An Innovative Neuro-Genetic Algorithm and Geometric Loss Function for Mobility Prediction

Stylianos Tsanakas*, Aroosa Hameed †, John Violos †, Aris Leivadeas †
* School of Electrical and Computer Engineering, National Technical University of Athens, Zografou, Greece
Email: el09727@mail.ntua.gr
† Department of Software and IT Engineering, École de technologie supérieure, Montreal, Canada
Email: aroosa.hameed.1@ens.etsmtl.ca, ioannis.violos.1@ens.etsmtl.ca, aris.leivadeas@etsmtl.ca

***Abstract*—In this research we design a time series geo-location prediction model based on Long Short-Term Memory (LSTM) with a custom geometric loss function. In order to estimate a close to optimal LSTM Recurrent Neural Network (RNN) architecture we use an innovative Genetic Algorithm (GA) tailored for RNN hypertuning. The proposed Neuro-Genetic Algorithm (Neuro-GA) includes a similarity function for the selection of the RNN that will be recombined and an early stopping criterion for the worse performing RNNs. In addition, we examine the applicability of an incremental learning approach for personalized RNN modeling. Compared with auto-machine learning and deep learning models, the proposed methodology shows substantially better prediction results and the early stopping criterion improves the speed of hypertuning convergence. The experiments also show that the incremental learning approach has significant better accuracy than a generic RNN as the personalized models are retrained to new users location data.**

***Index Terms*—Mobility, Deep Learning, Recurrent Neural Networks, Time Series, Genetic Algorithms, Incremental Learning**

I. Introduction

The human movement behaviour has no strong theoretical understanding and we cannot apply full numerical models. The decision of a moving entity to move straight, change direction or accelerate his speed cannot be modelled efficiently by kinematic equations [1]. However, analyses of historical mobility data have shown repeated behaviour patterns and strong autocorrelation values making a data-driven approach a feasible solution [2].

The methodology and the approach for mobility prediction modeling depends on the use case and the specification of the target values. The three main approaches are: (a) the distribution of the mobility entities in points of interests [3], (b) the full trajectory prediction [4], and (c) the one-step-ahead geo-location prediction [5]. In this research we focus on the one-step-ahead geo-location prediction following a time series forecasting technique that analyse a look-back window of previous geo-locations and predict the next-step geo-location. The geo-locations are the target values and represented by the latitude (lat) and longitude (lon).

The one-step-ahead geo-location prediction is a challenging task mainly due to the fact that the mobility direction and speed can change abruptly and that the environment that the mobile entities are moving are usually very dynamic (e.g. smart cities, transportation systems, etc.). This motivated us to focus our research in an adaptive data driven solution that leverage geo-location data, the time component of the chronologically data sequences, and the ability to learn from long-term dependencies. LSTMs are a popular option for time series forecasting, they are a special case of neural networks layers capable to capture long-term dependencies but they have the disadvantage of computational heavy training. This problem becomes more severe if the amount of data is large and and we use a hypertuning method to find a close to optimal LSTM-RNN architecture.

Genetic Algorithms (GA) is an evolutionary computation method capable of solving hyper-parameter optimization problems. In the context of RNN, GA are used for the estimation of a close to optimal LSTM-RNN architecture, through a population of candidate neural networks and a natural selection process. The generic GA has two important limitations in the case of RNN hypertuning. Firstly, the mating selection is not a straightforward process for heterogeneous topologies and secondly the evaluation of a population of neural networks through consecutive generations is a computational heavy method. In this paper, we propose the Neuro-Genetic Algorithm (Neuro-GA) which is a GA tailored in the peculiarities of RNN and tackles the aforementioned limitations.

The three major contributions of our research are:

- We propose a geometric loss function for the RNN training in mobility data.
- We significantly improve the Genetic Algorithm for Hypertuning RNN with early stopping criterion and a new similarity function for recombination.
- We apply an incremental learning approach for personalized mobility prediction and evaluate the results.

The rest of the paper is structured as follows: Section 2 highlights the related work in mobility prediction. Section 3 explains how the mobility prediction can be modelled with time series RNNs. Section 4 explains an automatic hypertuning methodology that leverage computational intelligence for smart and efficient neural network adaptation. Section 5 introduces a simple but efficient methodology for personalisation mobility training and prediction. Section 6 describes the experimental setup and the evaluation results of our proposed

methods. Finally, Section 7 concludes the paper and suggest future directions.

II. Related Work

Mobility prediction is used to represent the estimation of future location of mobile entities. In the literature, there exists several state of the art algorithms that predict mobility using mobile entities past movements. In [6], the authors proposed mobility prediction in Ad Hoc networks for the purpose of improving routing in a network. For this, they trained a multi-layer and RNN model that predicts the location of mobile entities using time series data of their locations. The neural network is a three-layered architecture and trained using back propagation through a time-series algorithm. In another study [7], the authors present a forecasting technique that predicts the pedestrian movements using the urban vehicular traffic traces collected in Australia. Firstly, they proposed a method to divide the pedestrian datasets in form of cell sequences using cell construction method. Then they applied the Deep Learning (DL) method called Recursive Neural Network with a Gated Recurrent Unit (GRU) to predict the next cell sequence which is the next location of a pedestrian using the available historic cell sequence. The advantage of this work is that it required limited computational effort for training.

Daksh et al. [8] also proposed a DL technique for human trajectory prediction. For this they used spatial matching network and modelled the spatial context of any subject of interest in a surrounding environment to get better trajectories prediction. They proposed a Spatially Static Context Network (SSCN) for modelling purposes and then they applied pooling mechanism and trained attention-based LSTM model. Next for the trajectory prediction, this work [9] provides a new method called Crowd Interaction Deep Neural Network (CIDNN) for displacement prediction. The method consists of the four different components as: firstly, motion encoder encodes trajectories using LSTM, secondly location encoder encodes the pedestrian's location and their influences, thirdly crowd interaction gives linear combinations of trajectory encoding and fourth displacement prediction gives the estimates of location displacement.

In the literature, there also exists a large group of studies for predicting the transport mode using the Geolife GPS trajectories datasets [2]. Sina et al. [10] proposed a Convolutional Neural Network (CNN) architecture based prediction mechanism for inferring transportation mode of user using GPS trajectories. The transportation modes include walking, biking, bus, driving and train. For the CNN the dataset is preprocessed and a suitable representation of data is designed where each data instance is composed of four features as acceleration, jerk, speed and bearing rate.

The authors in [11] proposed a supervised learning approach to predict the transportation mode of users using the Geolife GPS data. The approach constitutes three steps as: firstly, GPS trajectory data is divided into different segments of transportation modes using the segmentation method. Secondly, features are extracted from the dataset, and thirdly an inference model is applied to classify the transportation modes using the extracted features. Lastly, some graph-based post processing algorithm is applied to improve a prediction performance. In [12], authors proposed a transportation estimation model based on Deep Neural Network (DNN) architecture. Firstly, the raw trajectories are converted to images for input to the neural network. Then features are extracted using the Stacked Denoising Autoencoder (SDA) and lastly the transportation mode is predicted using DNN.

In our previous work, a visitor's prediction in the next timestep for large events using the current distribution of timestamps is proposed [3]. For this, a fog architecture collects data, features are analyzed and then prediction is done using classification algorithms and regression techniques such as Random Forest (RF), K-nearest neighbor (KNN), Support Vector Classifier (SVC), Naïve Bayes (NB)/Kernel Ridge (KR) and DL models. Next position prediction can be also done using the LSTM model [5]. The proposed work consists of three pipelines: the first is training, in which position data are preprocessed as distance and bearing, then using genetic algorithm hyperparameters they are optimized for the neural network and they are inserted to knowledge base; the second pipeline is transfer learning in which DL model is retrieved from the knowledge base using similarity based functions, transfer learning is performed and the final model is concluded; the third pipeline is inference in which prediction of next timestamp is performed.

In this paper, we also solve the problem of one-step-ahead geo-location prediction with a LSTM approach. However, this research is fundamentally different from previous methods because we are proposing custom functions along with the incremental learning approach. For instance, for the training of neural network i.e., LSTM for urban mobility data, we provided the custom geometric loss function. Furthermore, for hyper tuning a neural network we present a modified GA with new similarity function and stopping criterion. Lastly, we apply an incremental learning approach for personalized mobility prediction.

III. Mobility Modeling

In this paper, we examine one of the most interesting and challenging mobility use cases which is the mobility in smart cities. In this case, there are different types of transport means such as taxi, bus, subway and walking. The different transport means imply significantly different movement patterns and statistical data properties. By analyzing smart cities mobility data, we have observed that the average and the maximum speed, the frequency of changing direction, the acceleration, and the duration of remaining motionless in the same position significantly differs based on the transport mean. LSTM-RNNs satisfy the two important requirements of the mobility modeling and forecasting, first the ability to generalize from multiple different mobility data patterns and second to hold information from previous steps in a look-back window.

A. Trajectory Modeling with Time Series and Distance Bearing Transformation

The one-step-ahead geo-location prediction model can process trajectories formalized as sequences of data observations and apply a time series forecasting approach. Modern mobile and IoT devices have built-in GPS receivers that periodically record their geographical locations. A sequence of recorded geo-locations in constant time intervals for a mobile entity defines a trajectory. At the time-step t_p a trajectory is defined by the sequence of $(lat_{t_i}, lon_{t_i})\{t_i = 0..p\}$ where t_0 is the first position in the trajectory and t_p is the current position. Knowing a sub-sequence of the c previous positions $(lat_{t_i}, lon_{t_i})\{t_i = 0..c\}$ with $c < p$ we can forecast the next position $(lat_{t_i+1}, lon_{t_i+1})$ as we can see in Fig. 1.

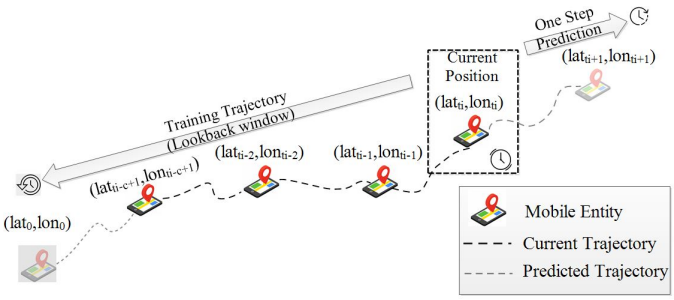

Fig. 1. One-step ahead Geo-location Forecasting

The LSTM-RNN follows a multi-variate, multi-output time series approach with a look-back window of size p taking as input a sequence of p lat and lon pairs and predicts the one-step-ahead lat and lon.Before the input of the LSTM-RNN, we apply a transformation of the lat and lon pairs into distance and bearing pairs. Lat and lon represent a point's distance from the equator and the prime meridian accordingly and they are measured in degrees and values between (- 90,90) and (-180,180). In distance/bearing system, the distance is used as the distance between the current and the previous geo-location and the bearing is the clockwise angular movement between the two positions.

Our previous research [5] has shown the advantage of using distance and bearing instead of plain coordinates. Predicting the next position, the network limits the search space in a way similar to how a human mind would perceive that there is a limited distance. This limited distance can be easily estimated by either the previous measurements or velocity and timespan until the next observation. The human is moving in a short time period to a limited distance, putting much focus on trying to estimate the next geo-location in a circle around the current geo-location.

B. RNNs for Geo-location Prediction

Recurrent Neural Networks (RNNs) model learn from series of observations arranged chronologically. RNN can capture temporal dependencies in sequential observations and perform tasks that require information from previous positions. As such, RNN has shown great performance in learning spatial and temporal patterns. In the problem formulation of the next geo-location prediction, the objective of the RNN is to predict the next-step distance and bearing with respect to the previous locations of an input sequence.

RNNs learn from sequential observations and use the hidden state that acts as the neural networks memory. The hidden state holds information on the previous observations the network has seen before. The hidden state is updated in every step with the combination of current input and the previous hidden state. The major limitation of the simple RNNs is that they only remember a few earlier steps in the sequences because of the gradient vanishing problem. This problem of RNN, that are capable to capture and remember long sequences of observations, is alleviated by the cell-state introduced in the Long Short-Term Memory (LSTM) recurrent networks.

The vanishing gradient problem is the phenomenon of small gradients updates during the training with the back-propagation and makes all the RNN weights almost do not change. LSTMs enhance the memorization through three gates that regulate the flow of past information. Through the gates forget, input, and output LSTMs regulate the cell state. The cell state is the memory of the network that keeps and transfers relative information all the way down the sequence chain. These gates learn which data in a sequence is important and pass the relevant information down the long chain of sequences to make predictions. The LSTM can effectively transfer information from the earlier time steps to later time steps, reducing the effects of short-term memory.

C. Geometric Loss Function

The mobility prediction has geo-spatial characteristics that we should use in the training of LSTM neural networks. The next geo-location in our approach is represented with the distance and bearing numerical values. Given the current lat and lon and the distance - bearing LSTM predictions, we can estimate the predicted next-step lat and lon. Thus, the LSTM is a multi-output multivariate regression model that learns by optimizing a loss function in distance-bearing data.

In order to adapt the training of the LSTM regression model to the geo-spatial problem we propose a custom Geometric Loss Function (GLF) instead of generic regression loss functions such Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE). For instance, the RMSE measures of how spread out the true data observations are than the predicted ones and cannot capture correctly the angular distances. In this case, if the predicted and the actual bearing are 0 and 359 degrees respectively the result would be a large loss value and the RMSE will not capture that these bearing values are very close. The GLF is given in Eq. 1 where d_p is the predicted distance, d_t is the true distance, b_p is the predicted bearing, and b_t is the true bearing.

$$GLF = \sqrt{(d_p cos(b_p) - d_t cos(b_t))^2 + (d_p din(b_p) - d_t sin(b_t))^2} \quad (1)$$

IV. HyperTuning

The RNN training process comprises of the parameters and the hyperparameters. The parameters are mostly the weights and the biases between the layers. The parameters are learnable by an optimizer which is based on a variation of the gradient descent iterative method. The hyperparameters are the major design decisions for a network topology and they belong in two categories the structural and the training. The structural hyperparameters defines the topology of the RNN, setting the number of layers, the number of neurons for each layer, the type of the activation functions and regularization values like the dropout rate. The training hyperparameters define how the learning process will take place in order to estimate the parameters specifying the optimizer and the learning rate.

Hypertuning is the automated process for hyperparameter selection of a DL model in order to minimize the loss of the objective function and maximize the accuracy of the predictions. Our goal is to take the human out of the learning method, not to rely on the experience of domain experts, and spend the minimum time and computational resources to build a neural network following a systematic and smart search methodology. Computational intelligence and population based algorithms are superior to grid search and random search methods because with a minimum number of trials achieve better accuracy, balancing an exploitation vs. exploration trade off. This means that instead of trying random hyperparameter combinations, computation intelligence methods locate where are the optimal until now solutions. They exploit the close to optimal areas and try sporadically unexplored areas of the hypothesis space in order to escape from local optima.

A. Neuro-Genetic Algorithm

A neural network is represented by a chromosome which consists of genes. The genes are the structural and the training hyperparameters. The genes can take a range of nominal or numerical values defining a hypothesis space. As example the gene of the number of dense layers is a numerical gene that can take values from one to a hypothetical maximum number of hidden layers. The gene of activation functions is a nominal variable that encodes the type of activation functions such as reLU, tanh, and sigmoid for a specific layer.

The first generation of the GA begins with a randomly initialized population of neural networks also named chromosomes in the evolutionary algorithm terminology. All the neural networks of the generation are evaluated using the geometric objective function described in the previous section. The top neural networks are selected from the population based on the objective function in order to ensure that only the best fit solutions survive into the next generation. The selected neural networks mate and create a set of offsprings in a process called crossover. The vanilla GA for hypertuning just recombines the selected neural networks and does not take into consideration the discrepancies between the neural networks. As example, a neural network with two hidden layers should not be recombined with a neural network with

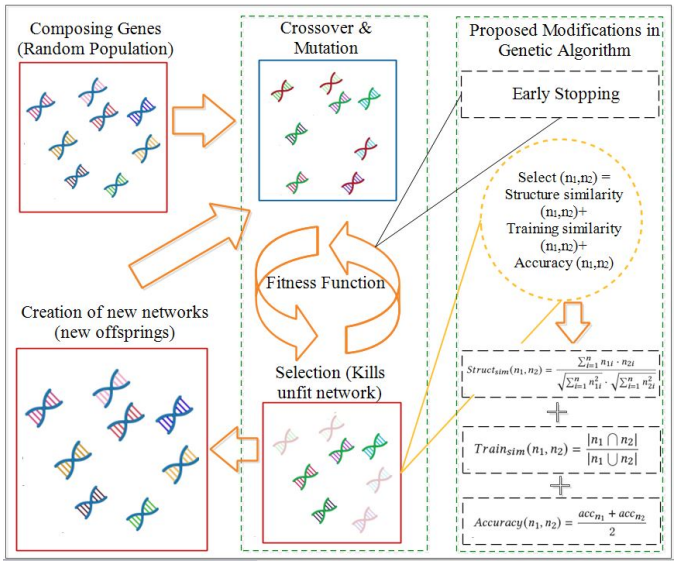

Fig. 2. Modified Genetic Algorithm

fifteen layers. The optimal activation functions from a two-layer neural network may not be the optimal solution in a fifteen-layer neural network. Different architectural components contribute in a different way based on the context and the topology they belong. This is the reason that we introduce a similarity function to select the neural networks that match to be combined.

After the crossover, the offsprings are mutated in order to create a new set of chromosomes that will be inserted into the population. We designed our GA to introduce in every generation new chromosomes based on crossover and mutation but we always keep the best chromosomes from the previous generations. Thus, we can guarantee that if we find the best fit solution early in the first generations, this solution will survive until the end of the evolution and the optimization process will converge. Once enough chromosomes have been inserted to replace the worst fit chromosomes of the population, a generation is said to have passed.

In every new generation the cycle of the evaluation, selection, crossover, mutation, and insertion starts again. The evaluation of the chromosomes is the most time consuming and computational heavy process in every generation. The training of a neural networks includes the iterative update of thousand RNN parameters for many epochs. Because the generations include a population of neural networks the GA should narrow down the search space applying a criterion to halt the training of chromosomes or continue it. This motivated us to introduce an early stopping criterion in the training of the candidate neural networks. The integration of the early stopping criterion and the selection function in the GA constitutes the Neuro-GA which is illustrated in Fig.2 and they will be described in the next two subsections. After a number of generations have elapsed, the hypertuning process converges, stops and outputs the most close to optimal chromosome. This chromosome also represents the most close

to optimal neural network based on the given mobility training data set.

B. Early Stopping Criterion

Our proposed genetic algorithm includes an early stopping criterion during the training of the neural networks. Doing so we incorporate the advantage of the hyperband hypertuning approach into every generation of the genetic algorithm. The hyperband approach makes a tradeoff between n random trials and b available computation resources. The core idea is that the training begins with n random neural networks, periodically evaluate their performance, discard the worse performing and allocate the free computational resources to the best performing.

Our hypertuning genetic algorithm begins in every generation with n neural networks binding b computational resources. After p epochs we compare the accuracy of each neural network, the neural networks of the current generation that have better average performance than the neural networks of the previous generation in the epoch p will continue the training, while the rest will early stop. In this methodology, we should mention that in the first generation we don't apply the early stopping criterion and in the last generation the final solution is trained for more than $2p$ epochs, until the validation error significantly surpasses the training error and the overfitting begins.

C. Selection Function for Recombination

The crossover process is the most complicated process in the genetic algorithm. In the crossover process the genes of the two parents should be recombined and one new offspring should be introduced in the population. In the generic design of the genetic algorithms there is no restrictions in the recombination of the best fit chromosomes, but the particularities of the neural networks impose constraints on the hypertuning.

The genes of neural networks during the evolution process are the structural and the training hyperparameters encoded with vector representations. A similarity function can quantify the similarity between two neural networks. A high similarity value denotes that the chromosomes have a similar structure in their topology, they learn efficiently with the same techniques, thus they have a good match to be recombined together. We propose the cosine similarity for the numerical hyperparameters and the Jaccard similarity for the nominal hyperparameters. We selected these two because they are monotonic on a range of values $[0, 1]$ and their perfect match corresponds to the value one. We also take into consideration the accuracy of the two candidate neural networks. Eq. 2 defines our proposed selection metric function two neural networks in a generation. The couples that have the highest selection value will be recombined and their offsprings will be inserted into the population of the next generation.

$$\begin{aligned} Select(n_1, n_2) = a \cdot Struct_{sim}(n_1, n_2) + \\ b \cdot Train_{sim}(n_1, n_2) + c \cdot Accuracy(n_1, n_2) \end{aligned} \tag{2}$$

The selection metric is the sum of three equations, the structure similarity, the training similarity and the accuracy. These three parts contribute with different weights in the selection metric with the parameters a, b and c to be in the range of $[0, 1]$ and their sum to one. In our experiments we have tested different values of these parameters and we conclude that a should be greater than c and c greater than b.

$$Struct_{sim}(n_1, n_2) = \frac{\sum_{i=1}^{n} n_{1i} \cdot n_{2i}}{\sqrt{\sum_{i=1}^{n} n_{1i}^2} \cdot \sqrt{\sum_{i=1}^{n} n_{2i}^2}} \tag{3}$$

The structure similarity Eq. 3 is based on the cosine similarity where n_1 and n_2 are the two candidate parents, n_{1i} and n_{2i} are the i-th structural hyperparameters. The cosine similarity is one of the most used functions to estimate the similarity between numerical features in the domain of data mining.

$$Train_{sim}(n_1, n_2) = \frac{|n_1 \bigcap n_2|}{|n_1 \bigcup n_2|} \tag{4}$$

The training similarity Eq. 4 is based on the Jaccard index and divides the intersection of their common nominal hyperparameters to the union. The Jaccard coefficient similarity is also widely used in data mining for the comparison of nominal data. The accuracy of the neural networks is the vanilla selection criterion in the genetic hypertuning. Our method should also favour the best fit neural networks to be recombined. Thus, in the Eq. 5 we define that the mean accuracy of the two candidate parents also contributes to selection process.

$$Accuracy(n_1, n_2) = \frac{acc_{n_1} + acc_{n_2}}{2} \tag{5}$$

The selection metric based on the ANN genes improves the exploitation of the genetic algorithm recombining the most similar and accurate chromosomes. The exploitation, in a smart search hypertuning process, declares that the evolutionary decisions are taken based on the best current information. The evolution should not only be guided by the exploitation of the most optimal solutions. It also needs a stochastic component in order to avoid the local optima. In order to explore very different areas of genes after the crossover the neural networks are mutated with a Gaussian distribution in the acceptable range of the hyperparamater values. Doing so we can escape from local minima and explore new solutions of the hypothesis space.

V. Generic vs. Personalized Prediction

In the methodology that we have described in the previous sections, we input a bunch of thousands trajectories of different mobility behaviours into the hypertuning algorithm and it outputs the most close to optimal generic mobility model. Then, the model provides predictions but it is never refit and extends its knowledge based on the new incoming data. From the other hand, personalised modelling and prediction concerns methodologies for an incremental adaptation of the data-driven

TABLE I
EVALUATION AND COMPARISON OF THE LSTM NEURO-GENETIC ALGORITHM FOR MOBILITY PREDICTION

method	unlabeled	walk	bus	bike	car	taxi	subway	train time
AutoSKLearn	58,44	21,81	83,65	54,75	97,77	132,85	107,47	3779
XGBoost	48,20	21,11	76,43	51,05	77,56	116,00	89,60	10463
GA LSTM	50,95	21,86	70,64	50,98	81,72	108,79	79,69	8310
Neuro-GA LSTM	30,53	14,85	51,28	24,52	43,88	44,23	52,98	4729

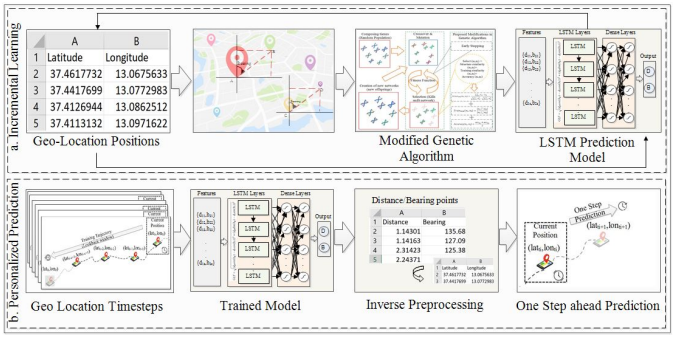

Fig. 3. Personalised One Step ahead Prediction Model

models based on individually users mobility behaviours and their new generated data. In a personalized modeling approach, we can start with a generic model and periodically re-fit and extend the models with new data observations as it is illustrated in Fig 3.

This approach is inline with the modern mobile and IoT devices with built-in GPS receivers that produce continuously users geographical locations. The same devices can also run the DL model that predict the next-step user geo-locations. This means that in every time-step new tuples $(lat_{pred}, lon_{pred}, lan_{true}, lon_{true})$ with predicted and true values are generated. These new on-device data can be used to locally fine-tune the generic models. Doing so, we can improve the accuracy of mobility prediction models for every individual user based on his own mobility behaviours.

The mobile devices can run a light-weight retrain of the generic mobility model with the on-device geo-location data. The retrain should be light-weight because the users do not want to spend their valuable mobile resources and battery to extensive neural networks training and also the transfer of the training on the Cloud would imply bandwidth consumption and privacy issues. The retrain process includes local fine-tune on the new batches of the sequences recorded by the device with an adaptive learning rate that begins intensively and gradually decays.

VI. EXPERIMENTAL EVALUATION

The proposed methodology has been implemented and experimentally evaluated in the Python 3 programming language using the libraries NumPy, pandas, Scikit-learn, SciPy, GeoPy, Scikit-Optimize, TensorFlow 2 and its higher-level API Keras. The environment we used for the experiments is a Jupyter notebook of the Google Colaboratory. The experiments' source code is available for any kind of reproduction and reexamination in the first author's GitHub repository [13]. The experiments took place with a real GPS trajectory dataset publicly available by the Geolife project of Microsoft Research Asia [2]. This trajectory dataset is appropriate for tasks like mobility pattern mining, user activity recognition and location recommendation and its 1.2GB size is sufficient an extensive research analysis.

The GPS trajectory dataset contains 17,621 trajectories from 182 users with a total distance of about 1.2 million kilometers. The dataset is constructed in a period of five years with a total duration of 48,000+ hours. The GPS trajectories are represented by sequences of time-stamped geo-locations, each of which contains the information of latitude and longitude. Most of the trajectories are logged every one to five seconds. In the data preparation stage, we filtered the trajectories that have not sufficient number of observations and the time-steps have been aligned at a constant interval of 5 seconds. The geo-locations are recorded by different GPS loggers and GPS-phones of moving people. The people perform everyday activities like go home, to work, entertainments and sports activities, such as shopping, sightseeing, dining, hiking, and cycling using different transport means.

A. Evaluation Methodology and Results

We split the dataset in nine separate groups of trajectories. The first group is for the training of the prediction model and the other eight trajectory groups are for the evaluation of the prediction models. These eight evaluation groups are labeled by the corresponding transport means: (a) walk, (b) bus, (c) bike, (d) car , (e) taxi, (f) subway and there is also one (g) unlabeled group that includes a mixture of trajectories by all the previous transport means. We experimentally compared our proposed model Neuro-GA, with our previous mobility prediction model GA [3] and two popular auto-ml models the AutoSKLearn and the XGBoost. The AutoSKLearn [14] is based on an automated machine learning workflow including the steps of pre-processing, regression, and hyper-parameter tuning using a Bayesian optimization process. It also contains a repository of previous optimization runs enabling training from previous saved settings. The XGBoost [15] is a software library that implements the ensemble learning approach of gradient boosted decision trees. XGBoost has gained much popularity because it has won several data mining and knowledge discovery competitions among them the prestigious Kaggle and KDD Cup.

Before the fitting of the models we applied the distance-bearing transformation described in the subsection 3.1. In our experiments the time-step is 5 seconds, the look-back window is a sequence of 15 steps and we predicted the next geo-location after 25 seconds. We selected a time horizon of 25 seconds because it has many practical applications in transportation and edge computing use cases. In addition, the largest amount of geo-locations in the dataset were recorded every 5 seconds. As future work researchers can try different values of time steps and size in the look-back window.

The training and the evaluation parts in our experiments have an equal number of trajectories and each trajectory has been used in its entirety. The evaluation parts also divided in the above mentioned eight trajectory groups in order to evaluate the accuracy of the models in each separate transport mean and also in the mixed unlabeled group. This is important because different transportation means have significantly different statistical properties. In the initial exploratory data analysis, we have seen that the average and max speed, the frequency of changing direction and the duration of being in the same position are significantly different based on the transport mean. This is a reasonable outcome but it raises the important question in the predictive data analysis, if the same model can be the most accurate in all the trajectory groups.

The training of the model took place using trajectories from all the transport means. Thus, we built a generic multivariate LSTM-RNN model using the Neuro-GA with the similarity function and the early stopping criterion described in section 4. In the evaluation stage, we saw the predictions of the LSTM-RNN model to have different characteristics and average errors based on the type of the transport mean of the trajectories. This signifies that the LSTM-RNN model takes as input the look-back window of the testing trajectory and regulates its predictions respectively. When we made the analysis on the predictions we saw that the maximum distance of the predictions between the current and the predicted geo-location is inside the limits of its transportation mean type. The same also happens for the speed and the changing rate in the bearing. As an example, in a testing walking trajectory the model never predicts distances bigger than the distance can cover a moving human in 25 seconds but, testing bus trajectories the predicted distances are analogous to the bus moving patterns.

Our conclusion is that even if the mobility model is generic, it can distinguish and provide predictions for each transport mean without the explicit input of their type. The model can adapt its predictions based on the data observations of the look-back window. The results of the experiments are summarized in the Table I. The columns with the transportation means indicate the average errors in meters for each step prediction in the next 25 seconds. The results show that in all transport means the proposed model has significant better results than the simple GA methodology and the two auto machine learning models. Comparing the simple GA with the Neuro-GA model we can see that the latter has significant improvements in the accuracy and the training time. The training time decreased from 8310 seconds to 4729 seconds making an improvement of 43% and the error decreased in a range from 27,4% to 52,98% in the different transport means. The redesigned genetic hypertuning algorithm is more efficient to manipulate the characteristics of the RNN architecture, learning behaviour and accuracy. In these significant improvements also contributed the geometric loss function that captures the peculiarities of the mobility data instead of a generic loss function like RMSE. In addition we can see in the last column of the Table I the computational time of training. The computation time of the inference is not mentioned as it has an order of magnitude of a few milliseconds in all cases.

B. Evaluation of the Neuro-Genetic Algorithm

Regarding the improvement of the early stopping criterion we see in the Table. II the completion time of each generation and the evolution in the accuracy between two consecutive generations. The accuracy in each generation is represented by the average value of the top half chromosomes of the population. The reason is that in each generation there is a percentage of chromosomes that have strong mutation in order to insert a stochastic component in the offsprings. These chromosomes are important because they make an exploration in new areas of the hypothesis space in order to escape from the local minima. The significant stochastic component may provoke these chromosomes to end up to low accuracy solutions. But, these chromosomes can be early detected with an early stopping criterion and stop their training. Thus, in the rows of Accuracy in the Table. II, we take the average accuracy only of the top half chromosomes of the population and we don't take into consideration the chromosomes that will not survive and be recombined.

In our experimental setup we used a population of twelve DL models for ten generations. We made multiple runs with different initialization in the first population of the chromosomes and in all cases we see that the accuracy of the fittest solutions converges and does not have significant improvement after the tenth generation. From the Table. II we see that the early stopping criterion decreased the training time in all the generations. The training time of each generation is defined by the time duration it needs all the population. Giving the computation resources from the worst fit solutions to the best fit solutions we saw improvements in both the accuracy and the training time. We also clarify that the accuracy improvements declare the percentage of the improvement between the current generation and the previous one. So, we don't have values in the column of the 1st generation. The second clarification we must give is that the training and the hypertuning process is stochastic. This is the reason that we didn't see improvement in the 9th generation but we had improvements in the 10th generation. In the 9th generation not all the recombinations and the mutations of the hyperparameters provided more accurate solutions, but, the stochastic component is the cause of slightly more accurate solutions in the 10th generation.

TABLE II
IMPROVEMENTS IN ACCURACY AND TRAINING TIME WITH THE NEURO-GENETIC ALGORITHM

	1st gen	2nd gen	3rd gen	4th gen	5th gen	6th gen	7th gen	8th gen	9th gen	10th gen
GA $Train_{time}$	1956	615	1470	700	598	552	564	528	776	551
GA Accuracy	-	0,1816	0,0487	0,0436	0,0123	0,0021	0,0006	0,0029	0,0019	0,0018
Neuro-GA $Train_{time}$	1164	532	413	311	361	465	292	477	367	347
Neuro-GA Accuracy	-	0,2862	0,1812	0,0914	0,0038	0,1224	0,1188	0,0026	0,0000	0,0511

C. Testing a Simple Personalized Modelling

Personalized modeling and prediction with a lightweight training on device can increase the accuracy of the predictions and ensure user data privacy. Mobile devices record geo-location sequences forming training mini-batches that can be used for a periodical fine-tune of the RNN model. The mini-batches include data observations with users mobility patterns that can be used for the adaptation of a generic mobility model to the specific users movement behaviours.

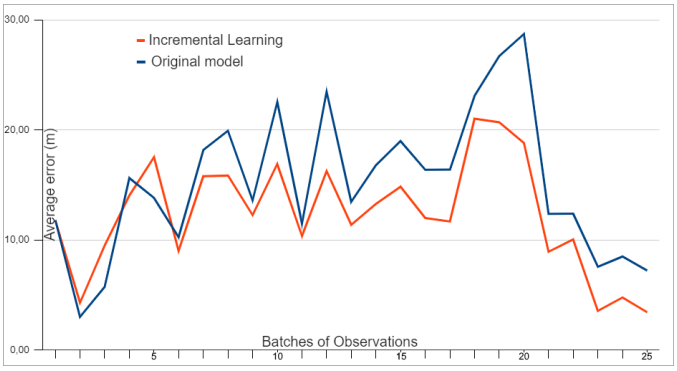

Fig. 4. Incremental Learning on Trajectory Data

In our experiments, we begun with a generic LSTM-RNN model hyper-tuned and trained as described in the previous sections. Then, we traversed different trajectories making a re-train and evaluation in consecutive mini-batches. This approach is based on the incremental learning methodology that continuously extends a model with a continuous retraining in the new incoming data. In the comparison of the original generic model with the incremental learning approach, we found out that in the first mini-batches the model accuracy is not improved by the continuous learning but, as the training batches increases the average error of the incremental learning becomes consistently lower than the original model as we can see in Fig. 4.

VII. CONCLUSIONS

The Neuro-Genetic algorithm combines the benefits of evolutionary algorithms with the neural networks in order to conclude to a close to optimal DL topology. Specifically this study improves the LSTM-RNN approach for one-step-ahead geo-location prediction. A geometric loss function was used instead of common loss functions and a transformation of the (lat, lon) sequences into distance-bearing was also applied in order to leverage the particularities of spatio-temporal data. To estimate the most close to optimal RNN architecture, we proposed an improved genetic algorithm including an early stopping criterion which determines whether to halt the training or continue and a similarity function for optimal RNN recombination. The experimental evaluation showed that the proposed methodology surpasses in terms of accuracy previous mobility prediction models and improves significantly the training time.

In addition, we tried a personalised mobility modelling with an incremental learning and fine-tuning approach. This is the topic that our future research is focused. Incremental learning can be further improved with more efficient techniques than periodic re-fit. Specifically, we plan to investigate a drift mechanism that can understand if a user changes a transportation mean and to apply different mobility models accordingly. Lastly, we are interested in the design of an incremental learning approach that addresses the catastrophic forgetting problem preventing the new knowledge to erase permanently what the model learnt in the past.

ACKNOWLEDGMENT

This work was supported in part by the CHIST-ERA-2018-DRUID-NET project "Edge Computing Resource Allocation for Dynamic Networks".

REFERENCES

[1] A. Valsamis, K. Tserpes, D. Zissis, D. Anagnostopoulos, and T. Varvarigou, "Employing traditional machine learning algorithms for big data streams analysis: The case of object trajectory prediction," *Journal of Systems and Software*, vol. 127, pp. 249–257, May 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016412121630084X

[2] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proceedings of the 18th international conference on World wide web*, ser. WWW '09. New York, NY, USA: Association for Computing Machinery, Apr. 2009, pp. 791–800. [Online]. Available: https://doi.org/10.1145/1526709.1526816

[3] J. Violos, S. Pelekis, A. Berdelis, S. Tsanakas, K. Tserpes, and T. Varvarigou, "Predicting Visitor Distribution for Large Events in Smart Cities," in *2019 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Feb. 2019, pp. 1–8, iSSN: 2375-9356.

[4] I. Varlamis, C. Sardianos, V. Bogorny, L. O. Alvares, J. T. Carvalho, C. Renso, R. Perego, and J. Violos, "A novel similarity measure for multiple aspect trajectory clustering," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, ser. SAC '21. New York, NY, USA: Association for Computing Machinery, Mar. 2021, pp. 551–558. [Online]. Available: https://doi.org/10.1145/3412841.3441935

[5] J. Violos, S. Tsanakas, M. Androutsopoulou, G. Palaiokrassas, and T. Varvarigou, "Next position prediction using lstm neural networks," in *11th Hellenic Conference on Artificial Intelligence*, ser. SETN 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 232–240. [Online]. Available: https://doi.org/10.1145/3411408.3411426

[6] H. Kaaniche and F. Kamoun, "Mobility prediction in wireless ad hoc networks using neural networks," 2010.

[7] S. Choi, H. Yeo, and J. Kim, "Network-wide vehicle trajectory prediction in urban traffic networks using deep learning," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2672, no. 45, pp. 173–184, 2018.

[8] D. Varshneya and G. Srinivasaraghavan, "Human trajectory prediction using spatially aware deep attention models," 2017.

[9] Y. Xu, Z. Piao, and S. Gao, "Encoding crowd interaction with deep neural network for pedestrian trajectory prediction," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 5275–5284.

[10] S. Dabiri and K. Heaslip, "Inferring transportation modes from gps trajectories using a convolutional neural network," *Transportation Research Part C: Emerging Technologies*, vol. 86, p. 360–371, Jan 2018. [Online]. Available: http://dx.doi.org/10.1016/j.trc.2017.11.021

[11] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on gps data," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, ser. UbiComp '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 312–321. [Online]. Available: https://doi.org/10.1145/1409635.1409677

[12] Y. Endo, H. Toda, K. Nishida, and A. Kawanobe, "Deep feature extraction from trajectories for transportation mode estimation," in *Advances in Knowledge Discovery and Data Mining*, J. Bailey, L. Khan, T. Washio, G. Dobbie, J. Z. Huang, and R. Wang, Eds. Cham: Springer International Publishing, 2016, pp. 54–66.

[13] STsanakas, "STsanakas/Neuro-Genetic_algorithm_for_mobility_prediction," Jul. 2021. [Online]. Available: https://github.com/STsanakas/Neuro-Genetic_Algorithm_for_Mobility_Prediction

[14] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Auto-sklearn: Efficient and Robust Automated Machine Learning," in *Automated Machine Learning: Methods, Systems, Challenges*, ser. The Springer Series on Challenges in Machine Learning, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds. Cham: Springer International Publishing, 2019, pp. 113–134. [Online]. Available: https://doi.org/10.1007/978-3-030-05318-5_6

[15] D. Nielsen, "Tree Boosting With XGBoost - Why Does XGBoost Win "Every" Machine Learning Competition?" 2016, accepted: 2017-03-13T07:58:50Z Publisher: NTNU. [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2433761