FocusLearn: Fully-Interpretable, High-Performance Modular Neural Networks for Time Series

Qiqi Su*
Department of Computer Science
City, University of London
London, UK

Christos Kloukinas
Department of Computer Science
City, University of London
London, UK

Artur d'Avila Garcez
Department of Computer Science
City, University of London
London, UK

Abstract

Multivariate time series have had many applications in areas from healthcare and finance to meteorology and life sciences. Although deep neural networks have shown excellent predictive performance for time series, they have been criticised for being non-interpretable. Neural Additive Models, however, are known to be fully-interpretable by construction, but may achieve far lower predictive performance than deep networks when applied to time series. This paper introduces FocusLearn, a fully-interpretable modular neural network capable of matching or surpassing the predictive performance of deep networks trained on multivariate time series. In FocusLearn, a recurrent neural network learns the temporal dependencies in the data, while a multi-headed attention layer learns to weight selected features while also suppressing redundant features. Modular neural networks are then trained in parallel and independently, one for each selected feature. This modular approach allows the user to inspect how features influence outcomes in the exact same way as with additive models. Experimental results show that this new approach outperforms additive models in both regression and classification of time series tasks, achieving predictive performance that is comparable to state-of-the-art, non-interpretable deep networks applied to time series.

1 Introduction

Deep Neural Networks (DNNs) are a popular method for analysing multivariate time series. DNNs can identify hidden patterns in data to obtain accurate approximations of time series [16]. Unlike classical statistical methods, DNNs make no assumptions about the statistical distribution of the underlying time series [26]. For this reason, they have been shown to improve on more traditional linear time series models such as Support Vector Machines and auto-regressive models [55]. Transformer-based models and attention mechanisms, originally introduced for natural language processing, have recently been shown to achieve even higher accuracy at forecasting time series [10, 17] due to their ability to capture long-term dependencies in data.

*Correspondence to: Qiqi Su <qiqi.su@city.ac.uk>

Despite DNNs' success, there is now widespread recognition that just effective predictive performance is insufficient; DNNs need to be interpretable or explainable, especially in safety-critical applications. As a result, the field of eXplainable Artificial Intelligence (XAI) has gained traction in recent years. XAI seeks to make DNNs more transparent and understandable to humans, and to provide greater trust in AI decisions. Interpretability approaches analyse the model's weights and features when determining a given output. Explainability seeks to derive meaning from the input data and model outputs to map out the behaviour of the model. XAI has been applied to help developers improve model learning, to help users make more informed decisions, and to identify biases in the trained models [34, 49, 52].

XAI research can be divided into post-hoc and ante-hoc. Post-hoc methods aim to explain a trained model. Ante-hoc methods seek to incorporate interpretability into model training. Post-hoc methods can only approximate the underlying model, which can make the explanations less faithful. Ante-hoc approaches may be less flexible as they prescribe a specific architecture or training regime. In general, a trade-off exists between accuracy and interpretability in ante-hoc methods. It is important to develop ante-hoc interpretable DNNs that can outperform traditional methods deemed to be inherently-interpretable, such as decision trees, while matching the performance of non-interpretable approaches [13].

To achieve an interpretable DNN for time series, we draw inspiration from modular networks [7] and additive models [2]. Our approach builds on the evidence pointing to the interpretability of linear directions in activation space, particularly linear combinations of neuron activations [6]. Our proposed architecture learns a linear combination of modular networks, each attending to a single selected feature. This process aims to decompose the complexity of DNNs into fundamental units of features, as done by Neural Additive Models (NAMs) [2]. Differently from [6], we emphasise the relevance of modularity during the learning process, highlighting its significance in steering the model away from convoluted structures that may hinder interpretability. Leveraging the NAM approach, we aggregate the contributions of individual neural networks by summing them up. The incorporation of modularity at learning time in our methodology will be shown to enhance interpretability compared to DNNs. It will be shown that learned modularity allows one to analyse and systematically discern the contributions of each feature, facilitating a more transparent understanding of the DNN's decision-making.

Following the neural additive approach, we propose FocusLearn, which is able to handle both time series forecasting and classification tasks for both large and small data sets. FocusLearn provides feature importance and the exact depiction of how the model arrives at a certain outcome, offering interpretability at the same level as NAMs. In addition, FocusLearn outperforms state-of-the-art interpretable methods, such as NAM itself, and it matches the performance of classic non-interpretable models on all four domains considered in this paper. FocusLearn achieves these results by introducing two novel ideas: *(i) it uses an attention mechanism for learning the relevant features*; and *(ii) it reuses the attention weights when training each modular network on the final prediction*, as detailed in what follows.

The paper is organised as follows. Section II discusses Related Work. Section III introduces FocusLearn. Section IV contains the main experimental results. Section V elaborates on and discusses variations of the experiments, and Section VI concludes the paper and discusses directions for future work.

2 Related Work

Popular XAI methods *Local Interpretable Model-agnostic Explanation* (LIME) [37] and *SHapley Additive exPlanation* (SHAP) [33] have been introduced in the hope of overcoming the accuracy-interpretability trade-off in DNNs. However, these methods suffer from low fidelity [1, 51], instability [18], and even inaccuracy [29, 39] as they seek to approximate the underlying model.

Many ante-hoc methods have been proposed for time series analysis, including Shaplets [15, 53], Symbolic Aggregate Approximation [42], and Fuzzy Logic-based XAI [35, 50]. Often, these methods address either regression or classification tasks, but not both. More importantly, these methods do not seem to scale well with large data sets.

2.1 Interpretable Additive Models

Adding interpretability to Generalised Additive Models (GAM) [21] has been shown applicable to both time series regression and classification problems. GAM variants such as NAM [2], GA^{2}M [32], Neural Interaction Transparency [45] and Neural Basis Model [36], are computationally efficient and can handle large data sets. For instance, NAM [2], an interpretable variant of GAM, learns a linear combination of a family of interpretable neural networks, each assigned to learn a single input feature. This family of networks is trained jointly using backpropagation and it can learn arbitrarily complex shape functions. The impact of a feature on the prediction can then be interpreted by plotting its corresponding shape function along with the coefficients of the linear combination. As such, this plot is the exact depiction of how NAM arrives at a prediction, under the assumption that the pre-defined separation of features holds. NAM is also capable of learning non-linear relationships, making it an attractive choice for many applications. Unfortunately, the predictive performance of these models is no match for more popular, non-interpretable time series models, so that they cannot be regarded as a substitute for non-interpretable models [13].

The *Scalable Polynomial Additive Model* (SPAM) [13] builds on NAM by allowing higher-order interactions between features. SPAM is intended to replace non-interpretable models for large scale data due to its higher performance than NAMs. The main difference between SPAM and NAM is that SPAM uses tensor rank decomposition of polynomials such that higher-order feature interactions can be learned. However, the explanations provided by SPAM cannot be readily visualised in the same way as NAM and, as a result, they are harder to interpret than NAM.

2.2 Other Relevant Topics

2.2.1 DNN for Multivariate Time Series Forecasting

Recurrent neural networks (RNN) [14] are very popular for analysing multivariate time series as they remember past inputs to decide on future outcomes. Many variants of RNN have been proposed over the years such as Long Short-Term Memory (LSTM) [22], Bidirectional LSTM (BLSTM) [41], and Gated Recurrent Units (GRU) [9]. LSTM and GRU were introduced to overcome the gradient vanishing problem faced by RNNs [5], and in turn have been shown to be useful for learning long-term dependencies [12]. Since LSTM has a more complex architecture than GRU, training a GRU requires fewer computational resources. On the other hand, LSTM can handle large amounts of data with more parameters to optimise performance. LSTM is a very popular choice for modelling multivariate time series problems across a variety of applications [23, 40, 44], although they are considered to be non-interpretable.

2.2.2 Attention Mechanism for Feature Selection

Attention mechanisms have been used as a way to provide interpretability by offering insight into the most relevant features. Temporal Fusion Transformer (TFT) [28] was introduced for a multi-horizon forecasting task with added interpretability by visualising the persistent temporal pattern using attention. A limitation of TFT is that it requires a large amount of data to achieve good predictive performance [54]. Interpretable Temporal Attention Network (ITANet) [57] used an attention mechanism to infer the importance of government interventions for COVID19 predictions. ITANet only provides feature importance as an explanation.

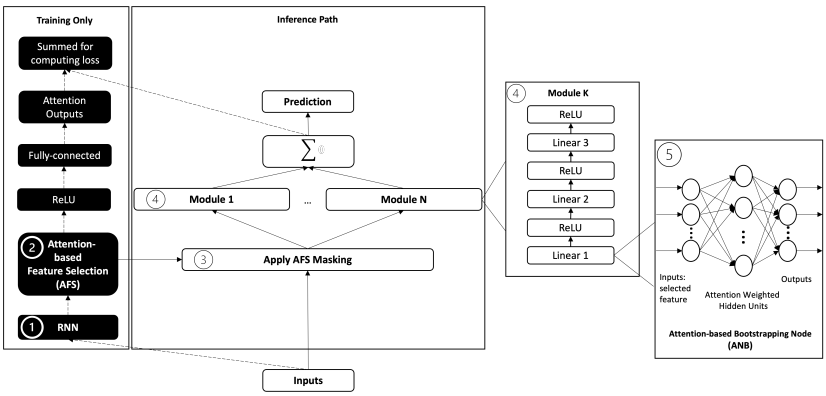


Figure 1: **Attention Modular Networks (FocusLearn) architecture.** FocusLearn consists of two main paths (training and inference) with five main components: ① *Recurrent Neural Network (RNN)* (see Section 3.1), ② *Attention-based Feature Selection (AFS)* with ③ AFS masking (see Section 3.2), and ④ a group of *interpretable Modular Neural Networks (MNN)* that learn from the top n input features selected by the AFS (see Section 3.3). The ⑤ *Attention-based Node Bootstrapping (ANB)* in each *module*'s first layer is weighted by the *AFS*'s attention weights (see Section 3.3). MNNs outputs are then aggregated. All components are required for training, but to maintain the interpretability of FocusLearn, only the components in the *inference path* box are used after training (see Section 3.6).

Feature selection has been an effective approach for preparing high-dimensional data for a variety of Machine Learning tasks [20]. Several different architectures that incorporate attention into DNNs for feature selection have been proposed recently, *e.g.,* Attention-based Feature Selection (AFS) [20], Multiattention-based Feature Selection (MFS) [8], and Attentive Interpretable Tabular Learning (TabNet) [3]. Attention mechanisms seek to learn the temporal relationships in the data, especially when used in conjunction with RNNs, better than traditional feature selection methods — wrapper feature selection methods tend to suffer from high computational complexity, while filter methods evaluate feature weights using only generic characteristics of the data [8, 20]. In AFS and MFS, several attention mechanisms are utilised in an *attention module* to select features directly from the input. A *learning module* then deploys different DNNs to learn from the selected inputs. In both AFS and MFS, interpretability only exists in the *attention module* and not in the learning module, the main part of the system. TabNet introduced a sequential attention mechanism to choose which features to use for each decision step. Although TabNet achieved state-of-the-art results compared to non-interpretable DNNs, the explanations provided by TabNet only concern feature attributions.

3 FocusLearn– An Attention Modular Network

FocusLearn[2] is in the category of interpretable additive models, such as NAM and SPAM. Although, like FocusLearn, NAM can provide both feature importance and explanation plots for model outcomes, its predictive performance can be poor (see Section IV). SPAM addresses the issue of performance but only offers feature importance explanations. As a result, the goal of this research was to develop a model that performs better than NAM while maintaining NAM's level of explainability. To accomplish this, inspiration was taken from AFS and MFS, where an attention mechanism is used for feature selection. Learning only the most salient features has been shown to be a useful method for improving interpretability and for allocating learning capacity more effectively [8, 20]. The entirety of the inference path of FocusLearn is interpretable since the computation for prediction is simply a series of linear combinations of interpretable networks. Predictive performance is improved, not by making that step nonlinear, but from a better selection of features using attention to learn the modular networks and to ignore features that have no contribution to the model.

[2] `https://github.com/qisuqi/FocusLearn`

3.1 Recurrent Neural Network (RNN)

Inputs are first processed by the RNN component, shown in Fig. 1 ①, which is responsible for learning the time-dependencies in the input data. The RNN component used in this paper is an LSTM network, but it can be replaced easily by an alternative architecture such as GRU, BLSTM or Transformer [25]. In the hidden layer of an LSTM, there are some special units called memory cells that are recurrently connected, as well as their corresponding gate units, namely input gate, forget gate, and output gate [22]. To avoid the problem of vanishing or exploding gradients and unstable hidden representations due to the highly dynamic nature of their state updates, layer normalisation is applied to the RNN outputs [4]. It is responsible for re-centering and re-scaling the activation values to obtain zero mean and unit variance, and facilitate the propagation of gradients through the network.

3.2 Attention-based Feature Selection (AFS)

This component of FocusLearn is inspired by the Multi-head Attention (MHA) proposed in [47]. Using an attention mechanism for feature importance or feature selection is often criticised, as using the attention weights of the MHA alone cannot be indicative of the importance of a feature, given that different weights assigned to different parts of the input data are used in each head [28]. Therefore, instead of making each head attend to a sub-sequence of the input, the MHA is modified here so each attention head computes the attention weight of each input feature independently. It is worth noting that this modification does not change the mathematical concept of the original MHA – it merely changes how the attention is applied.

$$AH_i = \text{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V) \tag{1}$$

$$\text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = [AH_1, AH_2, ..., AH_n]\boldsymbol{W}_{AH} \tag{2}$$

Given the final hidden states of the RNN component, $H = h_1, h_2, ..., h_T$, the attention function in AFS, (1), is applied to each H_i, where each AH_i is an attention head of the MHA, and $\boldsymbol{Q}\boldsymbol{W}_i^Q \in \mathbb{R}^{d_q \times d_v}$, $\boldsymbol{K}\boldsymbol{W}_i^K \in \mathbb{R}^{d_k \times d_v}$, $\boldsymbol{V}\boldsymbol{W}_i^V \in \mathbb{R}^{d_q \times d_v}$ are the projections of attention weights for $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$, respectively. In (2), $\boldsymbol{W}_{AH} \in \mathbb{R}^{(AH \cdot d_v) \times d_q}$ linearly combines concatenated outputs from all attention heads, thus completing the MHA function. Finally, $\boldsymbol{W}_{AH}$ is averaged to compute the attention weights for each H_i by applying a feature-wise pooling operation. The output of the AFS component, $\boldsymbol{F_i}$, is obtained by applying the $\mathrm{softmax}$ function to the averaged attention weights, $\text{Attn}(H_i)$, such that $\boldsymbol{F_i} = \mathrm{softmax}(\text{Attn}(H_i))$. F_i allows us to select the top n features from the inputs in component ③ of FocusLearn, *Apply AFS Masking*, Fig. 1 ③, filtering out features that do not contribute much to model prediction. In FocusLearn, n can be pre-defined or a hyper-parameter. To ensure that gradients are propagated adequately through the MNN (discussed in more detail in Section 3.3), $\boldsymbol{F}$ is put through a Rectified Linear Unit (ReLU) and a fully-connected layer so that the loss of the RNN-AFS components can be calculated (discussed in more detail in Section 3.5).

3.3 Modular Neural Networks (MNN)

The idea of modularity has been researched as early as the 1980s and adopted in neural information processing in the late 1990s [7]. It was recognised then that traditional neural networks are black-boxes and MNNs were more interpretable alternatives. MNNs follow the divide-and-conquer principle [46] and are inspired by the important biological fact that neurons in human brains are sparsely connected in a clustered and hierarchical fashion, rather than completely connected [7]. MNNs use several smaller simple Neural Networks (NNs), where each NN focuses on a different part of the same problem and their outputs are combined together into the outcome for the entire network. In this way, MNNs break down complex learning problems in resolving large-scale tasks.

Each module in the MNN component of FocusLearn consists of a single selected input feature trained independently, so that the impact of each feature on the prediction is independent of other features. It is worth noting that the MNN component reads the top n features directly from the inputs. Thus, this component methodologically belong to the GAM family, $g(\mathbb{E}(y)) = \beta + f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n)$, where y is the target variable, g is the link function, $\boldsymbol{x} = x_1, x_2, ..., x_n$ is the input (or the selected input in the case of FocusLearn), and each f_i is a univariate shape function with $\mathbb{E}(f_i) = 0$.

The MNN component renders the FocusLearn interpretable because each univariate shape function is parameterised by a NN through a series of linear transformations, as done with NAM. The main difference between FocusLearn and NAM is that each feature is initially weighted using the attention weights $\boldsymbol{F}$. This will be shown to improve predictive performance considerably compared to NAM. To achieve feature weighting, we introduce Attention-based Node Bootstrapping (ANB) to each module (see Fig. 1 ④). The objective is to learn the weights based on the inputs factored by the attention weights and shifted by a bias. For each module, Xavier Initialisation [19] sets the biases (to zeros) and weights at each layer. The previously-computed attention weights for the selected features, $\boldsymbol{F}$, are then multiplied by the initialised weights, $\boldsymbol{W}_{\text{init}}$, as in (3). Finally, the ReLU activation function is applied to each ANB for each scalar input from x, as in (4). In practice, ANB is applied to the first layer of each module (Fig. 1 ④, Linear 1) so each $\boldsymbol{F_i}$ can be used directly. The modular networks are then trained jointly with the RNN and AFS components using back-propagation, as follows.

$$\boldsymbol{W}_{\text{mod}_i} = \boldsymbol{W}_{\text{init}_i} * \boldsymbol{F_i} \tag{3}$$

$$ANB(x) = \text{ReLU}((x - b) * \boldsymbol{W}_{\text{mod}}) \tag{4}$$

3.4 Learning Rate Scheduler

Since FocusLearn resembles the Transformer architecture, we employ the Cosine Annealing learning rate (LR) warm-up scheduler [31] to further accelerate convergence, stabilise the training, and hopefully improve generalisation using the Adam optimisation algorithm [30]. Unlike fixed or decreasing LR strategies, the LR warm-up gradually increases LR from zero to the specified LR in initial training iterations. This is particularly important for adaptive optimisation, as in the case of Adam, which leverage bias correction factors, potentially causing higher variance in early iterations. Additionally, the RNN component's use of layer normalisation can contribute to elevated gradients in the initial phases.

3.5 Loss Functions

To ensure that gradients flow through all components of FocusLearn and improve convergence during training, the RNN-AFS and MNN components are trained jointly, with each one aiming to minimise the Mean Squared Error Loss ($\frac{1}{N}\sum(y_i - \tilde{y_i})^2$) in the case of regression tasks, and a Binary Cross-Entropy with Logits Loss ($-\frac{1}{N}\sum_{i=1}^{N}(y_i * log(\tilde{y_i}) + (1 - y_i) * log(1 - \tilde{y_i}))$) in the case of classification tasks, where y is the target value, $\tilde{y_i}$ is the predicted value, and N is the number of data points. The FocusLearn loss function to minimise is the sum: $\text{Loss}_{\text{FocusLearn}} = \text{Loss}_{\text{RNN-AFS}} + \text{Loss}_{\text{mod}}$.

3.6 Training and Inference Path

All components of FocusLearn are required in the training phase, but only the inference path is required at deployment time. To ensure the explainability of the deployed model, the inference path includes by construction only the MNN component. This means that at inference time the RNN-AFS components are excluded from the forward pass. Without gradients traversing through the RNN-AFS components, the utility of ANB remains unrealised, as the gradients would not be

back-propagated to the original inputs. In response to this, we implemented a streamlined inference path where components linked by dashed arrows in the *training only* box of Fig. 1 are omitted. This means that the RNN-AFS components exclusively handle feature selection during training, and once the top n features are selected, FocusLearn becomes as explainable as NAM, as it no longer relies on the excluded components for decision-making.

4 Experimental Results

Predictive performance was evaluated on two regression and two classification tasks against state-of-the-art interpretable and non-interpretable methods[3]:

- **Neural Additive Model (NAM)** [2]. FocusLearn is an extension of NAM with the objective of maintaining interpretability while improving upon its predictive performance;
- **Scalable Polynomial Additive Model (SPAM)** [13]. Although SPAM uses feature interactions to produce more accurate predictions than NAM, SPAM cannot produce NAM-like feature curves as explanations;
- **Long Short-Term Memory (LSTM)** [22]. Stand-alone LSTM offers a baseline for evaluating multivariate time series models, but it is non-interpretable;
- **Extreme Gradient Boosted Trees (XGBoost)**[4]. Although a single tree can be interpretable, interpretability degrades rapidly as the number of trees grows. Both NAM and SPAM are compared with XGBoost in earlier work.

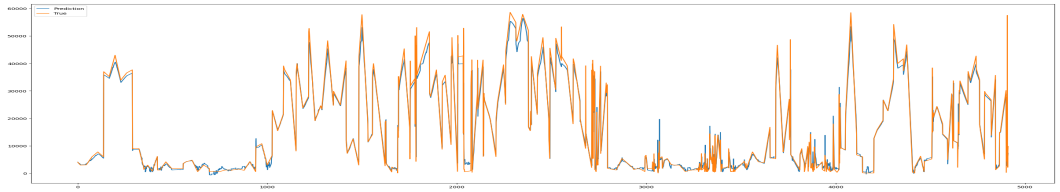
Experimental results are summarised in Table 1. The top 10 features are selected for all data sets (*i.e.,* we use MNN with 10 modules). FocusLearn outperforms both interpretable models NAM and SPAM[5] in both regression and classification tasks. This happened for every standard metric used, namely, Symmetric Mean Absolute Percentage Error (sMAPE), Mean Absolute Scaled Error (MASE) and Weighted Absolute Percentage Error (WAPE) for regression, accuracy, F1 score and Area Under the Curve (AUC) for classification tasks. FocusLearn also has significantly faster computation speed than both NAM and SPAM.

As Table 1 shows, FocusLearn also matches the performance of non-interpretable approaches on the *OtiReal* and *Weather* data sets, or it outperforms non-interpretable approaches (*Air* and *EEG* data sets). In the case of the *OtiReal* results, FocusLearn and LSTM had the lowest SMAPE and WAPE but did not have a good MASE. Reversely, XGBoost had the lowest MASE error, but its SMAPE and WAPE errors were among the worst. Both SMAPE and WAPE are percentage-based error estimators, whereas MASE evaluates the ability of a model relative to a simple benchmark model. Therefore, given the discrepancy in results depending on the choice of metric, a visual inspection of the prediction results would be helpful to decide which model performed better in the *OtiReal* data set. While metrics are essential for quantitative evaluation, the ultimate goal is to achieve predictions that align with the underlying pattern in the data and expert intuition. Fig. 2 shows that FocusLearn and LSTM are better at capturing the pattern and the trend in the *OtiReal* data set than XGBoost: the predicted values are much better aligned with the true values. Fig. 2 also shows that the magnitude of the true values vary significantly in the *OtiReal* data set and both SMAPE and WAPE are better at comparing models in this situation than MASE.

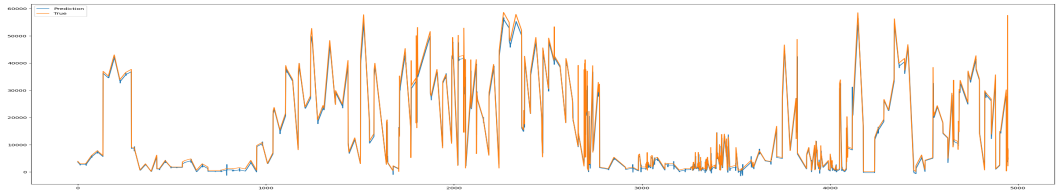
[3] Although NAM was not explicitly designed for time series, its flexibility allowed it to be applied to time series data [24, 48].

[4] `https://xgboost.readthedocs.io/en/stable/`

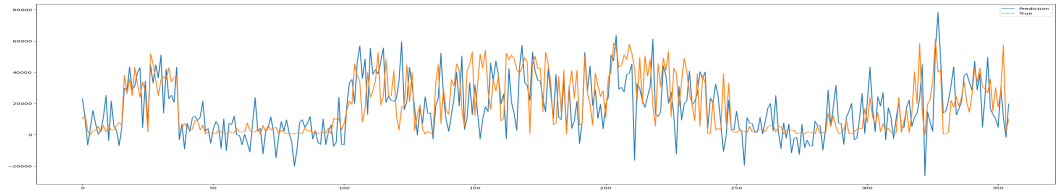
[5] NAM and SPAM implementations lacked an *inference step* for making predictions on unseen data. Results reported here are based on test set performance. We have implemented an inference step for NAM and SPAM in order to make a direct comparison with FocusLearn on the test sets.



(a) FocusLearn



(b) LSTM



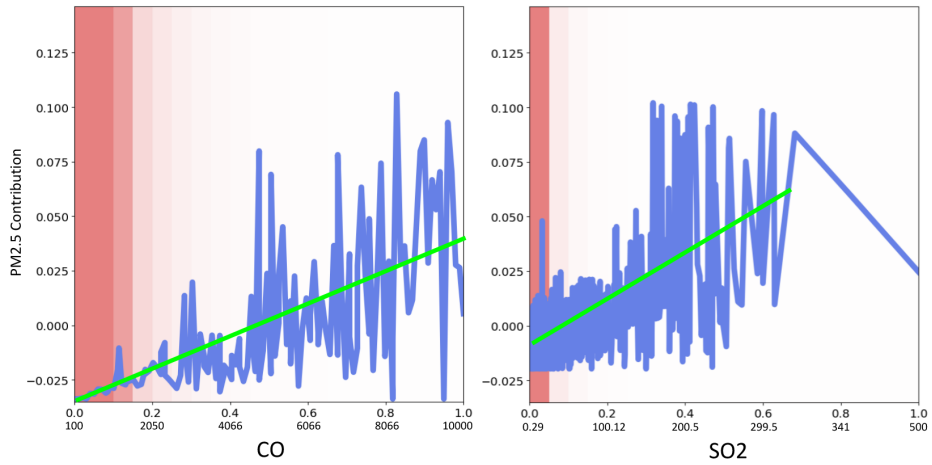
(c) XGBoost

Figure 2: Prediction visualisations. The x-axis represents predicted time-steps and the y-axis represents feature values. *Note: data are pre-processed differently for XGBoost, which supports missing values by default, whereas missing values are interpolated for training with FocusLearn and LSTM.*

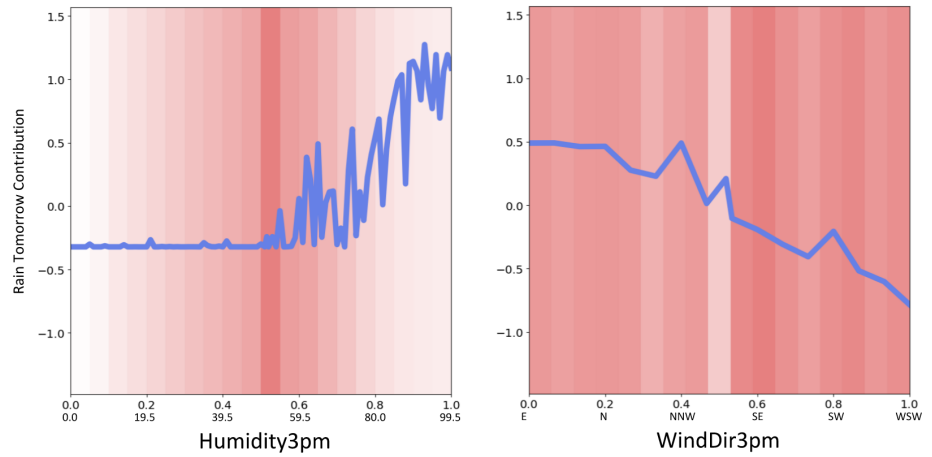
NAM generally performs worse in a classification task than in a regression task, achieving only 1.2% and 24.4% F1 score in the *EEG* and *Weather* data sets, respectively, as shown in Table 1. By incorporating AFS in its architecture, FocusLearn overcomes NAM's limitation and produces the best results of all approaches on *EEG*, while matching the performance of XGBoost with the best results on *Weather*. This is discussed further in Section 5.2.

Explainability: FocusLearn is an interpretable model, it produces explanatory plots showing the influence of selected features on model prediction. A selection of such plots is shown in Fig. 3, with further analyses of the explanations provided for each data set in Appendix C of the extended version of this paper [43]. As with NAM, the interpretability of FocusLearn comes from visualising the shape of the plots for each modular network. Since each selected feature is learned independently, the shape functions show the exact contribution of a module (*i.e.,* feature) to a prediction. For example, it can be seen that there is a positive correlation between the quantity of carbon monoxide *CO* and the predicted quantity of particulate matter $PM_{2.5}$.

Notice how in Fig. 3b, the contribution of *SO2* decreases linearly when the value of *SO2* exceeds 320 $\mu g/mg^3$. This should warrant further investigation as the plot shows that there is insufficient data (with a lighter shade of red) when *SO2* values exceed 100 $\mu g/m^3$. With these plots, the effect of *CO* and *SO2* on $PM_{2.5}$ can be further investigated, hopefully to produce a better understanding of the model and drive model intervention and improvement.



(a) ***Air* data set**. Showing graphs (shape functions) learned by FocusLearn at predicting future $PM_{2.5}$ value given *Carbon Monoxide* and *Sulfur Dioxide* concentration. The shape function plot for *CO* shows an overall positive correlation (highlighted by the superimposed green line) with large variation. Similarly for *SO2* there is a positive correlation until the sudden drop when *SO2* exceeds 320 $\mu g/mg^3$. Notice how simply sampling the input to obtain such correlations would not be informative or feasible in practice with an increased number of input variables.



(b) ***Weather* data set.** The graphs (shape functions) learned by FocusLearn predict whether it will rain the following day given *Humidity* and *Wind Direction*. The shape function plot for *Humidity3pm* shows that humidity from 60 to 99% at 3pm today will likely lead to *rain* tomorrow. The start of the upward trend of the contribution of *humidity* occurs in a region of high data density. The plot for *WindDir3pm* shows higher chances of rain when there is a northerly wind compared with lower chances of rain when the wind direction changes to a more southerly wind.

Figure 3: NAM-style explanations for selected features learned by MNN. Network outputs are shown on the y-axis. Feature values are shown on the x-axis (normalised values above and actual values below). The blue line represents the learned shape function. Normalised data densities are shown using the red bars, the darker the red, the more data is available in that region.

5 Additional Experiments

5.1 Effectiveness of ANB

In these additional experiments, we attempt to identify the impact of using an LSTM as the RNN and that of using ANB as proposed in Section 3.3. For the RNN, we consider GRU and BLSTM as other options, while as an alternative to ANB we consider a standard linear layer and a variation of the exponential linear unit with inputs shifted by a bias (ExU), as used by NAM in [2]. For these experiments, all models are trained with the same hyper-parameters with 128 batch size, 0.002 learning rate, 128 hidden RNN units, 64 and 32 hidden units for the second and third linear layers in each module (*i.e.,* Layers 2 and 3 in Fig. 1 ④, respectively), trained for 100 epochs with the Adam optimiser with early call-backs to avoid over-fitting.

Using the *Air* data set, Table 2 shows that the LSTM+ANB combination achieves the lowest loss. More importantly, any combination with ExU performed worse than its *Linear* and *ANB* counterparts. It can be seen, even before other data sets are considered, that ANB can improve the performance of FocusLearn, with LSTM+ANB, GRU+ANB and BLSTM+ANB being the best performers.

5.2 Effectiveness of AFS

As observed in Table 1, NAM tends to perform worse in the classification tasks than in the regression tasks. Therefore, to further validate the effectiveness of the proposed AFS, we examine the effect of changing the number of features to be selected by AFS. Using the *Weather* classification data set, we compared test set accuracy when n = 10 (reported in Table 1) with n = 21 (*i.e.,* including all available features). Optimal hyper-parameters already obtained for the *Weather* data set are used, with both networks trained for 100 epochs on the Adam optimiser. The results show that the test set accuracy when n = 10 is more than double the test set accuracy when all the features are used (n = 21).

6 Conclusion and Future Work

An interpretable modular neural network named FocusLearn that uses attention for feature learning was introduced and evaluated in this paper. It has two main innovations: Attention-based Feature Selection (AFS) and Attention-based Node Bootstrapping (ANB). In AFS, Multi-head Attention is modified to identify the most relevant features from the original input, while ANB acts as a bridge between AFS and each module of the interpretable network by weighting each selected feature based on its learned attention.

The goal of FocusLearn was to improve predictive performance compared to NAM and yet retain interpretability. Experiments have shown that FocusLearn not only outperforms NAM on all metrics (and its variant SPAM), but it can match the predictive performance of state-of-the-art non-interpretable models LSTM and XGBoost, outperforming the state-of-the-art in some cases.

Related work AFS, MFS and TabNet share some similarities with the proposed FocusLearn; however, FocusLearn is distinct from them because attention weights are learned from a recurrent network in FocusLearn and used for weighted feature selection. FocusLearn also has flexibility using any recurrent neural networks. FocusLearn was designed to be closely related to architectures with modularity, such as additive models. A common assumption of such modules such as GAM and NAM is feature independence. This has been proved to be too strong an assumption in certain areas of application such as time series, as indicated by the poor predictive performance of such models in the experiments reported in this paper. FocusLearn tackles this problem by using AFS for feature selection during training. Variations of NAM on the other hand, such as SPAM, seek to tackle the same problem by allowing interaction terms. This, however, may increase the complexity of the explanations provided for these models. As future work, we shall evaluate this and other

trade-offs, such as the number of modules to use, in practice, since it is likely that the answers to these questions will depend on the characteristics of the specific application domain and quantitative expert evaluation.

In comparison with SPAM, a given application may require the use of higher-order feature interactions, not available in FocusLearn. We shall investigate the possibility of the attention mechanism creating such features for implementation into a single network module. Finally, we shall continue to investigate the practical value of the explanations provided, how they may inform model intervention, and the interactions that may exist between attention-based explanations of RNNs and interpretable modular neural networks.

Acknowledgment

This work was supported by the European Commission's Horizon 2020 research and innovation program under the SMART BEAR project (Grant agreement number: 8557172/H2020-SC1-FA-DTS-2018-2.) The authors would like to express their gratitude to the anonymous reviewers for their very helpful comments and to the authors of [11] for giving us access to the *OtiReal* data set.

References

[1] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[2] R. Agarwal, L. Melnick, N. Frosst, X. Zhang, B. Lengerich, R. Caruana, and G.E. Hinton. Neural additive models: Interpretable machine learning with neural nets. *Advances in Neural Information Processing Systems*, 34, 2021.

[3] S.O. Arik and T. Pfister. Tabnet: Attentive interpretable tabular learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:6679–6687, 8 2021.

[4] J.L. Ba, J.R. Kiros, and G.E. Hinton. Layer normalization, 2016. arXiv:1607.06450.

[5] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5:157–166, 3 1994.

[6] Trenton Bricken, Adly Templeton, Joshua Batson, and et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

[7] T. Caelli, Ling G., and W. Wen. Modularity in neural computing. *Proceedings of the IEEE*, 87:1497–1518, 1999.

[8] L. Cao, Y. Chen, Z. Zhang, and N. Gui. A multiattention-based supervised feature selection method for multivariate time series. *Computational Intelligence and Neuroscience*, 2021:1–10, 7 2021.

[9] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014. arXiv: 1409.1259.

[10] K.S. Choi, S.H. Choi, and B. Jeong. Prediction of idh genotype in gliomas with dynamic susceptibility contrast perfusion mr imaging using an explainable recurrent neural network. *Neuro-Oncology*, 21:1197–1209, 9 2019.

[11] Jeppe H. Christensen, Gabrielle H. Saunders, Michael Porsbo, and Niels H. Pontoppidan. The everyday acoustic environment and its association with human heart rate: evidence from real-world data logging with hearing aids and wearables. *Royal Society Open Science*, 8, 2 2021. DOI: 10.1098/rsos.201345.

[12] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014. arXiv:1412.3555.

[13] Abhimanyu Dubey, Filip Radenovic, and Dhruv Mahajan. Scalable interpretability via polynomials. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 36748–36761. Curran Associates, Inc., 2022.

[14] J.L. Elman. Finding structure in time. *Cognitive science*, 14:179–211, 1990.

[15] Z. Fang, P. Wang, and W. Wang. Efficient learning interpretable shapelets for accurate time series classification. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 497–508. IEEE, 4 2018.

[16] W. Gao, M. Aamir, A.B. Shabri, R. Dewan, and A. Aslam. Forecasting crude oil price using kalman filter based on the reconstruction of modes of decomposition ensemble model. *IEEE Access*, 7:149908–149925, 2019.

[17] Wendong Ge, Jin-Won Huh, Yu Rang Park, Jae-Ho Lee, Young-Hak Kim, and Alexander Turchin. An interpretable icu mortality prediction model based on logistic regression and recurrent neural networks with lstm units. In *AMIA Annual Symposium Proceedings*, volume 2018, page 460. American Medical Informatics Association, 2018.

[18] A. Ghorbani, A. Abid, and J. Zou. Interpretation of neural networks is fragile. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:3681–3688, 7 2019.

[19] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256. {PMLR},, 5 2010.

[20] N. Gui, D. Ge, and Z. Hu. Afs: An attention-based mechanism for supervised feature selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:3705–3713, 7 2019.

[21] T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, 1, 8 1986.

[22] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–1780, 1997.

[23] Y. Huang, C-H. Chen, and C-J. Huang. Motor fault detection and feature extraction using rnn-based variational autoencoder. *IEEE Access*, 7, 2019.

[24] Wonkeun Jo and Dongil Kim. Neural additive time-series models: Explainable deep learning for multivariate time-series prediction. *Expert Systems with Applications*, 228:120307, 2023.

[25] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

[26] S. Kaushik, A. Choudhury, P.K. Sheron, N. Dasgupta, S. Natarajan, L.A. Pickett, and V. Dutt. Ai in healthcare: Time-series forecasting using statistical, neural, and ensemble architectures. *Frontiers in Big Data*, 3:4, 2020.

[27] Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-Tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. A system for massively parallel hyperparameter tuning. *Proceedings of Machine Learning and Systems*, 2:230–246, 2020.

[28] B Lim, S.O Arık, N Loeff, and T Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37:1748–1764, 10 2021.

[29] Z.C. Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16:31–57, 2018.

[30] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han. On the variance of the adaptive learning rate and beyond, 2019. arXiv:1908.03265v4.

[31] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2016. arXiv:1608.03983.

[32] Y. Lou, R. Caruana, J. Gehrke, and G. Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–631. ACM, 8 2013.

[33] S.M. Lundberg and S-I Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

[34] Kwun Ho Ngan, James Phelan, Esma Mansouri-Benssassi, Joe Townsend, and Artur S. d'Avila Garcez. Closing the neural-symbolic cycle: Knowledge extraction, user intervention and distillation from convolutional neural networks. In Artur S. d'Avila Garcez, Tarek R. Besold, Marco Gori, and Ernesto Jiménez-Ruiz, editors, *Proc. NeSy2023, Siena, Italy, July 3-5, 2023*, volume 3432 of *CEUR Workshop Proceedings*, pages 19–43. CEUR-WS.org, 2023.

[35] R.P. Paiva and A. Dourado. Interpretability and learning in neuro-fuzzy systems. *Fuzzy Sets and Systems*, 147:17–38, 10 2004.

[36] Filip Radenovic, Abhimanyu Dubey, and Dhruv Mahajan. Neural basis models for interpretability. *Advances in Neural Information Processing Systems*, 35:8414–8426, 2022.

[37] M.T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *In Proc. 22nd ACM SIGKDD*, pages 1135–1144, 2016.

[38] Oliver Roesler. Eeg eye state, 07/2023.

[39] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206–215, 2019.

[40] H. Sak, A.W. Senior, and F. Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2014.

[41] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45:2673–2681, 1997.

[42] P. Senin and S. Malinchik. Sax-vsm: Interpretable time series classification using sax and vector space model. In *2013 IEEE 13th International Conference on Data Mining*, pages 1175–1180. IEEE, 12 2013.

[43] Qiqi Su, Christos Kloukinas, and Artur S. d'Avila Garcez. Focuslearn: Fully-interpretable, high-performance modular neural networks for time series, 2023. Extended version. arXiv: 2311.16834.

[44] M.A.I. Sunny, M.M.S. Maswood, and A.G. Alharbi. Deep learning-based stock price prediction using lstm and bi-directional lstm model. In *2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES)*, pages 87–92, 2020.

[45] M. Tsang, H. Liu, S. Purushotham, P. Murali, and Y. Liu. Neural interaction transparency (nit): Disentangling learned interactions for improved interpretability. *Advances in Neural Information Processing Systems*, 31, 2018.

[46] S. Varela-Santos and P. Melin. A new modular neural network approach with fuzzy response integration for lung disease classification based on multiple objective feature optimization in chest x-ray images. *Expert Systems with Applications*, 168:114361, 4 2021.

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[48] Julius Vetter, Kathleen Lim, Tjeerd M. H. Dijkstra, Peter A. Dargaville, Oliver Kohlbacher, Jakob H. Macke, and Christian F. Poets. Neonatal apnea and hypopnea prediction in infants with robin sequence with neural additive models for time series. *medRxiv*, 2023.

[49] Benedikt Wagner and Artur d'Avila Garcez. Neural-Symbolic Integration for Fairness in AI. In *AAAI Spring Symposium AAAI-MAKE*, 2021.

[50] Z. Wang, W. Yan, and T. Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1585. IEEE, 5 2017.

[51] Adam White and Artur S. d'Avila Garcez. Measurable counterfactual local explanations for any classifier. In *ECAI 2020*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2529–2535. IOS Press, 2020.

[52] Adam White, Kwun Ho Ngan, James Phelan, Kevin Ryan, Saman Sadeghi Afgeh, Constantino Carlos Reyes-Aldasoro, and Artur S. d'Avila Garcez. Contrastive counterfactual visual explanations with overdetermination. *Mach. Learn.*, 112(9):3497–3525, 2023.

[53] L. Ye and E. Keogh. Time series shapelets. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*, page 947. ACM Press, 2009.

[54] H. Zhang, Y. Zou, X. Yang, and H. Yang. A temporal fusion transformer for short-term freeway traffic speed multistep prediction. *Neurocomputing*, 500:329–340, 8 2022.

[55] P.G. Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 1 2003.

[56] S. Zhang, B. Guo, A. Dong, J. He, Z. Xu, and S.X. Chen. Cautionary tales on air-quality improvement in beijing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473:20170457, 9 2017.

[57] B. Zhou, G. Yang, Z. Shi, and S. Ma. Interpretable temporal attention network for covid-19 forecasting. *Applied Soft Computing*, 120:108691, 5 2022.

Table 1: Average test set results of FocusLearn compared with other methods on four data sets for time series regression and classification. For regression tasks, lower results are better (↓). For classification tasks, higher results are better (↑).

Regression ↓					
		SMAPE	MASE	WAPE	Speed (s)
Air	**FocusLearn**	$\mathbf{0.2640}^{**}$	$\mathbf{0.1654}^{**}$	$\mathbf{0.1637}^{**}$	1,198.10
	NAM	0.5221^{*}	0.4166^{*}	0.4206^{*}	3,598.02
	SPAM	0.5231	0.2765	0.2778	1,571.26
	XGBoost(50)	0.2704^{**}	0.1704^{**}	0.1706^{**}	15.42
	LSTM	0.2935^{*}	0.1774^{**}	0.1772^{**}	1,289.57
OtiReal	**FocusLearn**	0.2105^{*}	1.4829	0.0806^{**}	96.32
	NAM	0.8543^{*}	12.3317	0.6661	238.95
	SPAM	0.2951	2.3075	0.1249^{*}	119.73
	XGBoost(100)	0.9574^{**}	$\mathbf{1.0840}^{**}$	0.6846^{**}	1.29
	LSTM	$\mathbf{0.1809}^{*}$	1.2694	$\mathbf{0.0666}^{*}$	92.77
Classification ↑					
		Accuracy	F1	AUC	Speed (s)
EEG	**FocusLearn**	$\mathbf{0.8203}^{*}$	$\mathbf{0.8131}^{*}$	$\mathbf{0.8203}^{*}$	24.28
	NAM	0.5430^{**}	0.0115^{**}	0.5020^{**}	136.56
	SPAM	0.5000^{*}	0.0159^{*}	0.5002^{*}	115.61
	XGBoost(5)	0.6959^{**}	0.6169^{*}	0.6824^{**}	0.02
	LSTM	0.5000^{**}	0.2667	0.5^{**}	53.53
Weather	**FocusLearn**	$\mathbf{0.8518}^{**}$	0.5157^{*}	0.6791^{**}	1,192
	NAM	0.8169^{**}	0.2442	0.5687^{*}	1,552.91
	SPAM	0.8369^{**}	0.4305	0.6444^{*}	1,231.40
	XGBoost(5)	0.8473^{*}	$\mathbf{0.5508}^{*}$	$\mathbf{0.7009}^{*}$	20.67
	LSTM	0.8419^{**}	0.4592^{*}	0.6509^{**}	664.46

$^{(*)}$ Variance across experiments is < 0.05.
$^{(**)}$ Variance across experiments is < 0.01.
FocusLearn, NAM, SPAM, XGBoost(5) are interpretable. XGBoost(50), XGBoost(100), LSTM are non-interpretable. The number in parentheses after XGBoost is the optimal number of trees; XGBoost is considered to be non-interpretable when the number of trees exceeds 10.
Note 1: Means and standard deviations are reported from 5 random trials with optimal hyper-parameters for each task (see Appendix A of [43]).
Note 2: Data set details are provided in Appendix B of [43]. Air [56] predicts future particulate matter values ($PM_{2.5}$), OtiReal [11] is used to predict future hearing aid usage, EEG [38] predicts eyes-open or closed states, and Weather predicts if it will rain the next day.

Table 2: Test set loss comparison between different RNN networks and Weighted Linear Layers in the *Air* data set.

Layer 1	RNN	Test Set Loss (Air Data set) ↓
ANB	LSTM	**0.0070**
	GRU	0.0072
	BLSTM	0.0216
Linears	LSTM	0.0216
	GRU	0.0251
	BLSTM	0.0338
ExU	LSTM	0.0513
	GRU	0.0591
	BLSTM	0.1334

.1 Optimal Hyper-parameters

The optimal hyper-parameters for each model for each dataset are listed below. Hyper-parameters are hyper-tuned using Ray-tune, specifically the Asynchronous Successive Halving Algorithm scheduler [27] with the Optuna search algorithm[6] for better parallelism, on the validation set.

Table 3: Optimal Hyper-Parameters

	Hyper-parameters	OtiReal	Air	EGG	Weather
FocusLearn	Initial Learning Rate	0.0130	0.0008	0.0021	0.0003
	Dropout rate in the RNN	0.0318	0.0349	$8.6209e^5$	0.1992
	Dropout rate in the MNN	0.1450	0.0142	0.0005	0.8378
	Dropout rate in the final output	0.0339	0.0513	0.0011	0.1851
	Batch Size	256	64	512	64
	Number of hidden unit in the RNN	256	64	64	1,026
	Number of hidden units in the MNN	[256, 128]	[256, 128]	[32, 16]	[512, 256]
NAM	Learning Rate	0.0065	0.0122	0.0007	0.0002
	Dropout Rate	0.4875	0.0405	0.0107	0.0218
	Batch Size	64	512	256	256
	Hidden Sizes	[128, 64]	[512, 256]	[64, 32]	[256, 128]
SPAM	Learning Rate	0.268	0.1902	0.0009	
	Dropout Rate	0.1072	0.0265	0.0590	
	Batch Size	64	256	64	
LSTM	Learning Rate	0.0260	0.0082	0.7330	0.0369
	Dropout rate	0.1153	0.0567	0.2340	0.1285
	Batch Size	256	64	128	64
	Number of hidden unit	64	128	128	256
XGBoost	Number of estimators	1,000	200	50	1,000
	Max Depths	100	50	5	5
	Subsample	0.2605	0.8069	0.3804	0.3191
	Learning Rate	0.3370	0.7600	$2.8467e^5$	0.4725

.2 Datasets Description

Table 4: Datasets Description

Name	OtiReal	Air	EEG	Weather
Source	[11]	[56]	[38]	http://www.bom.gov.au/climate/data/
Instances	24,736	420,768	14,892	137,910
Features	16	16	14	21
Class Distribution	-	-	1: 0.82	1: 0.26
Feature Type	Mixed	Mixed	Continuous	Mixed
Top Features	10	10	10	10

Real-world longitudinal and observational hearing aid (HAid) data are collected from patients signed up for the HearingFitness™ feature with the Oticon ON™ remote control app on the Oticon Open Hearing Aids (Oticon A/S, Smørum, Denmark). Christensen *et al.* [11] extracted a sampled data (*OtiReal*) of 98 users between June-December 2019 that contains no personal information characterising the patients to preserve privacy. *OtiReal* contains sound data concerning the acoustic

[6] `https://www.ray.io/`, `https://optuna.org/`

environment detected by the HAids and logged by the connected smart-phones. These sound data include Sound Pressure Level (SPL), Modulation Index (MI), Signal-to-Noise Ratio (SNR), and Noise Floor (NF), all measured in a broadband frequency range of 0-10kHz in decibel units. HAid usage is calculated from the timestamps when the sound data are collected. The task is then to predict future daily HAid usage. In the Beijing Multi-Site Air-Quality (*Air*) data set [56], 7 years of meteorological data along with 4 years of $PM_{2.5}$ data of Beijing at 36 monitoring sites were collected. $PM_{2.5}$ refers to the fine particulate matter (PM) concentration in the air with aerodynamic diameter of less than 2.5 μm. The tasks is to predict future $PM_{2.5}$ values.

In the EEG Eye State (*EEG*) data set, continuous electroencephalogram (EEG) measurement are collected from the Emotive EEG Neuroheadset [38]. The duration of the EEG measurement is 119 seconds, and eye state (*e.g.,* whether the eyes are closed or open) are detected by a camera during EEG measurement and added to the dataset manually later, where 1 indicates eye-closed and 0 indicates eye-open. The task is to classify eye state based on the EEG measurement. Another binary classification dataset is the Rain in Australia (*Weather*) dataset, which contains 10 years of daily weather observations drawn from a number of weather stations in Australia. The task is to predict whether or not it will rain tomorrow.

.3 Model Explanations

.3.1 Explanations for OtiReal Dataset

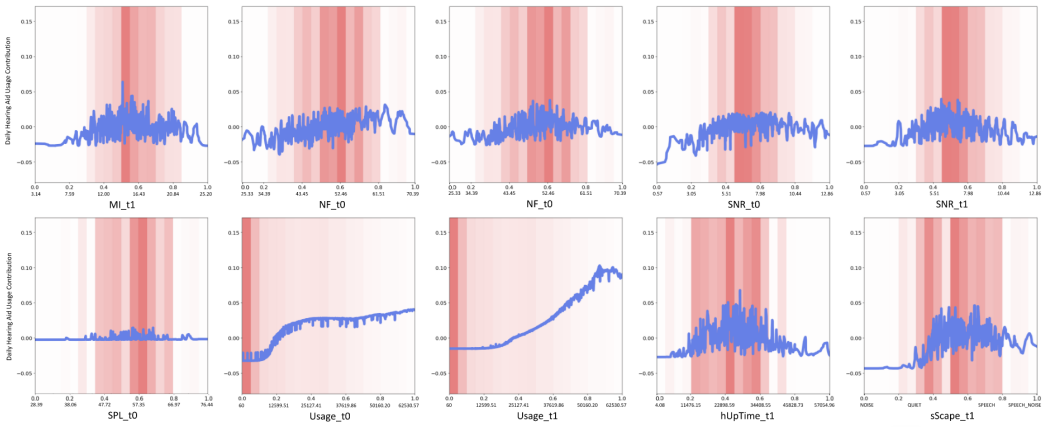


Figure 4: **OitReal** Graphs learned by FocusLearn in predicting future hearing aid usage (regression) on the *OtiReal* dataset. These plots show top 10 features selected by the AFS component in the FocusLearn, where selected features with normalised and original values are on the x-axis, and daily future hearing aid usage prediction contribution are on the y-axis. In OtiReal, two timesteps of data are transformed and used as inputs, such that we are using data at t_i and t_{i+1} to predict hearing aid usage at t_{i+2}.

The plots for *MI*, *NF*, and *SNR* show that patients tend to use their HAids more in the future if the median values of these variables are sensed by their HAids. The lack of contribution of *SPL* warrants further investigation with the audiologists as *MI* and *SNR* are derived from *SPL*. The *Soundscape* (sScape) classifies momentary sound environment into four categories by a proprietary HAid algorithm using *MI*, *SNR*, and *SPL* values. Its plot shows that patients tend to use their HAids more if the soundscape is at the speech setting, whereas patients tend to use their HAids less with the noise or quiet setting. This could be an interesting observation to the audiologists as this might show that those patients who actively use their HAids to assist their hearing loss will tend to continuously use their HAids more in the future. The plot for *Hearing Aid Up Time* (hUpTime) shows patients will be most likely to use their HAids in the future if their HAids have been activated for more than

23,000 seconds (or 6.3 hours). The plots for *Usage* show that patients with higher daily HAid usage are more likely to have higher future daily HAid usage, and patients with lower daily HAid usage are more likely to have lower future daily HAid usage. These plots also show that the maximum contribution to future HAid usage of $Usage_{t0}$ and $Usage_{t1}$ is 0.05 and 0.1, respectively. This means that patients are more likely to have an even higher future daily HAid usage if they use their HAids for a sufficiently long period in a day continuously (for at least 2 days in this case).

.3.2 Explanations for Air Dataset

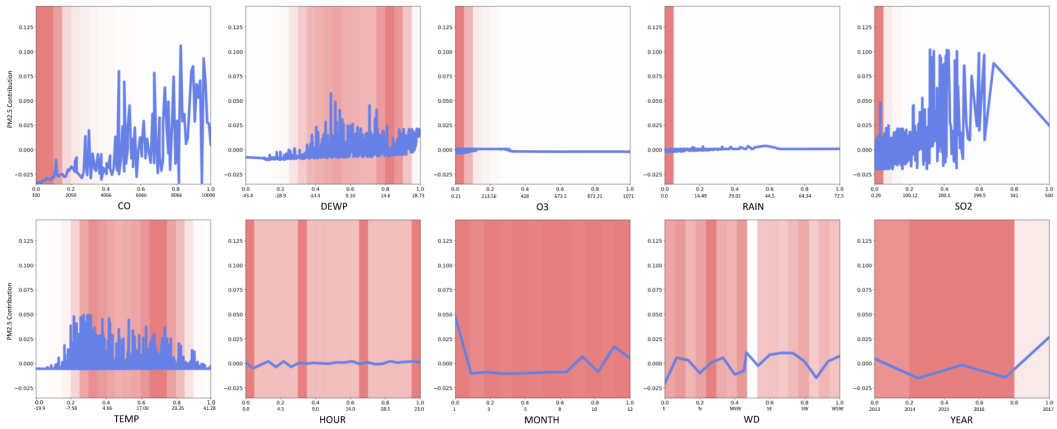


Figure 5: **Air** Graphs learned by FocusLearn in predicting future $PM_{2.5}$ values (regression) on the *Air* dataset. These plots show top 10 features selected by the AFS component in the FocusLearn, where selected features with normalised and original values are on the x-axis, and future $PM_{2.5}$ values prediction contribution are on the y-axis.

The plot for *Dew Point Temperature* (DEWP) shows that when *DEWP* is at around 0 degree Celsius, this would lead to highest predicted future $PM_{2.5}$ values. *O3 Concentration* (O3), *Precipitation* (RAIN), *HOUR* contribute almost nothing towards predicting future $PM_{2.5}$ values, and the contribution of *O3* even becomes negative with high *O3* values. The plot for *Temperature* (TEMP) shows that when temperature is in the range of -8 and 28 degrees Celsius, future $PM_{2.5}$ values are predicted to be the highest, whereas when the temperature is above 28 and below -8 degree Celsius, this will lead to a low future $PM_{2.5}$ values. This observation is in line with many literature studying correlation between $PM_{2.5}$ and season such that there is a negative correlation in summer and autumn and positive correlation in spring and winter. The plot for *MONTH* confirms this observation, such that future $PM_{2.5}$ values are predicted to be the highest in January, followed by November and September. The plot for *Wind Direction* (WD) shows that future $PM_{2.5}$ values are predicted to be the lowest when there is a Eastern wind, and future $PM_{2.5}$ values are predicted to be the highest when the wind changes from a northerly to southerly wind. Lastly, the plot for *YEAR* shows that more recent data (*i.e.,* data from 2016 onward) have more impact on the model in predicting higher future $PM_{2.5}$ values.

.3.3 Explanations for EEG Dataset

Fig. 6 shows that *AF4*, *F4*, *F7*, *FC5*, *O1*, *O2*, and *T8* have almost no contribution towards FocusLearn in predicting eye state from the EEG, and *AF3* and *P7* have minimal contribution towards the model prediction. This means that electrodes placed at these regions do not contribute much to FocusLearn in predicting the eye state. On the other hand, the electrode placed on the *T7* region is the most important one in predicting the eye state. Its plot shows that higher *T7* values means it is more likely

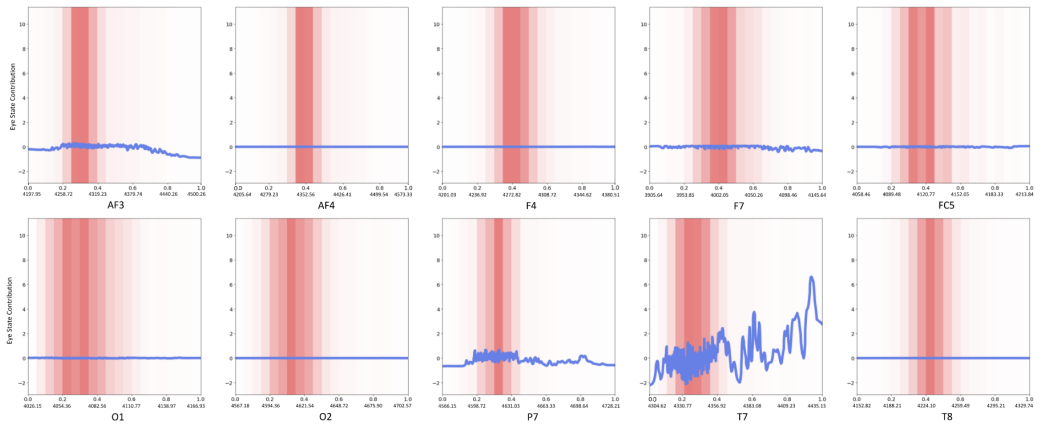


Figure 6: **EEG** Graphs learned by FocusLearn in predicting eye state (classification) on the *EEG* dataset. These plots show top 10 features selected by the AFS component in the FocusLearn, where selected features with normalised and original values are on the x-axis, and contribution towards the prediction are on the y-axis.

that the eyes are open (*i.e.,* label 1), and lower *T7* values means it is more likely that the eyes are closed (*i.e.,* label 0). Interestingly, *T7* is a feature that is neither the most correlated to the outcome nor the one with the highest multicollinearity. Meaning that FocusLearn managed to capture the non-linear relationship in the underlying data.

.3.4 Explanations for Weather Dataset

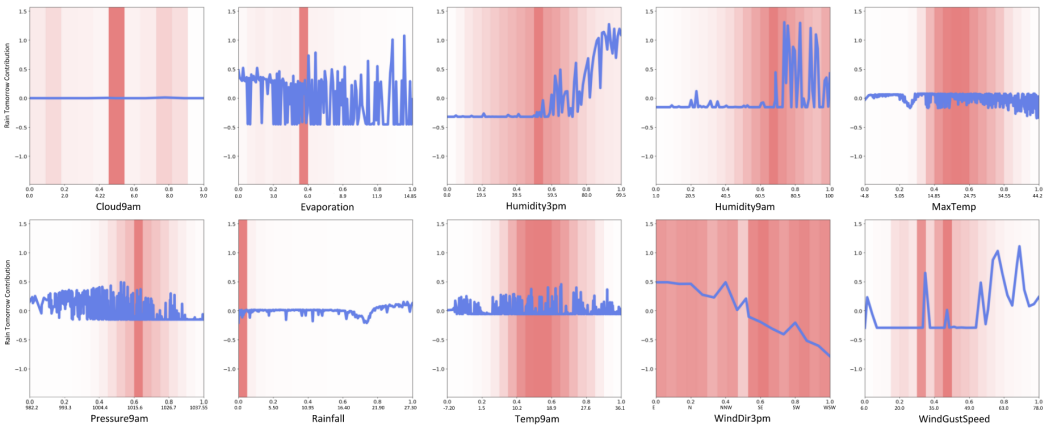


Figure 7: **Weather** Graphs learned by FocusLearn in predicting whether it will rain tomorrow (classification) on the *Weather* dataset. These plots show top 10 features selected by the AFS component in the FocusLearn, where selected features with normalised and original values are on the x-axis, and contribution towards the prediction are on the y-axis.

The plot for *Cloud9am* shows that this feature contributed nothing in FocusLearn predicting whether or not it will rain tomorrow. The relationship between evaporation and rain fall is rather complicated as it depends external factors including global climate phenomena, nevertheless, the shape plot for *Evaporation* provides a detailed analysis such that when *Evaporation* reaches 13mm, it is most likely lead to rain tomorrow. This complicated relationship can also be observed with *Pressure9am* and *Temp9am*, and how atmospheric pressure and temperature affects the chance of raining the next day

also depends on other factors. The plot for *Humidity9am* shows a drastic increase in the chance of raining tomorrow when the humidity is above 60%. The plot for *Maximum Temperature* (MaxTemp) shows that higher maximum temperature means there is a less chance of raining tomorrow. The plot for *Rainfall* shows that, as expected, the more rainfall detected on the recorded day, more likely to rain tomorrow. There is a drop in chance of raining tomorrow when rainfall detected on the recorded day is above 17mm, and this might warrant a further investigation with the meteorologist. The plot for *WindGustSpeed* shows that when the speed of the strongest wind is at around 70km/h, it is more likely to rain tomorrow.