Random Vector Functional Link Networks for Road Traffic Forecasting: Performance Comparison and Stability Analysis

Eric L. Manibardo*, Ibai Laña*, and Javier Del Ser*†
*TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Bizkaia, Spain
†University of the Basque Country (UPV/EHU), 48013 Bilbao, Bizkaia, Spain
Email: {eric.lopez, ibai.lana, javier.delser}@tecnalia.com

2021 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-3900-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/IJCNN52387.2021.9533576

***Abstract*—Nowadays, Machine Learning algorithms enjoy a great momentum in multiple engineering and scientific fields. In the context of road traffic forecasting, the number of contributions resorting to these modeling techniques is increasing steadily over the last decade, in particular those based on deep neural networks. In parallel, randomization based neural networks have progressively garnered the interest of the community due to their learning efficiency and competitive predictive performance. Although these two properties are often sought for practical traffic forecasting solutions, randomization based neural networks have so far been scarcely investigated for this domain. In particular, the instability of these models due to the randomization of part of their parameters is often a deciding factor for discarding them in favor of other modeling choices. This research work sheds light on this matter by elaborating on the suitability of Random Vector Functional Link (RVFL) for road traffic forecasting. On one hand, multiple RVFL variants (single-layer RVFL, deep RVFL and ensemble deep RVFL) are compared to other Machine Learning algorithms over an extensive experimental setup, which comprises traffic data collected at diverse geographical locations that differ in the context and nature of the collected traffic measurements. On the other hand, the stability of RVFL models is analyzed towards providing insights about the compromise between model complexity and performance. The results obtained by the distinct RVFL approaches are found to be similar than those elicited by other data driven methods, yet requiring a much lower number of trainable parameters and thereby, drastically shorter training times and computational effort.**

***Index Terms*—Randomization based neural networks, random vector functional link networks, road traffic forecasting.**

I. Introduction

The capabilities of Machine Learning techniques to learn from data have pushed them towards their success for many Artificial Intelligence modeling applications [1]. Among them, road traffic forecasting has also joined this massive adoption [2], enabling the upsurge of intelligent transportation management systems for metropolitan and inter-urban areas. A reliable short-term prediction of traffic state measurements (e.g. flow per time interval or average speed) supports the operational decision making process when dealing with traffic congestion in the form of tracing alternative routes or sending messages to traffic information displays [3].

Since Machine Learning merges several categories of data driven algorithms, distinct learning methods have been applied to problems related to road traffic forecasting. Addressed with a variety of approaches, road traffic forecasting methods have evolved over the years from shallow learning algorithms, to modern and complex Deep learning architectures [2]. These latter models currently lead the performance benchmark of the majority of recent publications about this topic, as revealed in recent surveys about this topic [4], [5]. Inspired by the structure of human brains, Deep learning is fundamentally composed by neural units, which allow the model to theoretically and asymptotically approximate any non-linear function. Moreover, Deep learning architectures can autonomously produce high-level features from raw data, so that increasingly informative representations of the raw data are learned, eventually yielding an enhanced predictive performance of the model. Deep learning also benefits from data fusion, wherein information from distinct data sources can be fused together and fed to the model (for instance, the queries formulated by the users of a navigation service can be employed as congestion predictors [6]). This is especially interesting when dealing with non-Euclidean data like graph representations [7].

While these arguments are commonly exposed as the main reason for selecting Deep learning solutions against other Machine Learning models, they are not straightforward in the context of road traffic data. Data fusion enhances the performance of network-wide forecasting models, as spatio-temporal relationships between nodes of a traffic network can be represented precisely with a graph. However, for models that predict a single point or multiple unrelated nodes of a traffic network, this fusion capability can not be exploited. Additional sources of knowledge from contextual data such as weather [8], calendar [9], or air pollution [10] can be adapted as regular input data, so other Machine Learning models can benefit from them as well. When dealing with self-descriptive data like traffic measurements, automatic data representation learning methods do not offer any major advantage either. Traffic congestion can be detected by high levels of traffic flow or by average speed drops relative to the nominal speed. In other fields such as medical image classification, domain expertise play a huge role, so Deep learning techniques have demonstrated remarkable results due to their ability to autonomously extract high-level features [11]. But again, road traffic data is easily interpretable.

Nevertheless, there are case studies that have access to

traffic data from multiple locations of a transport network. If so, Deep learning architectures could discover high-quality features that would combine multiple traffic measurements into more representative information. But for applications addressed as an auto-regressive time series forecasting task (i.e. input data is conformed by past values of the time series to predict), learning new features from data can be shown not to improve the model performance [4].

Lastly, to approximate complex natural non-linear functions, model complexity (number of internal parameters) has to be necessarily increased. This double-edged capability sometimes yields overly complex Deep learning architectures for traffic forecasting, which can be alternatively modeled by shallow and less computationally expensive algorithms.

Inspired by the above arguments, randomization based neural networks seem to be an option worth to be explored for modeling road traffic data. These models differ from conventional neural networks in the way their parameters are tuned [12]. While Deep learning models adjust their internal parameters by backpropagating loss gradients, randomization based neural networks keep some parameters of their architecture fixed after a random initialization of their values. Only some weights are adjusted (those concerning the output layer), so that the training time is dramatically reduced, ultimately allowing developers to deploy these models on computationally constrained/inexpensive machines. Additionally, randomization based neural networks tend to be less complex as per the depth and number of parameters per layer, so together with their random initialization, their automatic feature extraction capability is not as powerful as that of avant-garde Deep learning architectures. However, as previously commented, the self-descriptive nature of traffic data makes this characteristic of little interest for this scenario. In turn, non-euclidean data is mostly exploited for network-wide predictions, so for a classical road traffic forecasting approach where the target task is to predict the next value of a time series at a single location (i.e. flow measurements per time-step), the data fusion capability of Deep learning models are beneath contempt.

In this context, Random Vector Functional Link (RVFL) networks and their multiple variants [13]–[15] can be interesting alternative models due to their good performance attained in time series forecasting tasks [16], [17]. To the best of the authors' knowledge, there are no previously published work that explores the effectiveness of these models in the context of road traffic forecasting. As transport networks grow both in size and complexity, traffic ma nagement would benefit from fast learning yet well performing RVFL-based forecasting models. Given this rationale and the challenge it represents, the contributions of the work presented in this manuscript can be summarized as follows:

- We measure the performance of RVFL-based methods when dealing with distinct context and types of road traffic forecasting problems. Furthermore, we compare their performance to that elicited by standard backpropagation-based multilayer neural networks and ensemble learning algorithms.
- We analyze the statistical stability of randomization based neural networks by measuring the dispersion in the results of several test runs resulting from the epistemic uncertainty induced by their random initialization. With this second study we aim to shed light on whether the reason for discarding these models in favor of others that convey greater reliability can be grounded on empirical evidence.
- We provide insights on the capabilities of RVFL models in the context of road traffic forecasting, providing an informed reflection on the utility of these computationally inexpensive methods for some implementation scenarios.

The rest of the paper is organized as follows: Section II introduces the traffic data collected for this investigation and also resumes the particularities of each of the RVFL based neural networks covered in this study. The experimental setup is defined at Section III. Two case studies are conducted towards analyzing both the performance of several state of the art learning methods and the stability of randomization based neural networks, in the context of traffic forecasting. Section IV gathers the obtained results and discuss the potential applications of RVFL architectures for the problem at hand. Finally, Section V summarizes the insights distilled from this work and examine future research lines involving randomization based learning methods.

II. Materials and Methods

As has been previously stated, there is no prior work that explores RVFL-based models for road traffic forecasting. Therefore, we first describe the collection of selected traffic data sources that gathers different traffic context, towards analyzing the performance of these randomization based neural networks when predicting distinct traffic profiles.

Thereafter, a short description of the considered learning methods is provided. For the sake of completeness, we also delve into the structure the RVFL variants included in the benchmark, which will help the reader understand conclusions later drawn from the experimental results.

A. Selecting traffic data sources

As stated in [4], the nature of collected traffic measurements largely defines the relationships between the traffic time series and congestion states. Although there are other traffic measurements such as travel time or occupancy, the most commonly used data sources contain *flow* and/or *speed* measurements. Even if collected at the same road, traffic measurements expressed as a time series describe different patterns according to the daily timeline. Typically, a traffic flow time series collected in urban environments exhibits two spikes, corresponding to the rush hours back and forth to working places. Events, weather, calendar, and other factors modify this basic profile by narrowing or expanding such flow spikes in time, or even removing them. In contrast, the average speed of a road remains in general stable and close to the nominal speed when the occupancy of the road remains below its maximum capacity (i.e. the so-called *free flow* status of the road). Disruptions in the speed time series come in the form

of valleys. Traffic congestion results in speed drops, directly related to the spikes of the flow time series as can be seen in the fundamental diagram of traffic flow [18].

Analogously, traffic behavior also varies between highways and city roads. Freeways and other high nominal speed inter-urban roads provide stable patterns that barely changes between close locations. Since they act as the link between major cities in a regional transportation network, the traffic behavior is scarcely influenced by contextual factors, as opposed to the fluctuations that might appear in city road traffic. As a result, urban networks pose more challenging forecasting problems due to the complex interplay between street crossings, traffic lights, urban transit and goods transportation.

Other factors can also slightly impact the behavior of the analyzed traffic recordings (e.g. the type of sensor used for data collection, or the traffic sampling rate). Nevertheless, in order to provide a solid comparison benchmark between forecasting models, only the previous points will be taken into account. Table I shows the considered data sources for this study, covering different scope and traffic measurements.

TABLE I
SELECTED DATA SOURCES AND THEIR CHARACTERISTICS

Location	Traffic measurement	Scope	Time resolution	Year
Madrid [19]	Flow	Urban	15 min	2018
California [20]	Flow	Freeway	5 min	2017
New York [21]	Speed	Urban	5 min	2016
Seattle [22]	Speed	Freeway	5 min	2015

B. RVFL variants and considered learning methods

Considering that randomization based neural networks are developed under the key idea of holding the initialization weights of specific components of the architecture, there is a large number of models that could be compared for traffic forecasting problems [12]. However, this study focus on RVFL architectures, motivated by the hypothesis that they can outperform more complex methods due to the self-descriptive nature of traffic measurements. The direct connection between the input and output layer featured by RVFL based models allow them to access both hidden features and original values of traffic recordings.

Inheriting the structure of a multilayer perceptron (MLP), the general architecture of RVFL is composed of the standard building blocks of a neural network: the neural unit [23]. Thus, the main difference between MLP and RVFL variants lies in the learning algorithm adjusting their internal parameters. The first optimizes all internal parameters (i.e. output/hidden weights and biases) via gradient backpropagation, whereas the latter only adjusts the parameters of the output layer. These output weights $\mathbf{w}_o$ map a vectorized version of the information processed on the input through the first part of the model to the target variable to be predicted. Mathematically, assuming row vector notation and that the output is a single real-valued variable $y \in \mathbb{R}$ (regression):

$$\widehat{y}_t = \mathbf{s}_t \cdot \mathbf{w}_o^\top = [\mathbf{x}_t, \mathbf{h}_t] \cdot \mathbf{w}_o^\top, \tag{1}$$

where t is an index that denotes the number of the data instance (in the context of time series forecasting, the time at which a prediction is issued), $[\cdot,\cdot]$ stands for vector concatenation, $\widehat{y}_t$ is the predicted value for the target, and $\mathbf{s}_t$ is a vector concatenating the input $\mathbf{x}_t$ and the hidden features $\mathbf{h}_t$. In the general form of a multilayered RVFL model (see Figure 1), the hidden features $\mathbf{h}_t^\ell$ computed for $\mathbf{x}_t$ at layer $\ell \in \{1, \ldots, L\}$ are given by:

$$\mathbf{h}_t^\ell = \mathbf{W}^\ell \cdot \left(\mathbf{h}_t^{\ell-1}\right)^\top, \tag{2}$$

where $\mathbf{h}_t^0 = \mathbf{x}_t$, and $\mathbf{W}^\ell$ is an intermediate $N^\ell \times N^{\ell-1}$ weight matrix, with N^ℓ denoting the number of neurons of each layer. Clearly, $N^0 = |\mathbf{x}_t|$, i.e., the number of input predictors upon which the forecast is made.

After a random initialization of $\mathbf{W}^\ell \ \forall \ell \in \{1, \ldots, L\}$, their values are kept fixed, which set the stage for a fast and computationally affordable single-step optimization process to compute the values of $\mathbf{w}_o$ [13]. The L_2 norm regularized least square (or ridge regression) provide the $\mathbf{w}_o$ values by solving the following optimization problem:

$$\min_{\mathbf{w}_o} \sum_{t \in \mathcal{T}_{train}} ||y_t - \mathbf{s}_t \cdot \mathbf{w}_o^\top||_2^2 + \lambda ||\mathbf{w}_o||_2^2 \tag{3}$$

where λ is the regularization parameter to be tuned, and $\mathcal{T}_{train}$ denotes the number of training examples. Despite not used in this article, RVFL can also be used for classification by replacing the ridge regression by a matrix pseudoinverse.

The original RVFL algorithm (denoted as *shallow RVFL* in this study) is composed of a single hidden layer, i.e. $L = 1$. By stacking more hidden layers, a RVFL network can be obtained, in which the output layer receives a vector $\mathbf{s}_t$ composed by $\mathbf{x}_t$ (the input itself) and the features produced by the last layer of the hierarchy (e.g. $\mathbf{h}_t^L$). The purpose of the hidden layers is to generate a high-level interpretation of the original input values that supports these input features for computing the final output prediction. This feature representation is given by the random initialization of the architecture parameters.

Among several other variants proposed for classification tasks [24], [25], two new versions of the shallow RVFL were proposed in [14]: the Deep RVFL (dRVFL) and the Ensemble Deep RVFL (edRVFL). The dRVFL algorithm is an extension of a RVFL network where the data interpretation of all hidden layers is handled by the output layer, i.e.:

$$\mathbf{s}_t = [\mathbf{x}_t, \mathbf{h}_t^1, \ldots, \mathbf{h}_t^L], \tag{4}$$

i.e. not only the final feature representation is considered for computing the output, but all the intermediate L hidden feature representations instead.

In the second RVFL variant (edRVFL), intermediate predictions issued from each feature vector produced by every layer are aggregated together to produce the predicted output $\widehat{y}_t$. Mathematically, we extend the notation in Expression (1) to define each of the intermediate predictions as:

$$\widehat{y}_t^\ell = \mathbf{w}_o^\ell \cdot \left(\mathbf{s}_t^\ell\right)^\top = \mathbf{w}_o^\ell \cdot [\mathbf{x}_t, \mathbf{h}_t^\ell]^\top, \tag{5}$$

i.e., the intermediate prediction is modeled based on the feature representation provided by the ℓ-th hidden layer. A separate vector of output weights $\mathbf{w}_o^\ell$ is then computed for every layer as per Expression (3), so that L predictions $\{\widehat{y}_t^\ell\}_{\ell=1}^L$ are obtained for input $\mathbf{x}_t$. Finally, all such predictions are averaged to yield the finally predicted value:

$$\widehat{y}_t = \frac{1}{L}\sum_{\ell=1}^{L}\widehat{y}_t^\ell, \tag{6}$$

yet any other strategy for fusing intermediate predictions can be used (e.g. median value or a stacking ensemble).

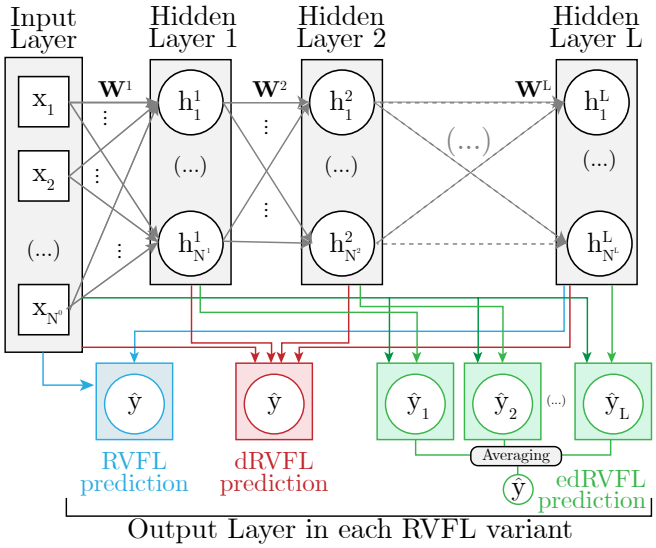

Fig. 1. Structure of the different RVFL variants covered in this study, departing from a generic multi-layer neural architecture with L hidden layers and N^ℓ neurons per layer. The input $\mathbf{x}_t$ and a subset of the hidden features $\{\mathbf{h}_t^\ell\}_{\ell=1}^L$ are utilized in the output layer to compute the final prediction. Only the weights of the connections $\mathbf{W}^\ell$ are randomly initialized. Depending on the values used for the prediction, different RVFL variants can be defined.

III. Experimental Setup

This section describes the experimental setup designed to provide an informed answer to two research questions (RQ):

- RQ1: How do the distinct RVFL variants for road traffic forecasting perform compared to other data driven methods?
- RQ2: Should the instability of randomization based neural networks be the reason for discarding them in favor of other data driven methods?

To address these questions, the experimental setup is split into two case studies. The first one analyzes the performance of several state-of-the-art learning methods for road traffic forecasting, whereas the second case study focuses on the reliability of randomization-based neural networks. Both case studies share the same forecasting problem formulation as a regression task. For a target road, the prediction of a traffic measurement at time $t+h$ is computed by a forecasting model based on the previously collected traffic recordings at the same location and times $\{t-4,\ldots,t\}$. A specialized single-step forecasting model is developed for 4 prediction horizons $h \in \{1,2,3,4\}$. According to these parameters, 10 traffic datasets are built for each of the selected data sources (see Table I). Each dataset comprises one-year traffic measurements from one target location of the traffic network of every source.

To verify the performance of the distinct compared learning methods, data have to be split into training and test sets. As traffic profiles vary between seasons and vacation days, both training and test sets should ideally have recordings from all months. Under this premise, the first three weeks of every month are used for model training, whereas the remaining days are kept for testing.

The chosen traffic forecasting problem requires a regression metric to quantify the performance of the analyzed models. The R^2 score (also referred to as *coefficient of determination*) provides a reliable, numerically bounded measurement of the difference between the real and predicted traffic output. This score is given by:

$$R^2 \doteq 1 - \frac{\sum_{t\in\mathcal{T}_{test}}(y_t-\widehat{y}_t)^2}{\sum_{t\in\mathcal{T}_{test}}(y_t-\bar{y}_t)^2}, \tag{7}$$

where $\mathcal{T}_{test}$ denotes the set of traffic measurements belonging to the test partition of the dataset, y_t denotes the real observed value at test time t, $\bar{y}_t$ its average, and $\widehat{y}_t$ the predicted one.

A. Considered learning methods

For answering RQ1, a collection of learning methods of different nature have to be selected. Along with the previously explained RVFL-based models, a selection of ensemble algorithms and Deep learning architectures is incorporated to our benchmark. In addition, a naive model denoted as *Latest Value* closes the comparison as a performance baseline: the output prediction is given by the last known traffic measurement.

Ensemble methods are composed of multiple weak learners (e.g. decision tree), which provide single predictions that are later combined under distinct strategies towards producing the final prediction of the ensemble model. Ensemble learning methods can differ in terms of 1) the way examples in the dataset are sampled and fed to the weak learners (namely, bagging or boosting [26]); 2) the strategy for aggregating the outputs of the learners; and 3) the base estimators in use, which in turn imposes the way(s) by which the overall ensemble can be configured (for e.g. regularization).

The considered Deep learning architectures are assembled by exploiting the capabilities of three fundamental building units: convolutional, recurrent, and neural. Convolutional units allow expanding the feature space by performing the convolution of a combination of input values. In the context of traffic forecasting, convolutional units can fuse the knowledge from several locations of a traffic network or from close time instants of a single location, potentially inferring more descriptive features therefrom. Likewise, recurrent units process segments of sequential data (e.g. a time series), aiming to craft features that model the evolution of traffic behavior over time. Lastly, neural units are specialized in processing the most meaningful features, ultimately generating the prediction of the overall Deep learning model.

Additionally, another randomization-based neural network besides RVFL variants is analyzed: the so-called Extreme Learning Machine (ELM) [27]. This algorithm works in a similar fashion to the standard RVFL, with the main difference among them being the absence of direct links between the input and output layer featured by ELM-based models. Therefore, the prediction is computed by using only the features provided by the last hidden layer of the neural hierarchy of ELM networks. In order to ensure a fair comparison of the performance of ELM to that of the corresponding RVFL counterparts, the benchmark includes a shallow ELM network with only one hidden layer, as well as an ELM network with multiple hidden layers.

TABLE II
CONSIDERED MODELS FOR ROAD TRAFFIC FORECASTING

Family	Models
Naive	Latest Value [LV]
Randomization based neural networks	Shallow RVFL [sRVFL], RVFL Network [RVFL], Deep RVFL [dRVFL], Ensemble Deep RVFL [edRVFL], Shallow ELM [sELM], ELM Network [ELM]
Ensemble learning	Adaboost [ADA], Random Forest Regressor [RFR], Extremely Randomized Trees Regressor [ETR], Gradient Boosting Regressor [GBR], Extreme Gradient Boosting Regressor [XGBR]
Deep learning	Multi Layer Perceptron [MLP], Convolutional Neural Network [CNN], Recurrent Neural Network based on LSTM units [LSTM], a mixed Convolutional-Recurrent Neural Network [CLSTM], Attention mechanism based Auto-encoder with Convolutional input layers [ATT]

The forecasting performance of all these models can differ depending on the configuration of their hyper-parameters. To override this dependence, a Bayesian optimizer is employed to automate the search for the best hyper-parameters for each learning method [28]. As the combination of the selected traffic datasets and forecasting horizons gives rise to different forecasting scenarios, a dedicated hyper-parameter configuration is established for every analyzed case. For this goal, the Bayesian optimizer evaluates 30 distinct configurations of each algorithm. A three-fold cross-validation process is used to estimate the generalization performance of every model configuration, where two weeks of every month are used for model fit and the remaining data for validation. Once the best hyper-parameter configuration is obtained for every scenario, each model is fit over all training data, and finally the R^2 score is gauged over the data left in the test partition.

B. Performance dispersion due to random weights

RQ2 focuses on randomization based neural networks, so only RVFL, dRVFL, edRVFL and ELM algorithms are under study. The random nature of the weight initialization process causes an statistical dispersion in the distribution of the performance metrics after several test runs. Therefore, the scope of this second case study is to numerically assess this dispersion.

For each analyzed algorithm, several number of hidden layers $L \in \{2, 4, 6, 8, 10\}$ are considered. The number of neurons per layer, which is kept equal for every hidden layer, varies in a discrete range of $N^\ell \in \{1, 10, 50, 100, 500, 1000\}$. To avoid a combinatorial explosion of simulated models, we assume that the number of neurons per layer is equal across layers, e.g. $N^\ell = N^{\ell'}\ \forall \ell, \ell' \in \{1, \ldots, L\} : \ell \neq \ell'$. To determine the stability of these models, every combination of hidden layers and neurons per layer between the above ranges is tested by fitting a model for each training dataset and forecasting horizon $h \in \{1, 2, 3, 4\}$. This process is repeated 100 times, each with a different random seed, issuing 100 R^2 score measurements per configuration. Since training and test data is the same for every test run, the instability associated to the random nature of the initialization process can be isolated.

For the dispersion analysis we resort to the coefficient of quartile variation (CQV), which provides a relative measurement of the dispersion of the obtained test performance metrics of a particular model and configuration (i.e. number of hidden layers and neurons per layer). The CQV of each dataset, model and configuration results from the first (25%) and third (75%) quartiles computed over the 100 R^2 score values obtained during the experimentation, namely:

$$CQV = \frac{Q_3(R^2) - Q_1(R^2)}{Q_3(R^2) + Q_1(R^2)}, \tag{8}$$

where $Q_n(X)$ denotes the n-th quartile of the probability distribution of variable X estimated from a sample of realizations. CQV values close to 1 stand for divergent quartiles and thereby, high statistical dispersion. On the contrary, low CQV values correspond to stable data distributions, since the gap between their first and third quartiles is narrow.

IV. RESULTS AND DISCUSSION

The execution of the experimental setup provides the required outcomes to answer both RQ1 and RQ2. Due to space constraints, only a representative portion of the obtained results from both case studies is portrayed in the manuscript. In any case, all datasets, Python source code, details on the hyper-parameter space considered for every model in the benchmark, and simulation results are publicly available at https://github.com/Eric-L-Manibardo/IJCNN2021.

RQ1: How do the distinct RVFL variants for road traffic forecasting perform compared to other data driven methods?

From all the considered learning methods listed in Table II, nine of them have been selected for this analysis. This selected subset of models allows for a more focused discussion on the differences found between the considered family of models. Nevertheless, results corresponding to all models (including those not shown in what follows) can be accessed in the repository linked above.

First, the naive method LV provides a hint that helps understanding the difficulty of the task at hand, acting as a baseline for other methods. The obtained performance score set a baseline boundary that should be exceeded by any other learning method for it to be considered as a valuable forecasting model. All randomization based neural networks

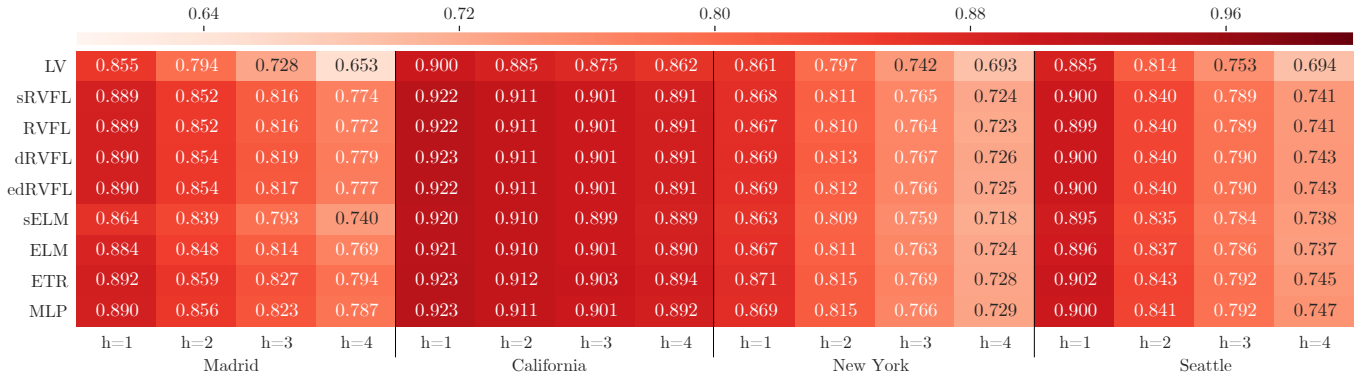

Fig. 2. Average R^2 test score for all the analyzed models. Each cell in the table displays the average R^2 value of one of the analyzed models, computed as the mean score from the 10 target locations of a certain traffic network and forecasting horizon. Row names indicate the analyzed learning method, whereas columns stand for the forecasting horizon h and the corresponding data source.

are included in the analysis. From the set of ensemble learning algorithms, only results corresponding to ETR are displayed. Ensemble methods have obtained similar results, so the best performing algorithm from this category is selected. Finally, Deep learning architectures also yield comparable performance scores. However, in this case, it is interesting to include the results of MLP as this method has a similar architecture to that of its randomization based counterparts (i.e. RVFL and ELM). The number of neurons, and other building blocks of these architectures are set according to the hyper-parameter configuration obtained after the Bayesian optimization.

Figure 2 summarizes the obtained R^2 test scores for each model, data source, and forecasting horizon combination. Results in this figure are calculated as the average of the 10 datasets extracted from each data source. The distinct scope and traffic measurement type of the selected data sources directly impact the complexity of the forecasting problem. The California data source (i.e. Caltrans Performance Measurement System (PeMS) [20]) collects traffic data from highways, where the traffic flow maintains a predictable profile. As such, even the LV model delivers good performance at $h = 4$, which is the most difficult considered scenario for every data source. Results obtained for the other data sources, however, degrade when the forecasting horizon increases, so the remainder of the discussion is focused on those diverging outcomes.

Focusing on the results obtained by each model, the analyzed RVFL variants provide similar performance metrics. This behavior is mainly explained by the direct links between input and output layer (see Figure 1). As previously explained, the self-descriptive nature of traffic measurements makes these input data act as high-quality predictors by themselves. Raw traffic information can (and *must*) be taken into consideration when furnishing the traffic forecast. Since all considered RVFL variants share this capability through their direct input connection, the test score remains almost identical for several of the analyzed scenarios.

The other considered randomization based neural networks provide remarkable results, but always below the performance of RVFL neural networks. In particular, sELM delivers the worst results of the benchmark after the naive baseline model LV. The ELM neural network outperforms its shallow variant for every scenario, which makes sense since it has more hidden layers and therefore can elaborate more complex feature representations from the incoming traffic data. Given the absence of direct links between the input and output layer, the ELM neural networks can only rely on the self-crafted features, granting the advantage to the architecture with more hidden layers.

Sharing the same number of hidden layers than the analyzed non-shallow randomization based neural networks, the MLP provide similar results. Only at the $h = 3$ and $h = 4$ the performance starts to differ from that obtained by RVFL variants. However, the increased computational cost associated to adjusting the inner weights via back-propagation has to be kept into consideration. A faster and more affordable training phase can be interesting in multiple scenarios like those analyzed in the current case study, bearing in mind the narrow gap between the R^2 test scores reflected in the figure.

Finally, ETR has obtained the best score metrics among all analyzed methods. The ensemble nature of this method makes it better fit the training traffic data. Even so, we can draw a similar conclusion than that held for back-propagation based architectures. Ensemble methods usually stand high in the ranks of every performance benchmark. By merging the outputs of several base learners, ensembles ensure that their overall performance does not get biased by potentially noisy training samples.

Nevertheless, the computational cost of adjusting several base learners to data distribution can be unaffordable when deployed on devices undergoing severely restricted computational resources. In those circumstances, randomization based neural networks approaches can achieve similar performance results but they also afford reduced computational requirements. Training time for all RVFL based methods is consistently under 0.2 seconds, while ensemble learning models require training times 10 to 20 times larger. In the case of Deep learning based methods, the training time goes from 9 seconds for MLP to 95 seconds for ATT. This showcases the relevant time-consumption improvement that these approaches provide,

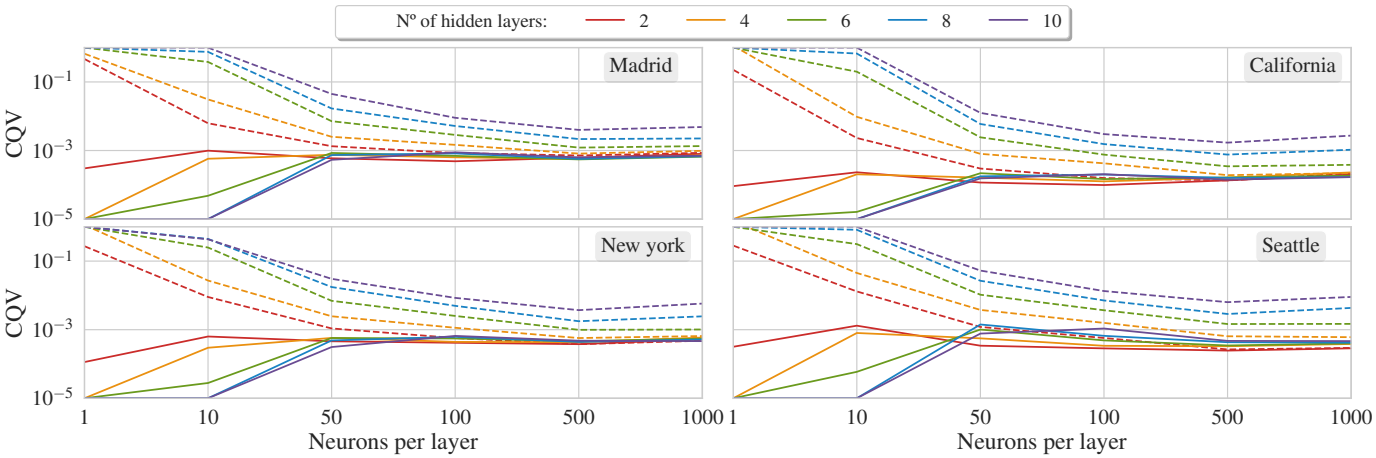

Fig. 3. CQV of RVFL and ELM for $h = 1$. The CQV obtained for the 10 datasets of every data source is averaged and displayed for each subplot. Lines are drawn according to the number of neurons per layer (X axis) and the number of hidden layers (color of every curve in the plot). Continuous lines stand for RVFL models while dashed lines for ELM models.

particularly interesting for limited hardware environments. Training times were computed on an Intel Xeon Gold 5118 CPU, 512 GB RAM and 4 Tesla V100 GPU server.

RQ2: Should the instability of randomization based neural networks be the reason for discarding them in favor of other data driven methods?

We now proceed by inspecting the dispersion of the R^2 performance scores resulting from repeatedly training the randomization based neural networks under consideration, each with a different random seed. The amount of hidden layers and neurons per layer is selected according to the ranges described in Section III-B. For each configuration, a R^2 performance distribution can be obtained from 100 test executions, where the shape of each distribution is directly related to the stability of the model under analysis. In this line, the selection of one data source or another only impacts on the median of the distribution, but is the architecture design what modifies the dispersion of the performance. Therefore, the same insights can be distilled from any of the subplots available in Figure 3. From the obtained 100 R^2 performance measurements, the $Q_1(R^2)$ and $Q_3(R^2)$ values are used to compute the CQV value as per Equation 8. Similarly to the previous subsection, only results for $h = 1$ are shown in the Figure 3, due to space limitations. Additionally, the different RVFL variants have obtained similar dispersion metrics, so only RVFL and ELM neural networks are displayed.

Both neural networks behave opposite for a low number of neurons per layer. On one hand, the direct link between the input and output layers endows RVFL architectures with stable results when they have a few neurons per layer. With such a small amount of hidden features, the optimization process grants more attention to the input features (due to their self-descriptive nature), which are not affected by the random initialization of the architecture weights. Therefore, the predictive behavior remains roughly unaltered, providing a low dispersion for the test results. In the case of RVFL architectures with a high number of layers but few neurons, the incoming input values are overprocessed, providing a set of hidden features that are overly unrelated to the original traffic measurements. Consequently, the optimization process grant even less attention to these hidden features. In this way, the behavior of the model remains apparently unaltered, disregarding the initialization values of the neural parameters.

On the other hand, ELM only relies on hidden features when issuing its prediction. Those configurations with a low number of neurons per layer cannot produce a reliable set of hidden features that contain enough knowledge to perform a successful prediction. In these cases, the initialization of the weights produces a huge impact on the quality of the hidden features. In contrast to RVFL, ELM reduces the statistical dispersion of its results for architectures of a few hidden layers. The input values are processed multiple times after each hidden layer, so the impact of badly initialized weights produce more disparate hidden representations after a high amount of neural layers, regarding more restrained architectures.

Finally, both neural networks converge to similar CQV values for a high amount of neurons per layer, which implies that the dispersion of R^2 performance along executions is similar. RVFL increases the variability of its results as hidden features become more significant and therefore have an increased impact on the output prediction. In the case of ELM, a high number of neurons results in a likewise higher number of hidden features and hence, more chances for the output layer to be fed with informative features towards computing the final output prediction.

V. Conclusions and Future Work

In this work we have showcased the capability of RVFL based architectures to model road traffic data. To this end, a comparative performance benchmark is conducted for a traffic forecasting problem with real-world data collected from different geographical locations. An autonomous hyperparameter tuning via a Bayesian optimizer ensures the fair

comparison of the models, without any bias with respect to the optimality of their hyperparametric configuration. Our results are conclusive: the predictive performance of RVFL variants match those provided by Deep learning models and ensemble methods. However, the faster training time and less computational effort demanded by these randomization based neural networks make them an appealing choice under severe restrictions in terms of computational resources.

Another issue covered in this work is the interplay between the initialization process and the stability in the prediction outputs of randomization based neural networks. Our experimentation reveals that a low dispersion in the performance distribution can be achieved by understanding the behavior of the distinct architectures. Bearing in mind the self-descriptive nature of traffic data, RVFL-based models offer an stable performance for all the considered architectures, with special emphasis on those having few neurons per layer. Since input features are not affected by the random initialization process, the hidden features have less influence on the forecasting output. The other analyzed randomization based neural networks, however, compute the output prediction based exclusively on hidden features. Therefore, stable results are only guaranteed for high amounts of neurons per layer. Only in this way the model has access to multiple features, part of which can be informative for the traffic variable under target.

Several research lines are planned for the near future. To begin with, we will delve into the effects caused by the direct links between input and output layer on the epistemic uncertainty of the prediction produced by randomization-based neural networks. Another direction to be pursued is to perform a sensitivity analysis of the input with respect to its output, aiming to unveil which predictors push statistically the output towards higher or lower values of the target variable. Finally, we plan to investigate how modern learning paradigms that have been so far tackled with conventional neural architectures can be extrapolated to randomization-based neural networks, including transfer and multitask learning. We envision that the mechanisms of knowledge transfer between networks with randomly initialized parameters are still to be explored.

ACKNOWLEDGMENTS

The authors would like to thank the Basque Government for its funding support through the ELKARTEK program (3KIA project, KK-2020/00049), the BIKAINTEK PhD support program (grant no. 48AFW22019-00002), and the consolidated research group MATHMODE (ref. T1294-19).

REFERENCES

[1] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.

[2] I. Laña, J. Del Ser, M. Velez, and E. I. Vlahogianni, "Road traffic forecasting: Recent advances and new challenges," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 2, pp. 93–109, 2018.

[3] P. Næss and A. Strand, "Traffic forecasting at 'strategic','tactical'and 'operational'level: A differentiated methodology is necessary," *disP-The Planning Review*, vol. 51, no. 2, pp. 41–48, 2015.

[4] E. L. Manibardo, I. Laña, and J. Del Ser, "Deep learning for road traffic forecasting: Does it make a difference?" *arXiv preprint arXiv:2012.02260*, 2020.

[5] K. Lee, M. Eo, E. Jung, Y. Yoon, and W. Rhee, "Short-term traffic prediction with deep neural networks: A survey," *arXiv preprint arXiv:2009.00712*, 2020.

[6] B. Liao, J. Zhang, C. Wu, D. McIlwraith, T. Chen, S. Yang, Y. Guo, and F. Wu, "Deep sequence learning with auxiliary information for traffic prediction," in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 537–546.

[7] Y. Zhang, T. Cheng, and Y. Ren, "A graph deep learning method for short-term traffic forecasting on large road networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 34, no. 10, pp. 877–896, 2019.

[8] S. Ryu and D. Kim, "Intelligent highway traffic forecast based on deep learning and restructured road models," in *Computer Software and Applications Conference*, vol. 2, 2019, pp. 110–114.

[9] Y. Ren, H. Chen, Y. Han, T. Cheng, Y. Zhang, and G. Chen, "A hybrid integrated deep learning model for the prediction of citywide spatio-temporal flow volumes," *International Journal of Geographical Information Science*, vol. 34, no. 4, pp. 802–823, 2020.

[10] F. M. Awan, R. Minerva, and N. Crespi, "Improving road traffic forecasting using air pollution and atmospheric data: Experiments based on LSTM recurrent neural networks," *Sensors*, vol. 20, no. 13, p. 3749, 2020.

[11] A. Maier, C. Syben, T. Lasser, and C. Riess, "A gentle introduction to deep learning in medical image processing," *Zeitschrift für Medizinische Physik*, vol. 29, no. 2, pp. 86–101, 2019.

[12] L. Zhang and P. N. Suganthan, "A survey of randomized algorithms for training neural networks," *Information Sciences*, vol. 364, pp. 146–155, 2016.

[13] ——, "A comprehensive evaluation of random vector functional link networks," *Information Sciences*, vol. 367, pp. 1094–1105, 2016.

[14] R. Katuwal, P. N. Suganthan, and M. Tanveer, "Random vector functional link neural network based ensemble deep learning," *arXiv preprint arXiv:1907.00350*, 2019.

[15] P. N. Suganthan and R. Katuwal, "On the origins of randomization-based feedforward neural networks," *Applied Soft Computing*, p. 107239, 2021.

[16] Y. Ren, P. N. Suganthan, N. Srikanth, and G. Amaratunga, "Random vector functional link network for short-term electricity load demand forecasting," *Information Sciences*, vol. 367, pp. 1078–1093, 2016.

[17] J. Del Ser, D. Casillas-Perez, L. Cornejo-Bueno, L. Prieto-Godino, J. Sanz-Justo, C. Casanova-Mateo, and S. Salcedo-Sanz, "Randomization-based machine learning in renewable energy prediction problems: Critical literature review, new results and perspectives," *arXiv preprint arXiv:2103.14624*, 2021.

[18] N. Geroliminis and J. Sun, "Properties of a well-defined macroscopic fundamental diagram for urban traffic," *Transportation Research Part B: Methodological*, vol. 45, no. 3, pp. 605–617, 2011.

[19] "Madrid Open Data," http://datos.madrid.es, accessed: 2020-11-06.

[20] "Caltrans, Performance Measurement System," http://pems.dot.ca.gov, accessed: 2020-11-06.

[21] "NYC Real Time Traffic Speed Data Feed," https://www.kaggle.com/crailtap/nyc-real-time-traffic-speed-data-feed, accessed: 2020-11-06.

[22] "Seattle Inductive Loop Detector Dataset," https://github.com/zhiyongc/Seattle-Loop-Data, accessed: 2020-11-06.

[23] D. Husmeier, "Random vector functional link networks," in *Neural Networks for Conditional Probability Estimation*, 1999, pp. 87–97.

[24] M. Ganaie, M. Tanveer, and P. Suganthan, "Minimum variance embedded random vector functional link network," in *International Conference on Neural Information Processing*. Springer, 2020, pp. 412–419.

[25] M. Tanveer, M. Ganaie, and P. Suganthan, "Ensemble of classification models with weighted functional link network," *Applied Soft Computing*, p. 107322, 2021.

[26] S. González, S. García, J. Del Ser, L. Rokach, and F. Herrera, "A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities," *Information Fusion*, vol. 64, pp. 205–237, 2020.

[27] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[28] J. Bergstra, D. Yamins, and D. D. Cox, "Hyperopt: A Python library for optimizing the hyperparameters of machine learning algorithms," in *Python in Science Conference*, 2013, pp. 13–20.