Neuroevolution of Recurrent Neural Networks for Time Series Forecasting of Coal-Fired Power Plant Operating Parameters

Zimeng Lyu
Rochester Institute of Technology
Rochester, New York, USA
zimenglyu@mail.rit.edu

Shuchita Patwardhan
Microbeam Technologies Inc.
Grand Forks, North Dakota, USA
shuchita@microbeam.com

David Stadem
Microbeam Technologies Inc.
Grand Forks, North Dakota, USA
dstadem@microbeam.com

James Langfeld
Microbeam Technologies Inc.
Grand Forks, North Dakota, USA
jlangfeld@microbeam.com

Steve Benson
Microbeam Technologies Inc.
Grand Forks, North Dakota, USA
sbenson@microbeam.com

Seth Thoelke
Microbeam Technologies Inc.
Grand Forks, North Dakota, USA
sthoelke@microbeam.com

Travis Desell
Rochester Institute of Technology
Rochester, New York, USA
tjdvse@rit.edu

ABSTRACT

This work presents how the Evolutionary eXploration of Augmenting Memory Models (EXAMM) neuroevolution algorithm is incorporated into Microbeam Technologies' condition-based monitoring power plant optimization software using a workflow that integrates coal-fired power plant data collection, evolved RNN predictions and analytic performance indices predictions. To the authors' knowledge, it is the first use of a neuroevolution strategy to evolve recurrent neural networks (RNNs) for forecasting of power plant parameters where the evolved networks have been incorporated into production software used at a coal-fired power plant. A preliminary exploration of the plant's performance shows that after incorporating this software, the amount of revenue lost due to power plant derates and outages decreased by $7.3 million, a savings of 42%, and increased efficiency under medium and low load conditions. A further investigation of the effect of training sequence length and time series data normalization methods on evolving and training RNNs for this system is given, providing practical results useful for real world time series forecasting. It is shown that dividing long time series sequences up into shortened training sequences can dramatically speed up training, and that using different normalization methods (min-max vs. z-score) can provide statistically significant results, dependent on the data sets.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; **Genetic algorithms**; • **Applied computing** → *Forecasting*.

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07…$15.00
https://doi.org/10.1145/3449726.3463196

KEYWORDS

Neuroevolution, Recurrent Neural Networks, Time Series Forecasting, Power Systems

ACM Reference Format:
Zimeng Lyu, Shuchita Patwardhan, David Stadem, James Langfeld, Steve Benson, Seth Thoelke, and Travis Desell. 2021. Neuroevolution of Recurrent Neural Networks for Time Series Forecasting of Coal-Fired Power Plant Operating Parameters. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3449726.3463196

1 INTRODUCTION

A coal fired power plant consists of different subsystems, such as the fuel system, water system, steam system, electrical system, and exhaust gas system. Each of these systems have multiple sensors which generate time series data which can be used to monitor the plant's performance, but can also be used to develop models which can predict future performance. Being able to predict operational parameters of coal fired power plants is an area of significant interest as accurate estimates can be used to improve plant efficiency, reduce emissions or inform plant operators about conditions within the system.

In particular, when conditions in the burners or other systems become poor due to coal quality or other effects, the operator may need to provide supplementary fuel to prevent the burner from going into shutdown. If these events can be predicted early enough then plant conditions can be modified to avoid shutdown without the use of supplementary fuel or other, potentially quite expensive, reactive methods. Additionally, it is important to perform long-term projections of the impact of fuel properties on boiler health. Operators may proactively adjust operations based on the boiler health projections in order to improve the efficiency of the boiler.

With these goals in mind, artificial neural networks (ANNs), and especially recurrent neural networks (RNNs), which specialize in temporal or sequential data, have seen significant use in predicting parameters of interest in coal-fired power plant data [1, 2, 14,

22, 27, 27, 29]. Due to the challenges of training RNNs, which suffer from vanishing and exploding gradients [25], other work has investigated using evolutionary strategies in place of the backpropagation through time (BPTT) algorithm [17, 19, 20, 35]. However, the automated design of RNNs for forecasting parameters of interest in power systems through neuroevolution has not yet begun to be examined, as to our knowledge neuroevolution has only seen some related use in power systems to design controllers for a fuel cell turbine hybrid energy system [6] and a hybrid power plant simulator [12].

To the authors' knowledge, this paper investigates the first use of neuroevolution to evolve RNNs for coal-fired power plant time series forecasting. The evolved RNNs were incorporated into Microbeam's Combustion System Performance Indices - Coal Tracker (CSPI-CT) program to assist the prediction of operating parameters, which is currently seeing production use in a coal-fired power plant. A preliminary exploration of the plant's performance from before the year before the installation of the production software and the last year of use with the software shows the amount of revenue lost due to power plant derates and outages decreased by $7.3 million, a savings of 42%, and increased efficiency under medium and low load conditions.

In addition, this work investigates practical performance enhancements for evolving and training RNNs by examining the effect of different normalization strategies for time series data as well as the effect of training sequence length on the accuracy and performance of neuroevolution and RNN training.

Main features of this work:

- The first use of neuroevolution to evolve RNNs for coal-fired power plant operating parameters forecasting, which have further been incorporated into software seeing production use.
- An exploration of how the use of different time series sequence lengths during training and the choice of different normalization methods affect the forecasting performance.
- Presents Microbeam's workflow of integrating data collection and operation decisions with the neuroevolution evolved RNNs in their CSPI-CT software.

2 RELATED WORK

Artificial neural networks (ANNs) have been widely used in predicting parameters of interest in coal-fired power plant data, such as excess air coefficients and emissions. Zhou *et al.* utilized ANNs to predict the nitrogen oxide (NO_x) emission characteristics of a large capacity pulverized coal fired boiler, showing a more convenient and direct approach compared to other modeling techniques, such as computational fluid dynamics [34]. Teruel *et al.* used ANNs to predict ash deposits in coal-fired boilers, having developed their model with the aid of a case study where a furnace was fouled as detected by heat flux meters [30]. Yao *et al.* used ANNs to predict the hydrogen content in coal in power station boilers from proximate analysis [32]. Smrekar *et al.* used two integrated ANNs, representing a turbine and boiler, to predict the power output of a coal fired power plant [28]. Kumari *et al.* predicted the fireside corrosion rate of super heater tubes in a coal-fire boiler using an ANN trained with operational data from an Indian thermal power plant [13]. Cheng *et al.* used ANNs to predict the maximum burning rate and fixed carbon burnout efficiency of 16 typical Chinese coals and 48 of their blends [3], as well as the ignition temperature and activation energy in another work [4]. Onat *et al.* have used an ANN system to predict the excess air coefficient (λ) on coal burners equipped with a CCD camera [22]. Adams *et al.* developed a deep neural network with a modified early stopping algorithm and least square support vector machine to predict SO_x and NO_x emissions during coal conversion process [1].

RNNs and memory cells such as LSTM [10] have seen more recent use, as they have the potential to perform better on time series prediction tasks and better capture long term dependencies. Safdarnejad *et al.* developed a dynamic data-driven model of a coal-fired utility boiler, using a nonlinear auto regressive neural network with external inputs (NARX, a type of RNN), that estimated NO_x and CO emissions 3 hours into the future simultaneously. They also observed that a dynamic model estimates NO_x and CO emissions with higher accuracy than a static model [27]. Chen *et al.* utilized long short-term memory (LSTM) recurrent neural networks (RNNs) to predict NO_x emissions from a catalytic reduction process [2]. Tan *et al.* have used dynamic modeling to predict NO_X emission in a 660 MW coal-fired boiler with their work showing that LSTMs out perform support vector machines (SVMs) in time series prediction[29]. Laubscher has also used an RNN encoder decoder network to predict coal-fired power plant reheater metal temperatures using plant operational data [14].

3 METHODOLOGY

This work utilizes the Evolutionary eXploration of Augmenting Memory Models (EXAMM) algorithm [23] to drive the neuroevolution process. EXAMM has a multi-threaded implementation for multi-core CPUs as well as an MPI [21] implementation that allows EXAMM to readily leverage high performance computing resources using an asynchronous, island based distributed computing strategy which allows the workers to complete the training of the generated RNNs at whatever speed they are capable of, yielding an algorithm that is naturally load-balanced. A master process maintains the populations for each island and generates new RNN candidate models from the islands in a round-robin manner. Workers receive candidate models and locally train them with back-propagation through time (BPTT), making EXAMM a memetic algorithm. When a worker completes the training of an RNN, that RNN is inserted back into the island that it originated from. Then, if the number of RNNs in an island exceeds the island's maximum population size, the RNN with the worst fitness score, validation set mean squared error (MSE), is deleted.

EXAMM evolves progressively larger candidate models (RNNs) in response to worker requests through a series of mutation and crossover (reproduction) operations. Mutations can be edge-based: *split edge*, *add edge*, *enable edge*, *add recurrent edge*, and *disable edge* operations, or work as higher-level node-based mutations: *disable node*, *enable node*, *add node*, *split node* and *merge node*. The type of node to be added is selected uniformly at random from a suite of simple neurons and complex memory cells: Δ-RNN units [24], gated recurrent units (GRUs) [5], long short-term memory cells (LSTMs) [10], minimal gated units (MGUs) [33], and update-gate

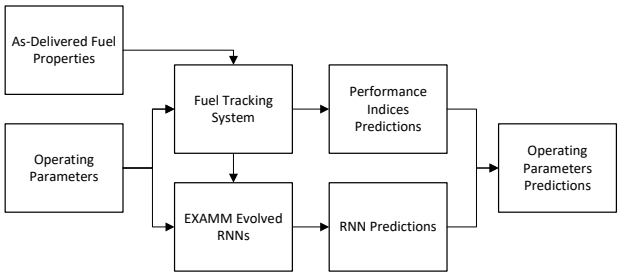

Figure 1: Integration of EXAMM's evolved neural networks into Microbeam's power plant optimization software

RNN cells (UGRNNs) [7]. This allows EXAMM to select for the best performing recurrent memory units. A *clone* operation also exists to allow existing networks to continue training without modification. EXAMM also allows for *deep recurrent connections* which enables the RNNs to directly use information beyond the previous time step. These deep recurrent connections have proven to offer significant improvements in model generalization, even yielding models that outperform state-of-the-art gated architectures [8]. Child genomes generated by crossover or mutation inherit their parents' weights with a Lamarckian weight inheritance strategy, which can significantly reduce the number of the epochs needed for training child genomes, improving the performance of the evolutionary process [18].

4 ARCHITECTURE

Figure 1 shows the workflow of integrating Microbeam's data collection and operation decisions with EXAMM. *As-delivered fuel properties* are accessed through an on-line analyzer. They provide real-time measurements of fuel properties such as ash content, heating value, and ash properties. The *fuel tracking system* tracks the fuel from delivery from the mine to the burner, maintaining an inventory of fuel properties and their position in the fuel handling system. It generates projected fuel properties a number of hours in advance of firing based on the inventory of fuel. In order to track the fuel, measurements of *operating parameters* such as belt positions and speeds are obtained online from the plant's Distributed Control System (DCS) archive. RNNs evolved by EXAMM make predictions of key operating parameters such as flame intensity based on the *operating parameters* from the plant and projected fuel properties from the *fuel tracking system*. These RNNs predict future operating parameters for an array of predefined time steps into the future, typically ranging between 1 minute and 8 hours.

Microbeam predicts performance indices for each individual time step in the future based on projected fuel properties. Performance indices are derived from Microbeam's understanding of the impacts of fuel properties on plant performance and are used to predict key operating parameters in parallel to RNN predictions of those same operating parameters. These *performance indices predictions* supplement the intermittent predictions from the RNNs. Composite *operating parameter predictions* are made via a weighted average of the current operating parameter value, RNN predictions, and performance indices-derived predictions.

RNNs are evolved by EXAMM using historical data and the best evolved RNNs are used to make predictions for intermittent time steps in the future. The *performance indices predictions*, meanwhile, have no recurrent connections or input from current operating parameters and predict the impact of fuel properties on operating parameters at all time steps into the future. Therefore, the indices-derived predictions are used as a secondary prediction of plant performance to determine the impact of short-term fluctuations in fuel properties on plant performance when neural-network-derived predictions are unavailable.

The final *operating parameters predictions* are averages of RNN derived and indices-derived predictions predictions, weighted by time and prediction type. More recent time steps have heavier weights than previous time steps, and RNN derived predictions are assigned heavier weights than indices-derived predictions.

Figure 2 is an example of combining *operating parameter predictions* from RNN predictions and *performance indices*. The evolved RNNs predict the future flame intensity values at 1, 15, 30, 60, 120, 240, and 720 minutes into the future. The *performance indices predictions* show short-term fluctuations in flame intensity as a supplement to the RNN-derived predictions.

A screenshot of Microbeam's CSPI-CT power plant optimization program is shown in Figure 3. It can alert the plant operators and engineers if poor boiler condition occurs based on coal conditions, power plant conditions, and operating parameter values. Clockwise from bottom left, the small sections are cyclone burners (coal and air mix and burn to produce flue gas), water wall (hot area of the boiler where the flame radiates heat into water-cooled walls), secondary superheater, reheater, primary superheater, and economizer sections (progressively cooler flue gas passes between steam or water-filled tubes, heating the steam inside), and air heater (warm flue gas is used to warm incoming air to boost achieve higher process efficiency).

In each section, Microbeam uses a combination of current measured *operating parameters* and forecasted *operating parameter predictions* to estimate the performance of that section. The performance of the section is scored by a value from 0 to 100, where lower values are better. The overall plant performance is also evaluated; the overall score is a combination of all individual sections' performances.

5 RESULTS

5.1 Datasets

Two datasets were used for our experiments, the first one comes from the coal fired power plant cyclone data, the second one is from the plant's boiler. Operational parameters for each dataset were combined with the coal tracking output to produce the final datasets. The Boiler dataset is composed of 42 operational parameters and 16 coal tracking parameters, consisting of over 407 days of hourly data. We only used the time sequences for training and evaluation when the coal properties values are available, which means each of the time sequences files have different length. There are 11 training files with time sequence length of 94, 1592, 361, 544, 64, 168, 1022, 1046, 1212, 184, 337, and 1687. There are 4 validation files with time sequence length of 1048, 245, 40, and 159. These files were divided by plant shutdowns, resulting in the varying lengths. The

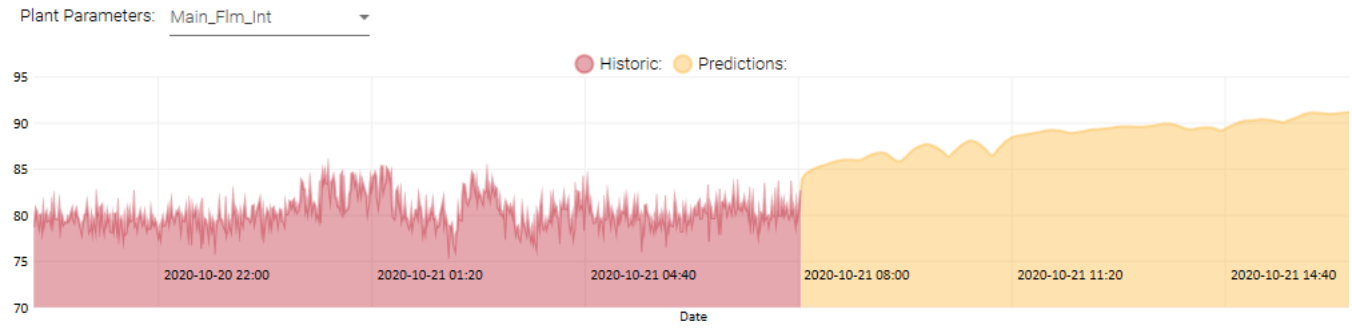

Figure 2: Microbeam's plant optimization program showing predicted burner flame intensity based on EXAMM and performance indices. Past measurements of flame intensity shown in red, predicted flame intensity shown in yellow

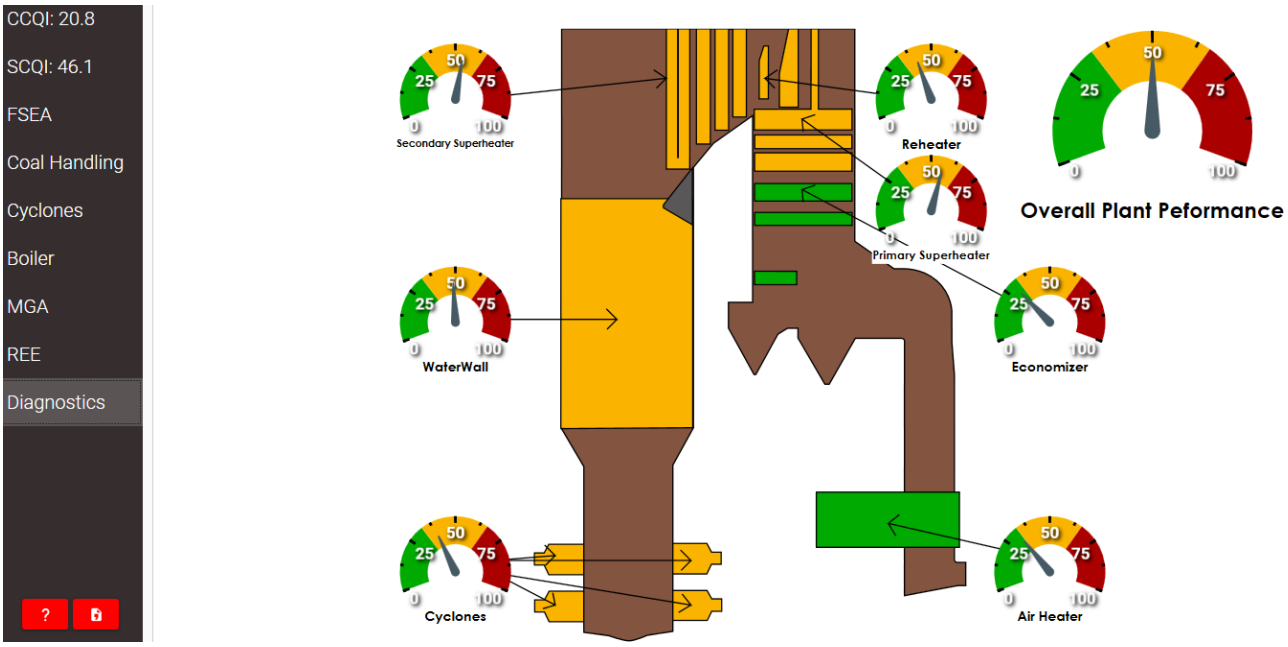

Figure 3: Microbeam's plant optimization program presents a boiler diagram with plant performance ratings by boiler section

cyclone dataset contains 15 operational parameters and 8 Coal Tracker parameters, consisting of over 44 days worth of per-minute data. The training set has time sequence length of 50687 and the validation set has a time sequence length of 12673.

The parameters of the most interest for the Boiler dataset were *Net Plant Heat Rate* and *Secondary Superheater Temperature*. These operating parameters were selected for prediction by EXAMM. Net Plant Heat Rate is a measure of the plant's overall efficiency, so minimizing this number means lower carbon emissions and reduced cost. Accurately predicting this value means operators can adjust operations as needed if the plant is expected to lose efficiency. Secondary Superheater Temperature refers to the temperature of the steam exiting the secondary superheater section, just before entering the steam turbine to be converted to electricity. This parameter is related to overall efficiency of the plant but is more sensitive to local changes in performance in the water wall and secondary superheater region. Therefore the Secondary Superheater Temperature allows Microbeam to determine performance in the secondary superheater section of the boiler. An example of the Secondary Superheater Temperature value is shown in Figure 4.

The parameter of the most interest for the Cyclone dataset is *Main Flame Intensity*. An optical pyrometer measures the intensity of the flame in the burner. Fuel properties often impact the flame intensity through a process known as slagging. If a slagging event occurs, a costly supplementary fuel additive may be required in order to restore the flame. Avoiding a slagging event means both cost and environmental savings. The plant saves money on the cost of using supplementary fuel and reduces carbon emissions.

5.2 Hyperparameter Settings

Each EXAMM run used 10 islands, each with a maximum capacity of 10 genomes. EXAMM was then allowed to evolve and train 20, 000

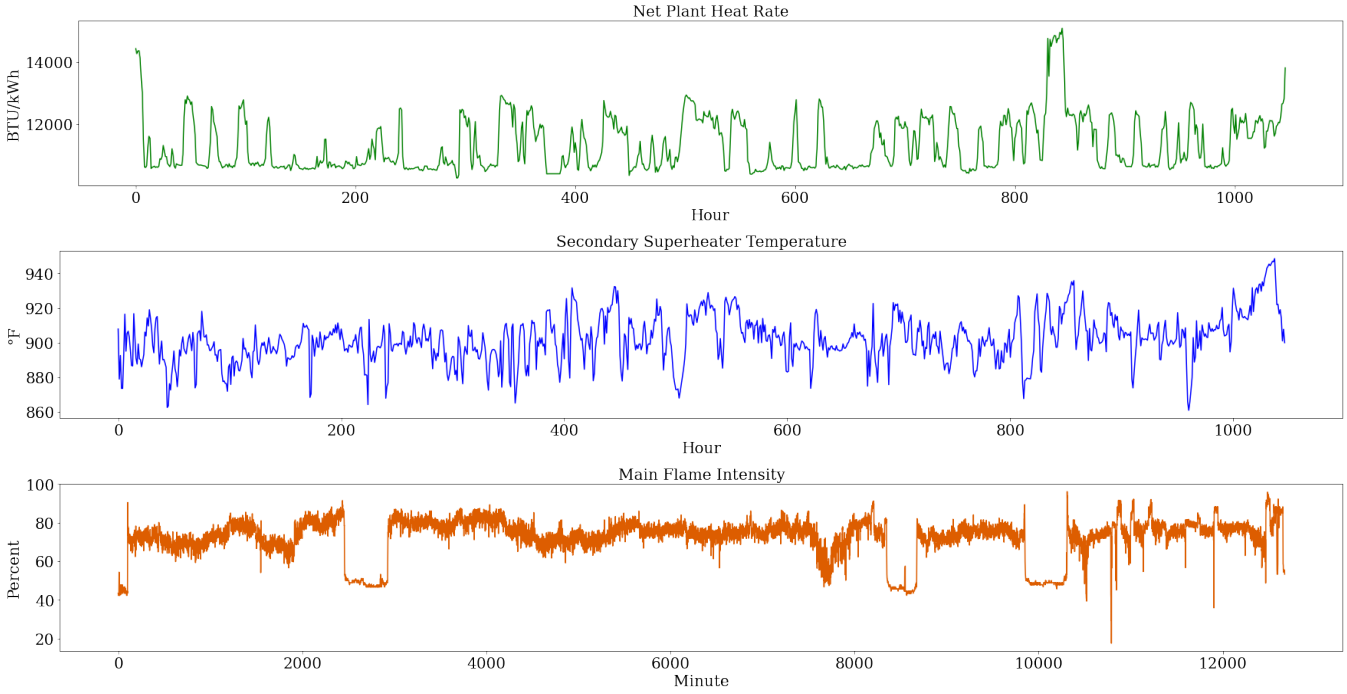

Figure 4: Example of output parameters from boiler and cyclone datasets

genomes (RNNs) through its neuroevolution process. The initial genome weights are initialized by the Xavier weight initialization method [9], after which child genome weights were initialized using EXAMM's Lamarckian weight inheritance method [18]. New RNNs were generated via mutation at a rate of 70%, intra-island crossover at a rate of 20%, and inter-island crossover at a rate of 10%. 10 out of EXAMM's 11 mutation operations were utilized (all except for *split edge*), and each was chosen with a uniform 10% chance. EXAMM generated new nodes by selecting from simple neurons, Δ-RNN, GRU, LSTM, MGU, and UGRNN memory cells uniformly at random. Recurrent connections could span any time-skip generated randomly between $\mathcal{U}(1, 10)$. All RNNs were locally trained for 10 epochs via stochastic gradient descent (SGD) and using back propagation through time (BPTT) [31] to compute gradients with a learning rate of $\eta = 0.001$ (for net plant heat rate and main flame intensity) and $\eta = 0.0005$ (for secondary superheater temperature) and used Nesterov momentum with $\mu = 0.9$. For the memory cells with forget gates, the forget gate bias had a value of 1.0 added to it (motivated by [11]). To prevent exploding gradients, gradient scaling [25] was used when the norm of the gradient exceeded a threshold of 1.0. To combat vanishing gradients, gradient boosting (the opposite of scaling) was used when the gradient norm was below 0.05. These parameters have been selected by hand-tuning during the prior experience.

5.3 Effects of Sequence Length

While it is generally assumed that recurrent memory cells sufficiently capture long term dependencies, it might be assumed that the length of training sequences would not have a significant effect on performance and accuracy of training RNNs. However, when training via BPTT, RNNs are unrolled to generate a feed forward graph over every time step of the input sequence, and weights are updated after either a backwards pass over the entire data set or batch of data sets. Because of this, there is a trade off – if the time sequences are short, weight updates will be frequent but there will be limited temporal information available for learning; however, if the time sequences are too long, it will significantly increase the training time because of the additional computation required for each weight update.

The optimal time sequence length for training RNNs will vary according to the task's expected outcome and correlations between parameters for different datasets. To explore how different lengths of time sequence affect the RNN time series training performance and prediction results, the training data was divided into varying time sequence lengths. The time sequence length for validation data sets were not modified (*i.e.*, they were left as long sequences) as this would not have an effect on training the RNNs. Additionally, how far in the future the RNNs predict (the prediction *time offset*) can also influence the prediction performance. For example, with a training sequence of length 50, training RNNs with a time offset 1, there will be 49 time steps available as input, however for a time offset 8, there will only be 42 time steps available. Due to this, larger time offsets might require longer input sequences.

To study the effects of training sequence length for a variety of time offsets, the combination of time offsets of 1, 2, 4, 8 on training the data sets with time sequence lengths of 50, 100, 200 and full for the boiler data set (*Net Plant Heat Rate* and *Secondary Superheater Steam Outlet Temperature*), and time sequence length of 50, 500,

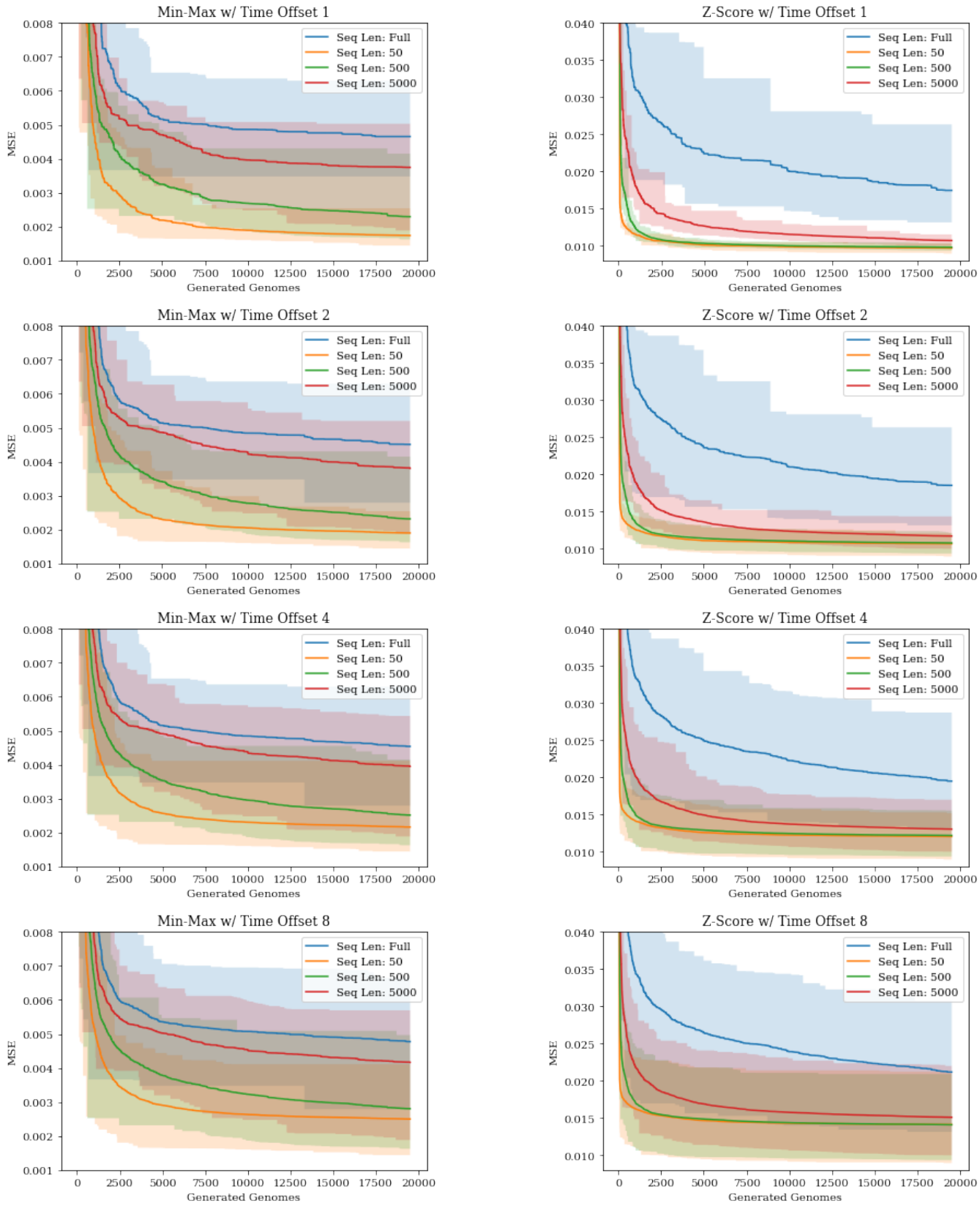

Figure 5: Main Flame Intensity

Table 1: Mean Absolute Percentage Error (MAPE) of the overall best and average best evolved RNNs from 20 repeated experiments for predicting net plant heat rate and secondary superheater temperature on the validation dataset

		Net Plant Heat Rate					Secondary Superheater Temperature				
		Best Case		Average Case			Best Case		Average Case		
Sequence Length	Time Offset	Normalization		Normalization		Mann-W U P-Values	Normalization		Normalization		Mann-W U P-Values
		Min-Max	Z-Score	Min-Max	Z-Score		Min-Max	Z-Score	Min-Max	Z-Score	
50	1	0.0196	**0.0184**	0.0216	**0.0198**	**0.0001**	**0.0066**	0.0067	0.0068	**0.0068**	0.4196
	2	0.0304	**0.0299**	0.0334	**0.0317**	**0.0017**	**0.0086**	0.0087	0.0089	**0.0089**	0.1427
	4	0.0408	**0.0394**	0.0429	**0.0421**	0.1251	0.0102	**0.0097**	0.0104	**0.0102**	**0.0001**
	8	0.0451	**0.0450**	**0.0472**	0.0481	**0.0266**	0.0114	**0.0111**	0.0117	**0.0113**	**0.0000**
100	1	0.0217	**0.0197**	0.0269	**0.0231**	**0.0014**	**0.0068**	0.0068	**0.0069**	0.0070	0.4516
	2	0.0315	**0.0301**	0.0361	**0.0339**	**0.0036**	0.0087	**0.0087**	0.0091	**0.0090**	0.0903
	4	0.0415	**0.0396**	0.0442	**0.0431**	**0.0301**	**0.0102**	0.0102	0.0106	**0.0104**	**0.0014**
	8	**0.0443**	0.0472	**0.0471**	0.0487	**0.0003**	0.0114	**0.0110**	0.0117	**0.0114**	**0.0000**
200	1	0.0252	**0.0248**	**0.0304**	0.0309	0.2804	**0.0070**	0.0070	**0.0072**	0.0073	0.0568
	2	**0.0315**	0.0356	**0.0378**	0.0389	0.2285	0.0090	**0.0089**	0.0092	**0.0092**	0.4838
	4	**0.0425**	0.0426	**0.0447**	0.0463	**0.0072**	0.0104	**0.0104**	0.0107	**0.0106**	**0.0003**
	8	**0.0426**	0.0464	**0.0479**	0.0495	**0.0111**	0.0115	**0.0112**	0.0118	**0.0116**	**0.0339**
Full	1	**0.0275**	0.0382	**0.0348**	0.0451	**0.0000**	**0.0072**	0.0073	0.0077	**0.0077**	0.3882
	2	**0.0355**	0.0405	**0.0406**	0.0511	**0.0000**	0.0091	**0.0090**	**0.0095**	0.0095	0.1042
	4	**0.0428**	0.0504	**0.0471**	0.0543	**0.0000**	0.0106	**0.0105**	**0.0109**	0.0109	0.3882
	8	**0.0469**	0.0517	**0.0497**	0.0557	**0.0000**	0.0117	**0.0114**	0.0119	**0.0118**	**0.0036**

Table 2: Mean Absolute Percentage Error (MAPE) of the overall best and average best evolved RNNs from 20 repeated experiments for predicting main flame intensity on the validation dataset

		Main Flame Intensity				
		Best Case		Average Case		
Sequence Length	Time Offset	Normalization		Normalization		Mann-W U P-Values
		Min-Max	Z-Score	Min-Max	Z-Score	
50	1	0.0310	**0.0297**	**0.0338**	0.0648	0.2124
	2	0.0320	**0.0317**	**0.0359**	0.0932	0.1488
	4	0.0362	**0.0339**	**0.0402**	0.1140	**0.0206**
	8	0.0407	**0.0379**	**0.0443**	0.0751	0.2989
100	1	0.0320	**0.0301**	**0.0391**	0.0601	0.1971
	2	0.0336	**0.0318**	**0.0389**	0.0807	0.1042
	4	0.0356	**0.0355**	**0.0421**	0.0787	0.2896
	8	0.0413	**0.0379**	**0.0456**	0.0603	**0.0339**
200	1	0.0360	**0.0314**	0.0514	**0.0502**	**0.0028**
	2	0.0457	**0.0335**	0.0528	**0.0363**	**0.0000**
	4	0.0423	**0.0354**	**0.0538**	0.0914	0.1685
	8	0.0461	**0.0400**	**0.0560**	0.0625	**0.0039**
Full	1	0.0492	**0.0359**	**0.0575**	0.0817	**0.0077**
	2	0.0423	**0.0365**	**0.0552**	0.0573	**0.0147**
	4	0.0481	**0.0418**	**0.0564**	0.0821	0.0702
	8	0.0524	**0.0431**	**0.0617**	0.1063	0.1754

5000 and full for the cyclone data set (*Main Flame Intensity*) were evaluated. The time sequence length "full" means using the original training data without slicing it into shorter time sequences (lengths are provided in Section 5.1).

Figure 5 shows an example of the convergence rate of validation mean squared error (MSE) of *Main Flame Intensity* using different time sequence lengths and with different time offsets over 20 repeated runs. Due to space limit, the convergence rate for *Net Plant Heat Rate* and *Secondary Superheater Steam Outlet Temperature*, which show similar results, can be found in supplementary materials. Interestingly, and potentially contrary to the recurrent memory cells being able to retain long term dependencies, for all three data sets and all time offsets, using training sequences of length 50 had both the fastest convergence rates and the reached the lowest validation MSE in both average and best cases. Additionally it was observed that as the training sequence length increased, the RNNs trained slower and had worse validation MSE performance.

For the same training dataset, using different time sequence lengths for training data not only effects the convergence rate, but also the overall prediction performance. While the optimal time sequence length that can best aid the training and performance process might vary for different datasets and different tasks, it is an easy and effective way to improve the prediction performance in practice.

5.4 Effects of Data Normalization

As another means of improving performance of the evolved RNNs for usage by Microbeam, two data normalization strategies were investigated. The most common neural network data normalization method is to use *z-scores*:

$$X_{normalized} = \frac{X - \mu}{\delta} \tag{1}$$

where μ and δ are the mean and standard deviation of the features. Z-scores generally handle outliers well, however if features have large variances, those normalized features can be very close to zero. Additionally, the normalized features do not have the same value ranges, which can lead to challenges in forecasting exact values.

Another common method is *min-max* normalization which scales all features within the range [0,1]:

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2}$$

where X_{min} and X_{max} are the minimum and maximum possible values for the parameter to be normalized. This method can be particularly effective in time series data prediction where these limits are known *a priori*, and it retains same scale as the original

data which can be beneficial in time series forecasting. However this method can suffer from large outliers.

All the experiments discussed in Section 5.3 were performed using both normalization methods with 20 replications. Figures 6, 7 and 5 show the test results using different data normalization methods. It should be noted that after min-max normalization, input data values will range [0,1], while for z-score normalization the values range is determined by δ and is normally larger than [-1,1] for our datasets, which will result in the z-score results having larger validation MSE values.

For a fair comparison, Tables 1 and 2 present the best case and average case prediction Mean Absolute Percentage Error (MAPE) when comparing the de-normalized output to the unnormlized data, respectively. The values marked in bold are the better results of the two normalization methods. Results for *Net Plant Heat Rate* show that using z-score normalization was better with the shorter sequence lengths of 50 and 100. The performance for both normalization methods are very close for predicting *Secondary Superheater Steam Outlet Temperature*, while min-max performs better than z-score for predicting *Main Flame Intensity*. Interestingly, the results show that the z-score and min-max normalization methods can have different effects on predicting different data sets, or even the same data set but with different output parameters or time offsets, in some cases even over 1% MAPE in the best and average cases. Tables 1 and 2 also show p-values of the Mann–Whitney U test between using min-max and z-score normalization methods over 20 repeated runs. The values marked in bold means two normalization methods are statistically significantly different with significant level $\alpha = 0.05$.

5.5 Performance Improvements at the Plant

A full examination of the effectiveness of Microbeam's optimization software is currently under analysis, however given preliminary results, significant improvements have already been noted at the plant. The online fuel analyzer and plant optimization software were installed in 2018, with several upgrades over time and ongoing support and maintenance into 2021. Due to this, the years 2017 to 2020 were selected to show operational parameters before installation and throughout the upgrade process. The amount of revenue lost due to derates and outages decreased by $7.3 million from 2017 to 2020, a savings of 42%. The net plant heat rate improved under medium and low load conditions, decreasing about 3% under medium load and 10% under low load since 2017. Under high load conditions, the plant heat rate did not significantly change over time. Other statistics, such as amount of supplementary oil fired, contribute to further savings and are under further investigation.

6 CONCLUSION

In this work, the Evolutionary eXploration of Augmenting Memory Models (EXAMM) neuroevolution strategy is used to evolve recurrent neural networks with varying memory cells and recurrent connections of varying time lags to perform time series forecasting of varying coal fired power plant parameters for varying amounts of time in the future, which are integrated into Microbeam's coal-fired power plant optimization software so that plant operators can better adjust operations to improve efficiency and reduce emissions. While there have been a number of studies utilizing artificial neural networks to perform prediction and classification tasks for power systems, to the authors' knowledge, this work presents the first work in which a neuroevolution strategy is used to design neural networks for use by a power plant, and even further the first example of evolved neural networks being used in production software by a power plant. A preliminary investigation of the impact of the use of this software by the coal-fired power plant show significant revenue savings due to reduced derates and outages, as well as other improved efficiencies.

This work additionally investigates the effect of different normalization strategies and training sequence lengths on the convergence rates and predictive accuracy of the evolved neural networks, areas which are commonly overlooked in machine learning experiments on benchmark datasets, but which, as shown in this work, can have a significant impacts on real world problems. Results show that training on shorter sequences dramatically improves convergence rates and predictive ability of the evolved neural networks, somewhat contrary to the common belief that recurrent memory cells are capable of capturing long term dependencies in temporal data. Further, results show that there is a "no free lunch" effect from the selection of data normalization rates, that with statistical significance the performance of networks potentially varying by over 1% mean average percent error (MAPE) even within a single data set with predictions of varying time offsets. The authors hope these results can help inform future time series forecasting efforts using neural networks for real world applications.

This work also opens up future research directions to further investigate and enhance the performance of time series forecasting in practice. 1) EXAMM currently only reads the input data stream one timestep at a time and predicts one timestep at a time. Future work could have EXAMM able to read and predict the complete time sequence. And it might potentially help better capturing the outliers and abnormal performance in the coal-fired power plant. 2) We could further investigate the effect of other data augmentation methods for training, such as *window slicing*, *window warping* [15], *add noise*, *scale samples* [16]. Finding the effective data augmenting methods could significantly improve the performance. 3) EXAMM currently use validation MSE as objective function to evaluate the fitness of a RNN. Other objective functions could also be used to evaluate RNNs, such as network size, inference time, shape and time distortion loss. We could use multi-objective optimization methods to evaluate and choose best RNN candidates.

ACKNOWLEDGEMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Combustion Systems under Award Number #FE0031547. Most of the computation of this research was done on the high performance computing clusters of Research Computing at Rochester Institute of Technology [26]. We would like to thank the Research Computing team for their assistance and the support they generously offered to ensure that the heavy computation this study required was available.

REFERENCES

[1] Derrick Adams, Dong-Hoon Oh, Dong-Won Kim, Chang-Ha Lee, and Min Oh. 2020. Prediction of SOx–NOx emission from a coal-fired CFB power plant with machine learning: Plant data learned by deep neural network and least square support vector machine. *Journal of Cleaner Production* 270 (2020), 122310.

[2] Jiyu Chen, Feng Hong, Mingming Gao, Taihua Chang, and Liying Xu. 2019. Prediction Model of SCR Outlet NOx Based on LSTM Algorithm. In *Proceedings of the 2019 2nd International Conference on Intelligent Science and Technology*. ACM, 7–10.

[3] Jun Cheng, Xin Wang, Tingting Si, Fan Zhou, Zhihua Wang, Junhu Zhou, and Kefa Cen. 2016. Maximum burning rate and fixed carbon burnout efficiency of power coal blends predicted with back-propagation neural network models. *Fuel* 172 (2016), 170–177.

[4] Jun Cheng, Xin Wang, Tingting Si, Fan Zhou, Junhu Zhou, and Kefa Cen. 2016. Ignition temperature and activation energy of power coal blends predicted with back-propagation neural network models. *Fuel* 173 (2016), 230–238.

[5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[6] Mitchell Colby, Logan Yliniemi, Paolo Pezzini, David Tucker, Kenneth" Mark" Bryden, and Kagan Tumer. 2016. Multiobjective neuroevolutionary control for a fuel cell turbine hybrid energy system. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. 877–884.

[7] Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. 2016. Capacity and Trainability in Recurrent Neural Networks. *arXiv preprint arXiv:1611.09913* (2016).

[8] Travis Desell, AbdElRahman ElSaid, and Alexander G. Ororbia. 2020. An Empirical Exploration of Deep Recurrent Connections Using Neuro-Evolution. In *The 23nd International Conference on the Applications of Evolutionary Computation (EvoStar: EvoApps 2020)*. Seville, Spain.

[9] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks.. In *Aistats*, Vol. 9. 249–256.

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[11] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning*. 2342–2350.

[12] Shauharda Khadka, Kagan Tumer, Mitch Colby, Dave Tucker, Paolo Pezzini, and Kenneth Bryden. 2016. Neuroevolution of a hybrid power plant simulator. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. 917–924.

[13] Amrita Kumari, SK Das, and PK Srivastava. 2016. Modeling Fireside Corrosion Rate in a Coal Fired Boiler Using Adaptive Neural Network Formalism. *Portugaliae Electrochimica Acta* 34, 1 (2016), 23–38.

[14] Ryno Laubscher. 2019. Time-series forecasting of coal-fired power plant reheater metal temperatures using encoder-decoder recurrent neural networks. *Energy* 189 (2019), 116187.

[15] Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. 2016. Data augmentation for time series classification using convolutional neural networks. In *ECML/PKDD workshop on advanced analytics and learning on temporal data*.

[16] Bo Liu, Zhenguo Zhang, and Rongyi Cui. 2020. Efficient Time Series Augmentation Methods. In *2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. IEEE, 1004–1009.

[17] Y.P. Liu, M.G. Wu, and J.X. Qian. 2007. Predicting coal ash fusion temperature based on its chemical composition using ACO-BP neural network. *Thermochimica Acta* 454, 1 (2007), 64 – 68. https://doi.org/10.1016/j.tca.2006.10.026

[18] Zimeng Lyu, AbdElRahman ElSaid, Joshua Karns, Mohamed Mkaouer, and Travis Desell. 2021. An Experimental Study of Weight Initialization and Lamarckian Inheritance on Neuroevolution. *The 24th International Conference on the Applications of Evolutionary Computation (EvoStar: EvoApps)* (2021).

[19] Liang Yu Ma, Yin Ping Ge, and Xing Cao. 2012. Superheated steam temperature control based on improved recurrent neural network and simplified PSO algorithm. In *Applied Mechanics and Materials*, Vol. 128. Trans Tech Publ, 1065–1069.

[20] Michalis Mavrovouniotis and Shengxiang Yang. 2013. Evolving neural networks using ant colony optimization with pheromone trail limits. In *Computational Intelligence (UKCI), 2013 13th UK Workshop on*. IEEE, 16–23.

[21] Message Passing Interface Forum. 1994. MPI: A Message-Passing Interface Standard. *The International Journal of Supercomputer Applications and High Performance Computing* 8, 3/4 (Fall/Winter 1994), 159–416.

[22] Cem Onat and Mahmut Daskin. 2019. A Basic ANN System for Prediction of Excess Air Coefficient on Coal Burners Equipped with a CCD Camera. *Mathematics and Statistics* 7, 1 (2019), 1–9.

[23] Alexander Ororbia, AbdElRahman ElSaid, and Travis Desell. 2019. Investigating Recurrent Neural Network Memory Structures Using Neuro-evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Prague, Czech Republic) *(GECCO '19)*. ACM, New York, NY, USA, 446–455. https://doi.org/10.1145/3321707.3321795

[24] Alexander G. Ororbia II, Tomas Mikolov, and David Reitter. 2017. Learning Simpler Language Models with the Differential State Framework. *Neural Computation* 0, 0 (2017), 1–26. https://doi.org/10.1162/neco_a_01017 arXiv:https://doi.org/10.1162/neco_a_01017 PMID: 28957029.

[25] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*. 1310–1318.

[26] Rochester Institute of Technology. 2019. Research Computing Services. https://doi.org/10.34788/0S3G-QD15

[27] Seyed Mostafa Safdarnejad, Jake F Tuttle, and Kody M Powell. 2019. Dynamic modeling and optimization of a coal-fired utility boiler to forecast and minimize NOx and CO emissions simultaneously. *Computers & Chemical Engineering* 124 (2019), 62–79.

[28] J Smrekar, D Pandit, Magnus Fast, Mohsen Assadi, and Sudipta De. 2010. Prediction of power output of a coal-fired power plant by artificial neural network. *Neural Computing and Applications* 19, 5 (2010), 725–740.

[29] Peng Tan, Biao He, Cheng Zhang, Debei Rao, Shengnan Li, Qingyan Fang, and Gang Chen. 2019. Dynamic modeling of NOx emission in a 660 MW coal-fired boiler with long short-term memory. *Energy* 176 (2019), 429–436.

[30] Enrique Teruel, Cristóbal Cortés, Luis Ignacio Díez, and Inmaculada Arauzo. 2005. Monitoring and prediction of fouling in coal-fired utility boilers using neural networks. *Chemical Engineering Science* 60, 18 (2005), 5035 – 5048. https://doi.org/10.1016/j.ces.2005.04.029

[31] Paul J Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proc. IEEE* 78, 10 (1990), 1550–1560.

[32] HM Yao, HB Vuthaluru, MO Tade, and D Djukanovic. 2005. Artificial neural network-based prediction of hydrogen content of coal in power station boilers. *Fuel* 84, 12 (2005), 1535–1542.

[33] Guo-Bing Zhou, Jianxin Wu, Chen-Lin Zhang, and Zhi-Hua Zhou. 2016. Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing* 13, 3 (2016), 226–234.

[34] Hao Zhou, Kefa Cen, and Jianren Fan. 2004. Modeling and optimization of the NOx emission characteristics of a tangentially fired boiler with artificial neural networks. *Energy* 29, 1 (2004), 167 – 183. https://doi.org/10.1016/j.energy.2003.08.004

[35] Hao Zhou, Jia Pei Zhao, Li Gang Zheng, Chun Lin Wang, and Ke Fa Cen. 2012. Modeling NOx emissions from coal-fired utility boilers using support vector regression with ant colony optimization. *Engineering Applications of Artificial Intelligence* 25, 1 (2012), 147–158.

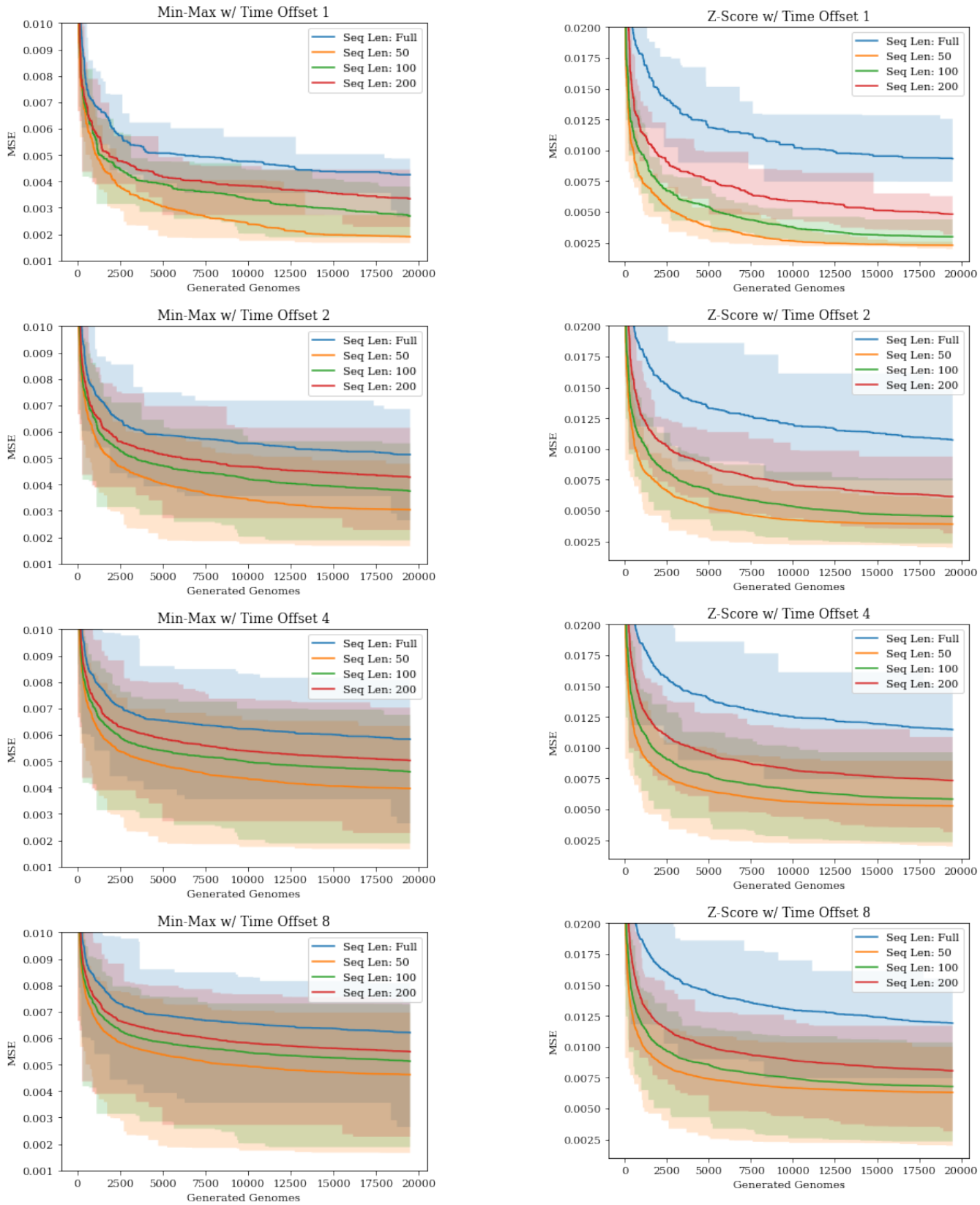

Figure 6: Net Plant Heat Rate

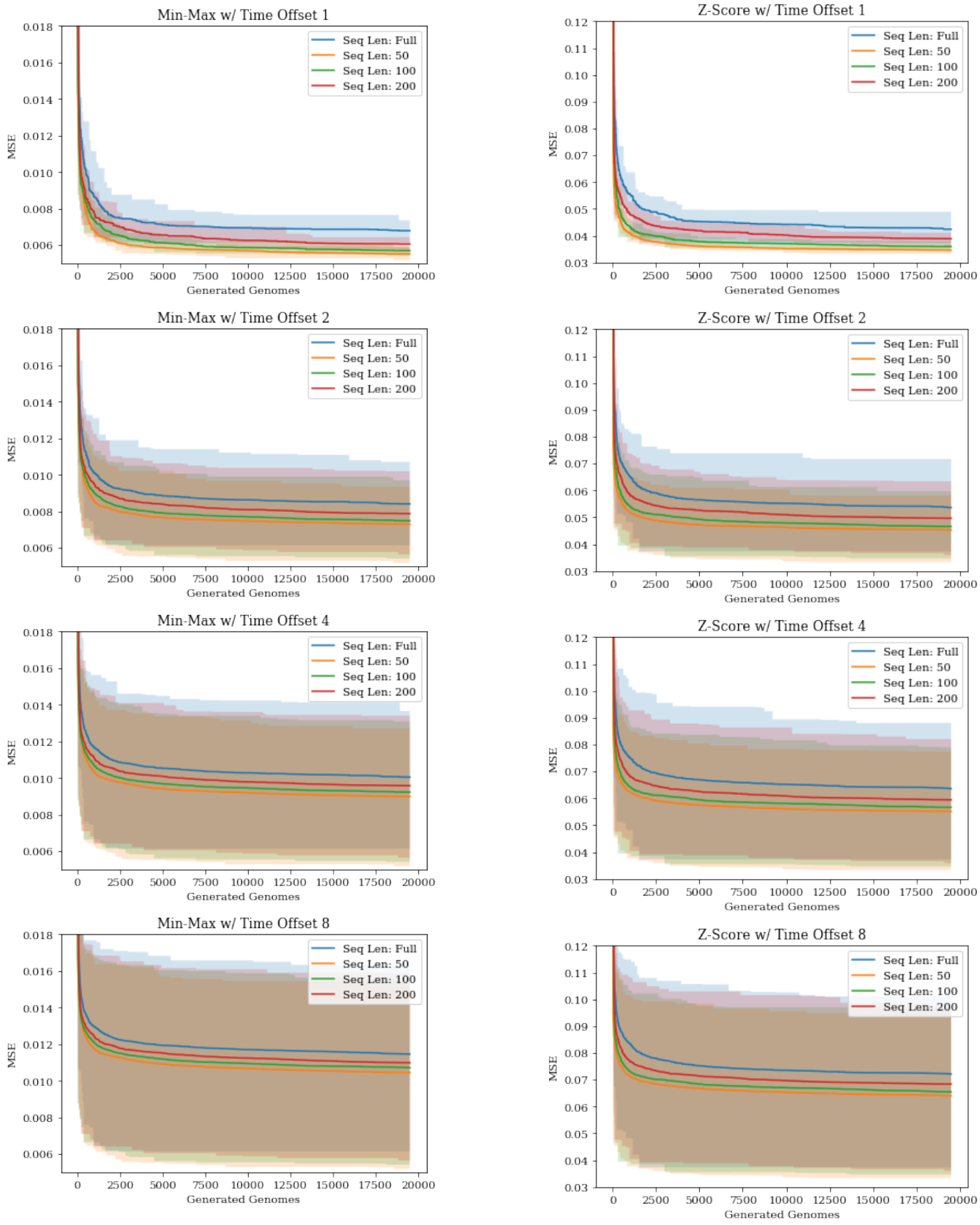

Figure 7: Secondary Superheater Temperature