuracy obtained by a distributed learning scheme based on HTL applied to the DCs. We assume that the network traffic decreases when the number of DCs increases, such that the case with only one DC corresponds to using a conventional global cloud platform, and the case when the number of DCs equals the number of physical locations corresponds to a completely decentralised solution where data is kept local where it is generated. Our results show that the accuracy of our solution does not significantly vary with the number of DCs. This is quite interesting, as it shows that it is possible to use a completely decentralised solution where data analysis is done entirely on edge devices, according to a fog computing paradigm. Moreover, our results show that factors like the amount of data stored at each location, the number of locations and the size of the models must be carefully considered because they may affect the optimal operating point of our solutions in term of the number of DCs, i.e. the point where network overhead is minimised, as accuracy is basically independent on the number of DCs.

Acknowledgment

This work is partly funded by the EC under the H2020 REPLICATE (691735), SoBigData (654024) and AUTOWARE (723909) projects.

References

[1] Cisco. (2015) Cisco visual networking index: Global mobile data traffic forecast update, 2014–2019. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html

[2] J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, and A. Marrs. (McKinsey Global Institute, 2013) Disruptive technologies: Advances that will transform life, business, and the global economy.

[3] E. Borgia, "The internet of things vision: Key features, applications and open issues," *Computer Communications*, vol. 54, pp. 1 – 31, 2014.

[4] ETSI TC M2M. ETSI TS 102 690 v2.1.1 (2013-10) – Machine-to-Machine Communications (M2M); Functional Architecture, 2013.

[5] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *MCC*, New York, NY, USA: ACM, 2012, pp. 13–16.

[6] P. G. Lopez *et al.*, "Edge-centric computing: Vision and challenges," *ACM Sigcomm Computer Communication Review*, 2015.

[7] L. Valerio, A. Passarella, and M. Conti, "Hypothesis transfer learning for efficient data computing in smart cities environments," in *Proc of SMARTCOMP*, 2016.

[8] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140.

[9] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. of Computer and System Sciences*, vol. 55, no. 1, pp. 119 – 139, 1997.

[10] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth, "Using and combining predictors that specialize," in *STOC '97*. New York, NY, USA: ACM, 1997, pp. 334–343.

[11] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*,vol. 3,no. 1,pp. 79–87, 1991.

[12] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *J. Mach. Learn. Res.*, vol. 6, pp. 1579–1619, Dec. 2005.

[13] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le *et al.*, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1223–1231.

[14] A. Coates,*et al.*, "Deep learning with cots hpc systems," *JLMR*, vol. 28, no. 3, pp. 1337–1345, 2013.

[15] A. Navia-Vázquez and E. Parrado-Hernandez, "Distributed support vector machines," *Neural Networks, IEEE Trans on*, vol. 17, no. 4, pp. 1091–1097, 2006.

[16] L. Georgopoulos and M. Hasler, "Distributed machine learning in networks by consensus," *Neurocomputing*, vol. 124, pp. 2 – 12, 2014.

[17] S. Scardapane, D. Wang, M. Panella, and A. Uncini, "Distributed learning for random vector functional-link networks," *Information Sciences*, vol. 301, pp. 271 – 284, 2015.

[18] I. Kuzborskij1, F. Orabona, and B. Caputo, "Transfer learning through greedy subset selection," in *Proc of ICIAP*, 2015.

[19] J.-L. Reyes-Ortiz, L. Oneto, A. SamÃ, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomputing*, 2015.

[20] J. W. Lockhart, *et al.*"Design considerations for the wisdm smart phone-based sensor mining architecture,," in *Proc of SensorKDD*, 2011.

978-1-5090-1131-5/16/$31.00 ©2016 IEEE