Adaptive Online Bagging using the Cascade Approach[*]

Sergiy Popov[1*†], Iryna Pliss[1†], Oleh Zolotukhin[1†] and Maryna Kudryavtseva[1†]

[1] *Kharkiv National University of Radio Electronics, 14 Nauky av., Kharkiv, 61166, Ukraine*

Abstract

This paper introduces a novel adaptive cascade bagging system designed for real-time processing of complex, dynamic signals. Leveraging ensemble learning and a cascade architecture, the system dynamically adjusts model weighting and member count to optimize performance in non-stationary environments. Simulation results demonstrate a progressive reduction in forecasting errors on each cascade, with the 4th submetamodel surpassing the best individual ensemble member and the 6th achieving a 1.23-fold error reduction. This proves the proposed approach effectiveness and computational efficiency.

Keywords

cascade architecture, adaptive bagging, online learning, time series forecasting

1. Introduction

The field of Data Mining has witnessed a dramatic shift in recent years, with a growing reliance on sophisticated computational intelligence techniques to tackle complex problems. Traditional statistical methods, while still valuable, often struggle to effectively handle the sheer volume and complexity of modern datasets. Consequently, artificial neural networks (ANNs), in both their deep and more traditional shallow forms, alongside neuro-fuzzy systems, neo-fuzzy systems, wavelet-neuro-fuzzy networks, and other hybrid computational intelligence systems, have become increasingly prevalent tools for a wide range of Data Mining tasks. These systems are particularly effective in classification problems, pattern recognition, extrapolation, regression analysis, diagnostics, modeling, and more.

The core appeal of these computational intelligence approaches lies in their universal approximation properties and the ability to adjust their internal parameters, and in some cases even their architecture, through a process of learning from training data. This adaptability allows them to tailor themselves to the specific characteristics of the problem at hand, leading to potentially superior performance. The training process involves feeding the system labeled data examples, allowing it to refine its internal workings to map inputs to desired outputs.

However, the selection of the right computational intelligence system for a particular Data Mining task is far from straightforward. While multiple systems may be capable of solving the same problem, determining which one will deliver the best results is often impossible *a priori*. Each system possesses its own strengths and weaknesses, making the choice a complex trade-off. For instance, deep neural networks (DNNs), the current favorites of many AI applications, are known for their potential to achieve extremely high accuracy. However, this performance comes at a significant cost. DNNs typically require massive amounts of training data – often tens of thousands or even millions of labeled examples – and can demand substantial computational resources and time for training. The training process can be iterative, requiring multiple passes through the data and careful tuning of hyperparameters. In contrast, Radial Basis Function Neural Networks (RBFNs) offer a considerably

[*] *ProfIT AI'25: 5th International Workshop of IT-professionals on Artificial Intelligence, October 15–17, 2025, Liverpool, UK*
[1*] Corresponding author.
[†] These authors contributed equally.

serhii.popov@nure.ua (S. Popov); iryna.pliss@nure.ua (I. Pliss); oleg.zolotukhin@nure.ua (O. Zolotukhin); maryna.kudryavtseva@nure.ua (M. Kudryavtseva)
0000-0002-1274-5830 (S. Popov); 0000-0001-7918-7362 (I. Pliss); 0000-0002-0152-7600 (O. Zolotukhin); 0000-0003-0524-5528 (M. Kudryavtseva)

faster learning process. This makes them attractive for applications where rapid deployment or real-time performance is crucial. However, RBFNs are susceptible to the “curse of dimensionality”, which arises when dealing with high-dimensional datasets. As the dimensionality increases, the performance of RBFNs degrades significantly, requiring exponentially more neurons to maintain accuracy.

Neo-fuzzy systems, another option, are known for their high learning speed and ability to incorporate human expertise through fuzzy logic principles. However, they don’t always guarantee the necessary approximation properties to accurately model complex relationships within the data. They might be fast to train, but the resulting model might not capture the underlying patterns effectively.

The challenge, therefore, isn't simply about applying these powerful tools; it's about understanding their nuances and selecting the most appropriate system – or even a combination of systems – for the specific problem, dataset, and desired performance characteristics. This often involves experimentation, careful evaluation of results, and a deep understanding of the strengths and limitations of each approach. The optimal solution frequently emerges through iterative refinement and a willingness to explore different architectural choices and training methodologies.

When faced with complex challenges where individual systems exhibit varying strengths and weaknesses, the use of ensemble approaches can prove remarkably effective [1-8]. The core principle behind ensemble methods is to harness the collective intelligence of multiple models to achieve a superior outcome compared to any single model acting alone.

Historically, the vast majority of ensemble methods have relied on a batch (offline) approach. This involves providing the entire training dataset upfront and repeatedly analyzing it to train and refine the individual models and the ensemble. While effective, this approach can be computationally expensive and less adaptable to dynamic data environments. However, there are several online ensemble methods [9-11] specifically designed to address Data Stream Mining problems, where data arrives sequentially and potentially at a very high rate.

Within the broader landscape of ensemble approaches, bagging procedures [12-14] have emerged as particularly powerful techniques. Bagging, short for “bootstrap aggregation,” involves training multiple individual models on different subsets of the training data. The results generated by each of these models are then fed into a metamodel, also sometimes referred to as a combiner or aggregator. This metamodel acts as a sophisticated decision-making engine, intelligently combining the output signals from all ensemble members to synthesize the final, optimal solution.

A common challenge in implementing bagging is determining the optimal number of ensemble members. A small number of members might not provide sufficient diversity to capture the full complexity of the problem, leading to limited accuracy gains. Conversely, an excessively large number can significantly complicate the training process of the metamodel, increasing computational cost and potentially leading to overfitting. A promising avenue of research addresses this challenge through evolutionary approaches [13-15]. These methods dynamically adjust the number of ensemble members during the metamodel learning process. This means the number of inputs to the metamodel is constantly changing, introducing a layer of complexity to both the metamodel itself and its learning process.

To simplify and accelerate the bagging process, we apply a cascade approach. Instead of relying on a single, complex metamodel, a cascade approach employs a series of relatively simple metamodels arranged in sequence. This significantly simplifies the synthesis of the metamodel and the subsequent tuning process.

2. Cascade bagging system architecture

Fig. 1 shows the cascade bagging system architecture consisting of a set of two-input bagging submetamodels.

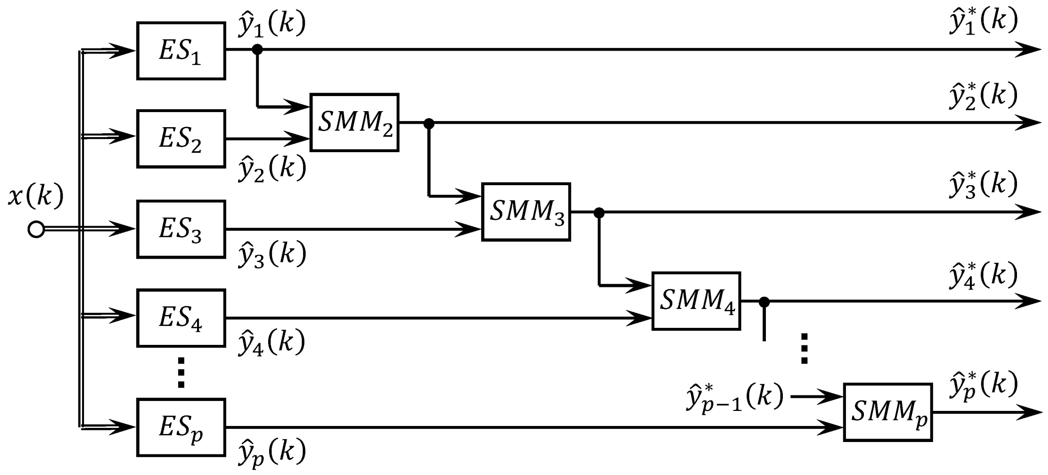

Figure 1: Architecture of the cascade bagging system

The ensemble consists of p ensemble subsystems ES_1, ES_2, $\dots$, $ES_r, \dots, ES_p$ from the simplest to the most complex. Thus, elementary Rosenblatt perceptrons, adalines, neo-fuzzy neurons, etc. can be used as ES_1, and sufficiently complex deep neural networks as ES_p. The inputs of these subsystems receive the same vector signal $x(k)=\left(x_1(k),\dots,x_i(k),\dots,x_n(k)\right)^T$ (here k is the current discrete time). Scalar output signals $\hat{y}_1(k),\dots,\hat{y}_r(k),\dots,\hat{y}_p(k)$ are calculated at their outputs. If the output signal $\hat{y}_1(k)$ satisfies the *a priori* specified accuracy requirements, the bagging procedure is not required and the output of the system as a whole is the output signal $\hat{y}^*_1(k)=\hat{y}_1(k)$. Otherwise, the signal $\hat{y}_1(k)$ is fed to the first input of the bagging submetamodel SMM_2, the second input of which is fed with the output signal of the second subsystems S_2 – $\hat{y}_2(k)$. The SMM_2 output signal is formed as follows

$$\hat{y}^*_2(k)=c_2\hat{y}_2(k)+(1-c_2)\hat{y}^*_1(k),$$

where c_2 is the single tuned parameter of the submetamodel SMM_2. Signal $\hat{y}^*_2(k)$ should be better in terms of accuracy than $\hat{y}^*_1(k)$ and $\hat{y}_2(k)$.

Then, signal $\hat{y}^*_2(k)$ is fed to the submetamodel SMM_3, whose other input receives $\hat{y}_3(k)$. SMM_3 produces the following result

$$\hat{y}^*_3(k)=c_3\hat{y}_3(k)+(1-c_3)\hat{y}^*_2(k),$$

here $\hat{y}^*_3(k)$ should be better in terms of accuracy than $\hat{y}^*_2(k)$ and $\hat{y}_3(k)$.

And finally, the last submetamodel SMM_p produces the following output signal

$$\hat{y}^*_p(k)=c_p\hat{y}_p(k)+(1-c_p)\hat{y}^*_{p-1}(k),$$

which should be better in terms of accuracy than output signals of all previous submetamodels.

In this setup, if the signal of any previous submetamodel SMM_r satisfies all the accuracy requirements, then the process of building up submetamodels can be stopped and only r members of the ensemble will be activated in the system.

The advantage of this approach is the simplicity of its implementation, since in each submetamodel only one parameter c_r is being tuned, which can be calculated in online real-time mode, while the ensemble itself contains only the required number of members – ensemble subsystems. Also note that in non-stationary situations the number of activated submetamodels can both decrease and increase during the operation depending on the required solution accuracy.

3. Submetamodels online learning

The process of training submetamodels consists in adjusting the parameters $c_2,\ c_3,\dots,c_p$ in each of the system cascades.

Let us write the output signal of the p-th cascade in the form

$$\hat{y}_p^*(k)=c_p\hat{y}_p(k)+(1-c_p)\hat{y}_{p-1}^*(k)=c_p\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)+\hat{y}_{p-1}^*(k),$$

and introduce the error signal

$$e_p(k)=y(k)-\hat{y}_p^*(k)=y(k)-c_p\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)-\hat{y}_{p-1}^*(k)=e_{p-1}(k)-c_p\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right),$$

where $y(k)$ is a reference signal.

The squared error has the form

$$e_p^2(k)=e_{p-1}^2(k)-2e_{p-1}(k)c_p\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)+c_p^2\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)^2.$$

And after summing up over the training set

$$\sum_k e_p^2(k)=\sum_k e_{p-1}^2(k)-2c_p\sum_k e_{p-1}(k)\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)+c_p^2\sum_k\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)^2.$$

Then we solve a differential equation

$$\frac{\partial\sum_k e_p^2(k)}{\partial c_p}=-2\sum_k e_{p-1}(k)\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)+2c_p\sum_k\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)^2=0,$$

and get a rather simple relation

$$c_p(k)=\frac{\sum_k e_{p-1}(k)\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)}{\sum_k\left(\hat{y}_p(k)-\hat{y}_{p-1}^*(k)\right)^2},$$

which in a single-step form can be written as

$$c_p(k)=\frac{e_{p-1}(k)}{\hat{y}_p(k)-\hat{y}_{p-1}^*(k)}.$$

Also note that when processing nonstationary signals disturbed by noise, it is appropriate to organize the process of the parameters $c_2,\ c_3,\dots,c_p$ tuning over a sliding window. This would provide a trade-off between the following and filtering properties of the bagging procedure.

4. Simulation results

To validate the effectiveness and practicality of the proposed adaptive cascade bagging system, we applied it to a challenging real-world problem: short-term electric load forecasting (STLF).

Specifically, our test case focuses on 1-step ahead forecasting of the daily electric load for a regional power system in Ukraine. This application presents a particularly demanding scenario due to the inherent complexities and non-stationarities commonly found in electric load data.

The dataset utilized for this simulation comprises an original time series containing $N=337$ samples of daily electric load data used as a reference signal represented in Figure 2. It exhibits a complex pattern characterized by several discernible trends corresponding to different seasons. The data also reveals periodic components, predominantly weekly fluctuations reflecting variations in energy consumption across the week. Furthermore, the series is punctuated by sudden changes and outliers, indicative of unexpected events impacting energy demand. A strong random component is also evident, which is a common characteristic of electric load data in large systems. This randomness arises from the multitude of external factors influencing energy consumption, many of which possess inherently random or chaotic behavior. Weather conditions, for instance, are a prime example of a factor significantly impacting energy demand and exhibiting complex fluctuations [16].

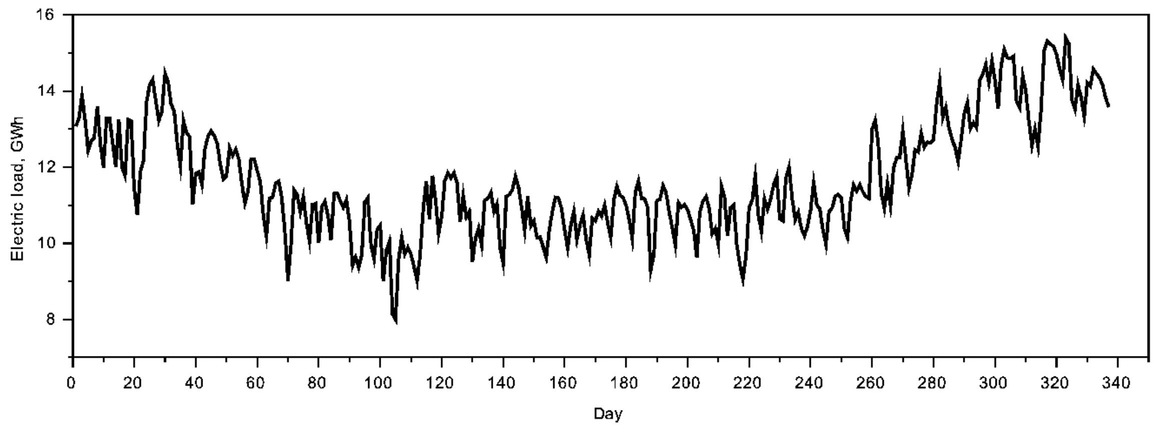

Figure 2: The original electric load time series

These trends, periodicities, sudden changes, outliers, and the significant random component bring non-stationarity and noisiness to the time series. Consequently, its forecasting presents a considerable challenge. In such scenarios, it is frequently observed that different forecasting models or methods demonstrate superior performance on particular segments of the series, while exhibiting inferior performance on others. It's rare for a single model or method to consistently outperform all others across the entire series. This is precisely the situation where bagging methods, and particularly our adaptive cascade bagging system, can prove useful. By combining the forecasts from multiple models, the bagging approach aims to extract the best predictions from each individual model, ultimately improving the overall forecasting accuracy and robustness.

We employed $q=6$ distinct and independent ensemble subsystems – computational intelligence models of various structures and complexity [17] – to produce six corresponding forecasts $\hat{y}_1(k),\dots,\hat{y}_6(k)$ to be further fed into the corresponding submetamodels. For the purpose of this study, we sorted the ensemble subsystems in terms of increasing complexity. Such a diversity is aimed at capturing different properties of different parts of the series under consideration. Figure 3 shows the last 30 days of the original time series and the ensemble subsystems' forecasts $\hat{y}_1(k),\dots,\hat{y}_6(k)$. We can see that long-term trends are more or less well captured by all subsystems, but short-term changes pose a problem to all of them so that no single subsystem is significantly better than the others for all data points.

The corresponding forecasts demonstrate a decreasing trend of the forecasting errors presented in Table 1. We used a set of error measures widely adopted in time series forecasting:
1. Mean Absolute Error (MAE);
2. Mean Absolute Scaled Error, scaled by a 1-step ahead naive forecast (MASE1);

3. Mean Absolute Scaled Error, scaled by a 7-step ahead (as the original time series has a weekly seasonality) naive forecast (MASE7);
4. Mean Absolute Percentage Error (MAPE);
5. Symmetric Mean Absolute Percentage Error (SMAPE);
6. Root Mean Square Error (RMSE);
7. Normalized Root Mean Square Error (NRMSE), normalized by the standard deviation of the original time series.

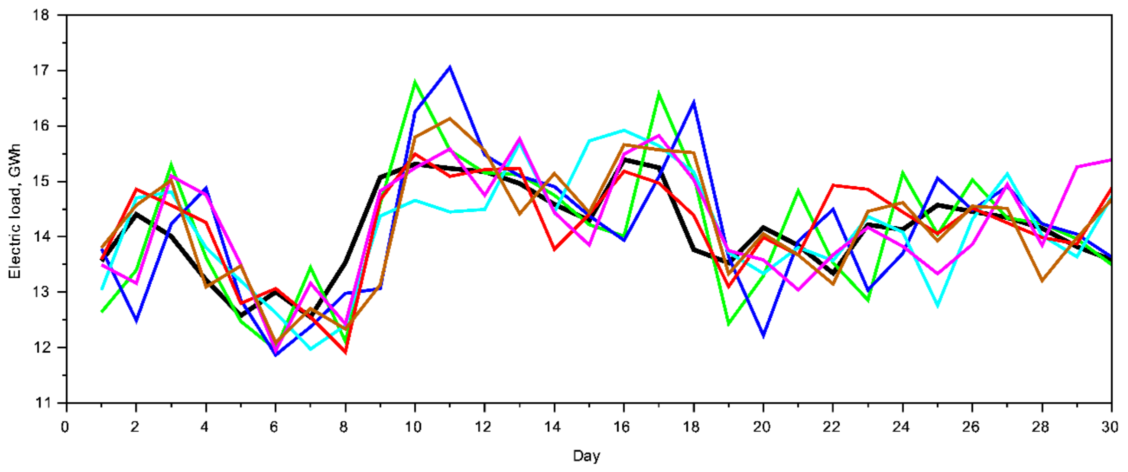

Figure 3: Illustration of the last 30 days of the dataset: original electric load $y(k)$ (black line) and 6 independent 1-day ahead forecasts $\hat{y}_1(k),\ldots,\hat{y}_6(k)$ (color lines).

Table 1
Forecasting errors at all the adaptive cascade bagging system inputs and outputs

System input #	#1	#2	#3	#4	#5	#6
MAE	0.8598	0.7829	0.5886	0.5802	0.5663	0.5641
MASE1	1.6377	1.4912	1.1211	1.1052	1.0786	1.0744
MASE7	1.2288	1.1189	0.8412	0.8293	0.8093	0.8062
MAPE	7.5066	6.8171	5.1015	5.0311	4.8827	4.8580
SMAPE	7.5078	6.8376	5.0851	5.0169	4.8576	4.8534
RMSE	1.0975	0.9682	0.7756	0.7721	0.7513	0.7279
NRMSE	0.7327	0.6464	0.5179	0.5155	0.5016	0.4860
System output #	#1	#2	#3	#4	#5	#6
MAE	0.8598	0.7156	0.5869	0.5131	0.4811	0.4550
MASE1	1.6377	1.3631	1.1178	0.9773	0.9164	0.8667
MASE7	1.2288	1.0227	0.8387	0.7333	0.6876	0.6503
MAPE	7.5066	6.2362	5.1075	4.4706	4.1881	3.9552
SMAPE	7.5078	6.2571	5.1038	4.4553	4.1686	3.9408
RMSE	1.0975	0.9284	0.7483	0.6690	0.6305	0.5951
NRMSE	0.7327	0.6198	0.4996	0.4467	0.4210	0.3973

During the simulation, we applied the proposed adaptive cascade bagging system to generate six forecasts $\hat{y}_1^*(k),\ldots,\hat{y}_6^*(k)$, where $\hat{y}_1^*(k)$ actually duplicates $\hat{y}_1(k)$ and $\hat{y}_2^*(k),\ldots,\hat{y}_6^*(k)$ are produced by the corresponding submetamodels. We treated the forecasting process as an online operation, mirroring the real-time nature of electric load management. This means the entire dataset was processed sequentially, sample by sample, without the traditional division into training, validation, and test sets. This online processing approach reflects a core design principle of the adaptive cascade bagging system – its ability to learn and adapt in real-time as new data arrives.

Figure 4 plots the original time series against 6 ensemble outputs $\hat{y}_1^*(k),\ldots,\hat{y}_6^*(k)$. The corresponding forecasting errors are presented in the lower part of Table 1. An analysis of the forecasts plots and the corresponding forecasting errors (here we refer to MAPE to avoid ambiguity) reveals the following:

1. The output error of each submetamodel is lower than that of the preceding one.
2. The output error of each submetamodel is usually lower than that at both of its inputs.

This progressive reduction in errors reflects the refinement process inherent to the cascading architecture, where each submetamodel builds upon the improvements made by the previous ones.

3. The output error of the 4th submetamodel already falls below than the output error of the best-performing (6th) ensemble subsystem alone. This demonstrates that even a relatively shallow cascade system can outperform the best individual subsystem within the ensemble.
4. The output error of the best (6th) submetamodel is significantly (by the factor 1.23) lower than the output error of the best (6th) ensemble subsystem.

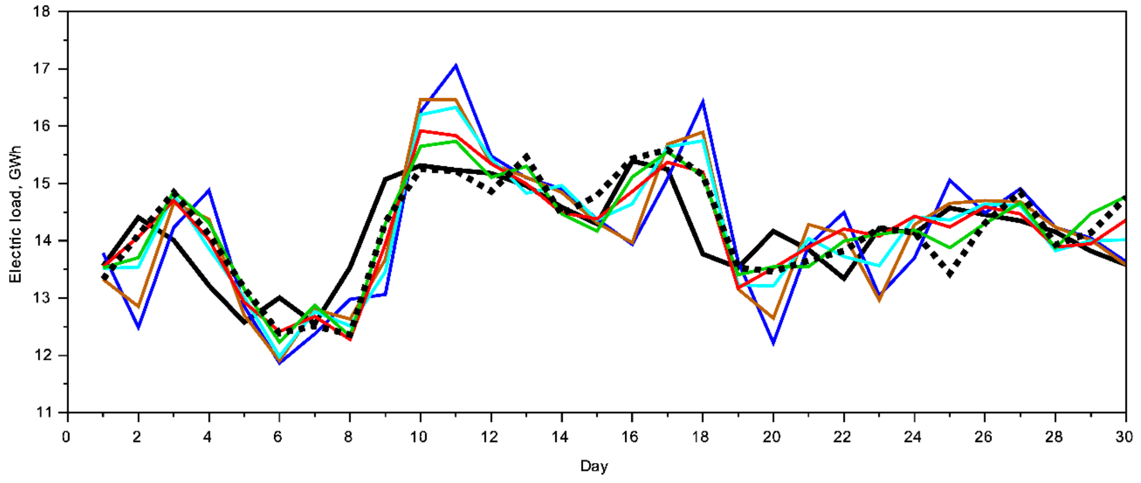

Figure 4: Original electric load $y(k)$ (solid black line) and the adaptive cascade bagging results: $\hat{y}_1^*(k)$ (blue line), $\hat{y}_2^*(k)$ (brown line), $\hat{y}_3^*(k)$ (cyan line), $\hat{y}_4^*(k)$ (red line), $\hat{y}_5^*(k)$ (green line), $\hat{y}_6^*(k)$ (dotted black line).

These observations demonstrate the effectiveness of the cascading approach and highlight the significant efficiency gains it provides.

Perhaps most importantly, these properties have direct implications for the system's efficiency. Consider a scenario where MAPE level of 4.5% is deemed acceptable for the task at hand. Our analysis reveals that none of the six individual ensemble subsystems alone can achieve this level of accuracy. However employing only the first four (simplest) ensemble subsystems within the proposed adaptive cascade bagging system is sufficient to consistently achieve the desired accuracy of 4.5% or better. This demonstrates a significant reduction in computational resources and complexity, as only a fraction of the overall system (4 simplest out of 6 total ensemble subsystems) is required to meet the performance target.

This means that on each forecasting step we can use only a minimally sufficient number of the simplest ensemble subsystems in the cascade system to achieve the desired result and skip calculations of more complex ensemble subsystems, hence conserving the computational resources which can be beneficial e.g. in embedded systems running on battery power. If the accuracy drops below the desired level, additional ensemble subsystems and the corresponding submetamodels can be switched on without retraining the preceding part of the adaptive cascade bagging system.

5. Conclusions

To address the challenges of processing complex and dynamic signals, we introduced a novel adaptive cascade bagging system. This system leverages the power of ensemble learning to achieve optimized results while maintaining the flexibility of online tuning. The system's design is rooted in the cascade approach, where the outputs of multiple computational intelligence systems (e.g., neural networks, support vector machines, fuzzy logic systems) are processed sequentially through a series of simple submetamodels. The core advantage lies in its ability to dynamically adjust both the weighting of individual ensemble subsystems and the number of ensemble subsystems themselves, all while processing data in real-time. This is particularly crucial when dealing with disturbed non-stationary signals – signals that are both noisy and whose characteristics change over time, making traditional, offline approaches less effective. This online tuning capability is essential for handling non-stationary signals where the optimal combination of ensemble subsystems may change as the signal characteristics evolve. It ensures that the system adapts to changing signal characteristics and maintains optimal performance over time, without requiring manual intervention or retraining.

From a computational standpoint, the proposed system is remarkably simple. It is specifically designed for online processing scenarios where data arrives at a sufficiently high rate. The cascade structure, combined with efficient optimization algorithm, minimizes the computational overhead required for processing each data point. This allows the system to operate in real-time, making it suitable for applications such as anomaly detection in network traffic, predictive maintenance of industrial equipment, or real-time financial trading.

A detailed analysis of the simulation results demonstrates the effectiveness of the proposed adaptive cascade bagging system, revealing a progressive reduction in output errors with each subsequent submetamodel, consistently outperforming the ensemble subsystems. Notably, the 4th submetamodel's accuracy already surpasses that of the best individual ensemble subsystem, and the final (6th) submetamodel achieves a 1.23-fold reduction in MAPE compared to the best individual ensemble subsystem. This cascade architecture allows for significant efficiency gains; specifically, achieving an acceptable MAPE level of 4.5% requires only the first four simplest ensemble subsystems, a substantial reduction in computational resources and complexity compared to utilizing the entire six-subsystems ensemble.

Our further research will focus on generalizing the proposed architecture and the learning algorithm to a multivariate case.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] H. Wu, D. Levinson, The ensemble approach to forecasting: A review and synthesis. Transportation Research Part C: Emerging Technologies, 132 (2021) 103357. https://doi.org/10.1016/j.trc.2021.103357

[2] N. Rane, S.P. Choudhary, J. Rane, Ensemble deep learning and machine learning: applications, opportunities, challenges, and future directions. Studies in Medical and Health Sciences, 1(2) (2024) 18-41. http://dx.doi.org/10.2139/ssrn.4849885

[3] T. N. Rincy, R. Gupta, Ensemble learning techniques and its efficiency in machine learning: A survey, in: Proc. 2nd international conference on data, engineering and applications (IDEA), IEEE, 2020, pp. 1-6. https://doi.org/10.1109/IDEA49133.2020.9170675

[4] A. Belayneh, J. Adamowski, B. Khalil, J. Quilty, Coupling machine learning methods with wavelet transforms and the bootstrap and boosting ensemble approaches for drought prediction. Atmospheric research, 172 (2016) 37-47. https://doi.org/10.1016/j.atmosres.2015.12.017

[5] G. Kunapuli, Ensemble methods for machine learning, Simon and Schuster, 2023.

[6] C. Zhang, Y. Ma, Ensemble machine learning: Methods and applications, Springer, 2012.

[7] D. Sarkar, V. Natarajan, Ensemble Machine Learning Cookbook, Packt Publishing Limited, 2019.

[8] X. Dong, Z. Yu, W. Cao, et al, A survey on ensemble learning. Front. Comput. Sci. 14 (2020) 241-58. https://doi.org/10.1007/s11704-019-8208-z

[9] B. Krawczyk, Active and adaptive ensemble learning for online activity recognition from data streams. Knowledge-Based Systems, 138 (2017) 69-78. https://doi.org/10.1016/j.knosys.2017.09.032

[10] L. Von Krannichfeldt, Y. Wang, G. Hug, Online ensemble learning for load forecasting. IEEE Transactions on Power Systems, 36(1) (2020) 545-548. https://doi.org/10.1109/TPWRS.2020.3036230

[11] Ye. Bodyanskiy, P. Otto, I. Pliss, S. Popov, An Optimal Algorithm for Combining Multivariate Forecasts in Hybrid Systems, in: V. Palade, R.J. Howlett, L. Jain (Eds) Knowledge-Based Intelligent Information and Engineering Systems. KES 2003, volume 2774 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2003. https://doi.org/10.1007/978-3-540-45226-3_132.

[12] B. Wang, J. Pineau, Online bagging and boosting for imbalanced data streams. IEEE Transactions on Knowledge and Data Engineering, 28(12) (2016) 3353-3366. https://doi.org/10.1109/TKDE.2016.2609424

[13] A. Bifet, G. Holmes, B. Pfahringer, R. Gavaldà, Improving Adaptive Bagging Methods for Evolving Data Streams, in: Z. H. Zhou, T. Washio (Eds) Advances in Machine Learning. ACML 2009. Lecture Notes in Computer Science, vol 5828. Springer, Berlin, Heidelberg, 2009.. https://doi.org/10.1007/978-3-642-05224-8_4.

[14] E. Lughofer, M. Pratama, I. Skrjanc, Online bagging of evolving fuzzy systems, Information Sciences 570 (2021) 16–33. https://doi.org/10.1016/j.ins.2021.04.041.

[15] A. Shafronenko, Ye. Bodyanskiy, I. Pliss, S. Popov, Evolving neo-fuzzy system for distorted data online processing, in: Proc. 10th International Conference on Advanced Computer Information Technologies (ACIT), IEEE, 2020, pp. 352-355. https://doi.org/10.1109/ACIT49673.2020.9208880

[16] Ye. Bodyanskiy, S. Popov, T. Rybalchenko, Feedforward neural network with a specialized architecture for estimation of the temperature influence on the electric load, in: Proc. 2008 4th International IEEE Conference Intelligent Systems, Varna, Bulgaria, 2008, pp. 7-14–7-18. https://doi.org//10.1109/IS.2008.4670444.

[17] P. Chernenko, O. Martyniuk, S. Popov, Ye. Bodyanskiy, Comparative analysis of two approaches to solving the problem of short-term forecasting of the total electrical load of a power system, Technical Electrodynamics 3, (2013) 61–72.