Automated Design of Grey-Box Recurrent Neural Networks for Fault Diagnosis using Structural Models and Causal Information

Daniel Jung DANIEL.JUNG@LIU.SE
Department of Electrical Engineering, Linköping University, SE-581 83, Linköping, Sweden.

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

Behavioral modeling of nonlinear dynamic systems for control design and system monitoring of technical systems is a non-trivial task. One example is fault diagnosis where the objective is to detect abnormal system behavior due to faults at an early stage and isolate the faulty component. Developing sufficiently accurate models for fault diagnosis applications can be a time-consuming process which has motivated the use of data-driven models and machine learning. However, data-driven fault diagnosis is complicated by the facts that faults are rare events, and that it is not always possible to collect data that is representative of all operating conditions and faulty behavior. One solution to incomplete training data is to take into consideration physical insights when designing the data-driven models. One such approach is grey-box recurrent neural networks where physical insights about the monitored system are incorporated into the neural network structure. In this work, an automated design methodology is developed for grey-box recurrent neural networks using a structural representation of the system. Data from an internal combustion engine test bench is used to illustrate the potentials of the proposed network design method to construct residual generators for fault detection and isolation.

Keywords: Recurrent neural networks, physics-informed machine learning, fault diagnosis.

1. Introduction

This work is motivated by the problem of fault diagnosis of technical systems which considers monitoring the system health, including detection of occurring faults and isolating their root cause. One of the main principles of fault diagnosis is to detect inconsistencies between sensor data and predictions based on a model of the nominal system behavior, e.g., using residual generators. Two of the main fields are model-based diagnosis and data-driven diagnosis.

Model-based diagnosis uses a mathematical model of the system to be monitored derived from physical insights to construct residual generators. Residuals have some attractive properties, with respect to using the original sensor data as features, such as filtering out system dynamics while still being sensitive to faults. Fault detection and isolation is done by matching the residual patterns with different fault hypotheses. An advantage of model-based diagnosis, with respect to data-driven methods, is that it is possible to isolate unknown faults by using model analysis and sets of residual generators where the effects of different faults on the residual outputs are decoupled.

Developing sufficiently accurate models for residual generation is a time-consuming process, especially for complex or large-scale systems, and requires expert knowledge about the system to be modeled. This have motivated the use of data-driven modeling and machine learning to design diagnosis systems (Qin, 2012; Xu and Saleh, 2021). However, collecting representative training data for fault diagnosis applications is a difficult task. Many types of faults could occur in the system

and each type of fault can have many different realizations due to varying operating conditions and fault magnitudes (Jung et al., 2018). Therefore, training data is expected to be incomplete, e.g., due to unknown fault classes, which complicates the use of conventional multi-class classification approaches (Dong et al., 2017; Sankavaram et al., 2015).

Recurrent Neural Networks (RNN) are powerful black-box models that can capture the behavior of non-linear dynamic systems (Arsie et al., 2006). Neural networks have a flexible model structure making them useful in many different applications. However, a general drawback of general-purpose models, such as neural networks, is to find a sufficiently flexible model structure to model the behavior of a given system. This is often done based on the developer's experience and trial and error. In addition, if training data is only available from nominal operation, it is not trivial how to use data-driven models to identify the root cause of abnormal system behavior. Designing grey-box RNN models based on physical insights for residual generation has been proposed in, e.g., Pulido et al. (2019) and Jung (2019). In Jung (2019), a simulation study showed that residuals computed using grey-box RNN models, can isolate unknown fault classes. The solution is to select network structures that make the residuals only sensitive to faults in certain parts of the system. However, to make this useful for fault diagnosis in complex systems, tools are needed for systematic design of grey-box RNN models with desired robustness properties (sensitivity to different fault classes) even though training data from faults is limited.

The objective of this research is to develop data efficient machine learning models, with focus on Recurrent Neural Networks (RNN), for dynamic systems when training data are limited and not representative of all relevant data classes. Instead of trying to fit a general-purpose model structure to data, the aim is to use physical insights to derive a neural network structure that models the causal relations, which reduces the risk of overfitting, while at the same time is sufficiently flexible to model the dynamic behavior of the system. With limited training data, the goal is not only to maximize the model accuracy but also to make predictions robust to disturbances in the system, improve generalizability, and to be able to classify data from unknown classes.

1.1. Problem Statement

The focus in this work, is to develop an automated design process of grey-box RNN models for modeling the behavior of nonlinear dynamic systems based on a structural model representing causal information and physical insights about the system. The purpose is to systematically find neural network structures that model different parts of the system and capture the causal relationships between the input signals and the predicted output signal for residual generation. The motivation is to design data-driven residuals that can be used to detect and isolate faults even though training data from faults is not available.

Structural models are bi-partite graphs that represent the qualitative relationship between different model variables and can be used even though the analytical relationship is not completely known (Cassar and Staroswiecki, 1997). The usefulness of structural models in model-based diagnosis for fault diagnosis analysis and diagnosis system design, such as residual generation and sensor placement, has been proved in, e.g., Pulido and González (2004); Krysander et al. (2007); Frisk et al. (2017).

1.2. Related Research

The benefits of combining physical insights and machine learning to model dynamic systems have been discussed in, e.g., Brunton et al. (2016); Choudhary et al. (2020) and Pulido et al. (2019). The connections between different neural network structures and ordinary differential equations have been analyzed in, e.g., Lu et al. (2018) and Chen et al. (2018). Including physical insights about the system to design grey-box neural networks have been proposed in previous works, such as Pulido et al. (2019); Wu et al. (2016) and Hofmann et al. (2019). In Choudhary et al. (2020), Hamiltonian dynamics are incorporated in the neural network structure. The authors in Wang et al. (2021) propose a training method for grey-box non-linear dynamic models used to forecast COVID-19 dynamics. In Pizzuto and Mistry (2021), neural networks are trained using a physics-based loss function to model robotic systems. With respect to previous work, an automated design of grey-box RNN is developed which combines structural models and deep learning techniques.

Neural networks have been used for fault diagnosis in many different applications (Lei et al., 2020). In Bidarvatan et al. (2014), neural networks are used to develop a grey-box simulation model of an HCCI engine. A deep convolutional neural networks approach is proposed in Li et al. (2018) for prognostics applied to an aero-engine dataset. In Rahimilarki and Gao (2018), neural networks are used for fault diagnosis of a wind turbine. A hybrid system identification approach combining mathematical models and neural networks is proposed in Lu et al. (2019) where a weighted prediction is computed from the model-based and data-driven models. Grey-box RNN, based on state-space neural networks (Zamarreno et al., 2000), is proposed in Pulido et al. (2019) for residual generation to perform fault diagnosis of an evaporator in a sugar beet factory. With respect to previous works, a data-driven method for residual generation is proposed for isolation of unknown faults using grey-box RNN where the network structure is selected based on a structural model of the system.

2. Structural Analysis of Dynamic Systems

Structural analysis can be used to analyze fault diagnosis properties of complex non-linear dynamic systems and for systematic design of diagnosis systems (Frisk et al., 2017). A structural model $\mathcal{M} = (\mathcal{E}, \mathcal{X})$ is a bipartite graph describing the relationship between model equations $\mathcal{E} = \{e_1, e_2, ...\}$ and variables $\mathcal{X} = \{x_1, x_2, ...\}$, i.e., which variables are included in each model equation. The bipartite graph can be represented using a biadjacency matrix where an X in row (i, j) means that variable x_j is included in equation e_i. Model variables are partitioned into unknown variables, known variables, and fault signals that model how the different faults are affecting the system.

The rows and columns of the biadjacency matrix can be reorganized using the Dulmage-Mendelsohn (DM) decomposition (Dulmage and Mendelsohn, 1958) to analyze the structural redundancy properties of the system (Frisk et al., 2017). The DM decomposition partitions the biadjacency matrix into an under-determined part $\mathcal{M}^-$, an exactly determined part $\mathcal{M}^0$, and an over-determined part $\mathcal{M}^+$ as illustrated in Figure 1. The over-determined part $\mathcal{M}^+ = (\mathcal{E}^+, \mathcal{X}^+)$ has more equations than unknown variables and describes the part of the system that can be monitored using residual generators (Krysander et al., 2007). The degree of structural redundancy of a model $\mathcal{M}$ is defined as $\varphi(\mathcal{M}) = |\mathcal{E}^+| - |\mathcal{X}^+|$where $|\cdot|$ denotes set cardinality (Krysander et al., 2007). In general, a higher degree of redundancy means that there is more freedom to design residual generators.

Let e_{f_i} denote the model equation modeling where the fault f_i manifests in the system. A fault that enters the system somewhere modeled in the over-determined part, i.e., $e_{f_i} \in \mathcal{M}^+$, is said to

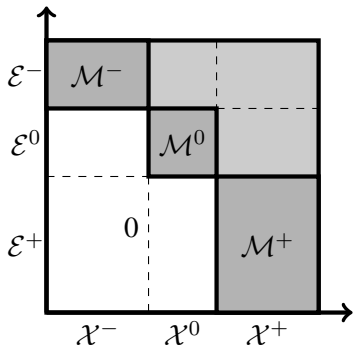

Figure 1: Dulmage-Mendelsohn decomposition of a model's biadjancency matrix.

be structurally detectable. Similarly, a fault f_i is said to be structurally isolable from fault f_j if $e_{f_i} \in \left(\mathcal{M} \setminus e_{f_j}\right)^+$, i.e., fault f_i is structurally isolable if e_{f_i} is still in the over-determined part when the equation e_{f_j} is removed from the model. In principle, structural fault detectability and isolability depend on if it is possible to design a residual generator modeling the part of the system where the fault f_i occurs or not. Over-determined sets that have a degree of redundancy equal to one, i.e., a set of equations where no subset have redundancy, are called Minimally Structurally Over-determined (MSO) sets (Krysander et al., 2007). An MSO set represents a minimal redundant equation set that can be used for residual generation. The MSO sets are interesting from a fault isolation perspective since they model a minimal part of the system that can be monitored using a residual.

2.1. Modeling Causal Relations Using Computational Graphs

If one equation is removed from an MSO set, the remaining set is exactly determined. This means that there is an equal number of unknown variables and equations. A matching algorithm can find a computational sequence that describes how to compute all unknown variables in the exactly determined set (Frisk et al., 2012). When all unknown variables have been computed, the redundant equation can be used as a residual equation.

The output from the matching algorithm can be represented by a computational graph. A computational graph is a directed graph where nodes either denote a variable or a function and edges show how the output of each node are fed as input to other nodes. The order of how the state variables is computed in the computational graph will determine its causality. If all the state variables are computed by integrating their derivatives, the computational graph is said to have integral causality. If the state variables are computed and then differentiated, the computational graph is said to have derivative causality. If the computational graph contains both states that are integrated and differentiated, it is said to have mixed causality (Frisk et al., 2012). Note that the one MSO set can be used to derive computational graphs with different causality depending on the matching, which is illustrated in the following example:

Example 1 *Consider the following MSO set*

$$e_1: \quad \dot{x}_1 = g_1(u) \quad e_2: \quad \dot{x}_2 = g_2(x_1) \quad e_3: \quad y = x_2 \quad e_4: \quad \dot{x}_1 = \frac{dx_1}{dt} \quad e_5: \quad \dot{x}_2 = \frac{dx_2}{dt} \tag{1}$$

where g_1 and g_2 are assumed invertible. When each of the equations e_1, e_2, and e_3, are selected as residual equation, respectively, the resulting computational graphs are given in Figure 2. The

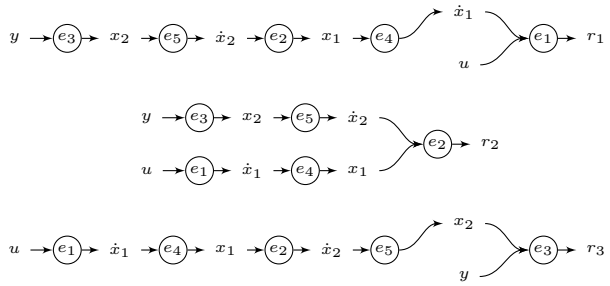

Figure 2: An illustration of three different computational graphs with different causalities using the same MSO set (1) but different residual equations.

computational graph when e_1 is used as residual equation has derivative causality, when graph e_2 is used has mixed causality, and integral causality when e_3 is used.

The computational graph gives the structural relationship between input and output variables and state variables, even though the analytical relationships are unknown. This provides useful information to incorporate physical insights and causal structure into the network structure of an RNN.

3. Residual Generation Using Grey-box Recurrent Neural Networks

Time-discrete non-linear state-space models can be modeled using RNNs. Here, computational graphs are derived from MSO sets with integral causality. These MSO sets are Differential Algebraic Equations (DAE), with an index equal to zero or one which means that they can be used to formulate residual generators in state-space form (Ascher and Petzold, 1998), i.e.,

$$\begin{aligned} \dot{x} &= \bar{g}(x, u) \\ r &= y - h(x, u) \end{aligned} \tag{2}$$

where x are state variables, u are known inputs to the RNN, y is the signal to predict, and $\bar{g} = (g_1, g_2, \ldots)^T$. Note that u could include both actuator and sensor signals depending on the computational graph. The arguments to each function $g_i(x, u) : \mathbb{R}^{|x|+|u|} \rightarrow \mathbb{R}$ are determined by backtracking from each corresponding state derivative $\dot{x}_i$ in the computational graph until a state variable x or an input signal u is found which give the arguments to each function $g_i(x, u)$. Similarly, the arguments to the function $h(x, u) : \mathbb{R}^{|x|+|u|} \rightarrow \mathbb{R}$ are determined by back-tracking from the residual equation in the computational graph until a state variable or and input signal is found.

When the arguments have been found, the time-continuous state-space model (2) can be formulated in discrete-time, using Euler forward, as

$$\begin{aligned} x_{t+1} &= x_t + T\,\bar{g}(x_t, u_t) \\ r_t &= y_t - h(x_t, u_t) \end{aligned} \tag{3}$$

where T is the sampling time.

Once the structure (3) of the discrete-time state space model is determined, an RNN is generated with the same structure (Jung, 2019). Here, the non-linear functions g_i and h are modeled as three fully connected layers where the last layer has a scalar output and the input vector to the first layer corresponds to the arguments in (3) derived from the computational graph.

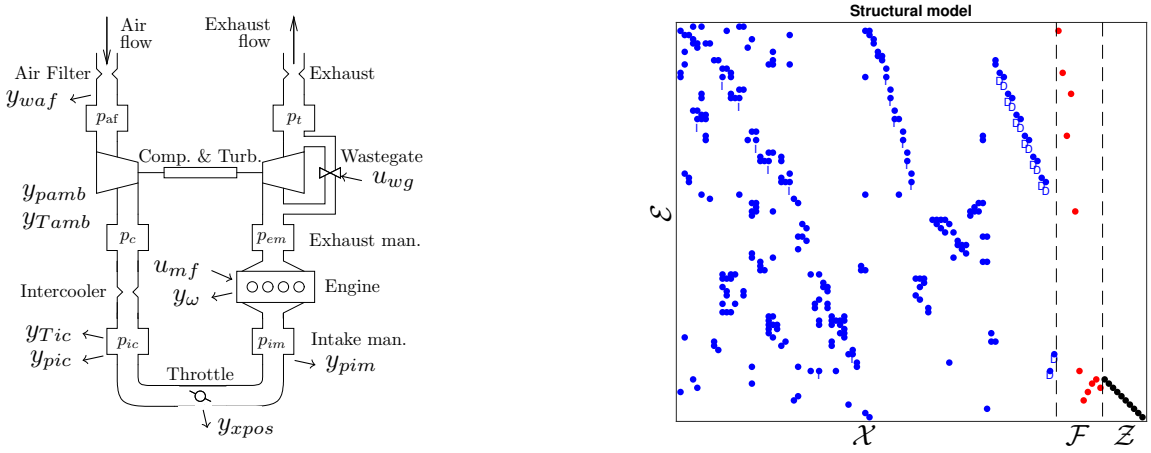

Figure 3: Left: A schematic of the model of the air flow through the engine (used with permission from Eriksson et al. (2002)). Right: Structural representation of the engine model.

4. Experiments

As a case study, the air path through an internal combustion engine is considered which is illustrated in Figure 3. The engine is an interesting case study because of its dynamic non-linear behavior and wide operating range. Due to the coupling between the intake and exhaust flow through the turbine and compressor, a fault somewhere in the system will not have an isolated impact in that component but is likely to affect the behavior in other parts of the system as well.

4.1. Model and Data

The structural model is based on a mathematical mean value engine model that has been used in previous works for model-based residual generation, see, e.g., Jung et al. (2018) and Ng et al. (2020). The mathematical model is similar to the model described in Eriksson (2007), which is based on six control volumes and mass and energy flows given by restrictions, see Figure 3.

A structural representation of the engine model is shown in the right plot in Figure 3 where a mark in position (i, j) denotes that the variable x_j is included in equation e_i. The model variables are organized in unknown variables $\mathcal{X}$ (including state variables), known variables $\mathcal{Z}$, including known actuators and sensor outputs, and fault signals $\mathcal{F}$. To clarify the relation between state variables and their derivatives in the structural model, state variables are marked as I and their derivatives as D in the figure (Frisk et al., 2017). The structural model has 94 equations, 90 unknown variables, including 14 state variables and their derivatives, 11 fault variables, and 10 known variables.

Operational data for training and validation has been collected from the engine test bench during transient operation. To cover a large range of operating conditions, data are collected from the engine when it follows the Worldwide Harmonized Light Vehicles Test Procedure (WLTP) cycle.

The set of available sensors in the engine corresponds to a standard setup used in a conventional car (Jung et al., 2018) and the signals are sampled in 20Hz to capture the dynamics of the engine. The known variables include sensors measuring pressures (y_{pic}, y_{pim}, y_{pamb}), temperatures (y_{Tic}, y_{Tamb}), air mass flow (y_{waf}), injected fuel (u_{mf}), throttle position (y_{xpos}), and engine speed (y_ω),

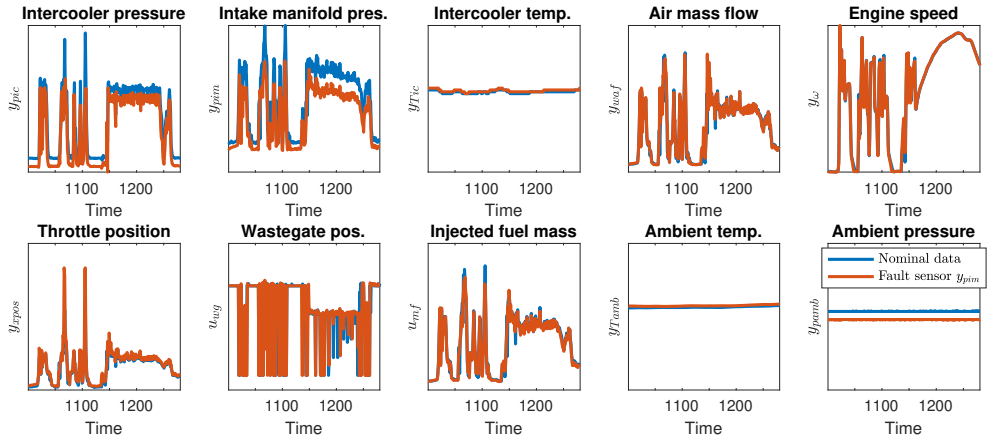

Figure 4: Example of engine data from fault-free and faulty operation when there is a fault in sensor y_{pim}. Note that the sensor fault affects the system behavior and is visible in other signals as well, e.g., y_{pic}.

and also, the wastegate actuator signal (u_{wg}), see Figure 3. An example of signal outputs from nominal and faulty operation (fault in sensor y_{pim}) is shown in Figure 4. Each signal is normalized such that nominal signal output is in the range of about [0, 1].

Experimental data have been collected from three different sensor faults ($f_{ypic}, f_{ypim}, f_{ywaf}$) and a leakage before the throttle (f_{iml}). Multiplicative sensor faults of different magnitudes have been injected by modifying the sensor signal directly in the engine control unit during operation. Each fault scenario represents a constant fault magnitude, but the impact of each fault varies with the operating conditions of the system.

4.2. Recurrent Neural Networks

An artificial neural network models the relationship between a set of inputs $u \in \mathbb{R}^{n_u}$ and outputs $y \in \mathbb{R}^{n_y}$ using a computational graph. Each node represents a non-linear operation on the weighted sum of the node inputs, x_{in}, usually in the form $x_{out} = g(a^T x_{in} + b)$. The non-linear function $g(\cdot)$ is called an activation function, a is a vector of weights, and b is a bias term. Some examples of common activation functions are the rectified linear unit (ReLU), $g(\xi) = \max(0, \xi)$, and different sigmoid functions, such as the logistic function or arc tangent function (Aggarwal, 2018).

Recurrent neural networks are used to model dynamic systems and time-series data. Internal states are modeled in the RNN by duplicating the network for each time instance and add connections in some nodes between different time steps, similar to state variables in a state-space model (Aggarwal, 2018). The model structure of neural networks is commonly designed such that the nodes are organized in different layers where the inputs to each node are the output of nodes in the previous layer and the nodes in the same layers have the same activation function. The first layer is referred to as the input layer, where data is fed into the neural network. The last layer is the output layer, which returns the output from the neural network model. The in-between layers are referred to as hidden layers.

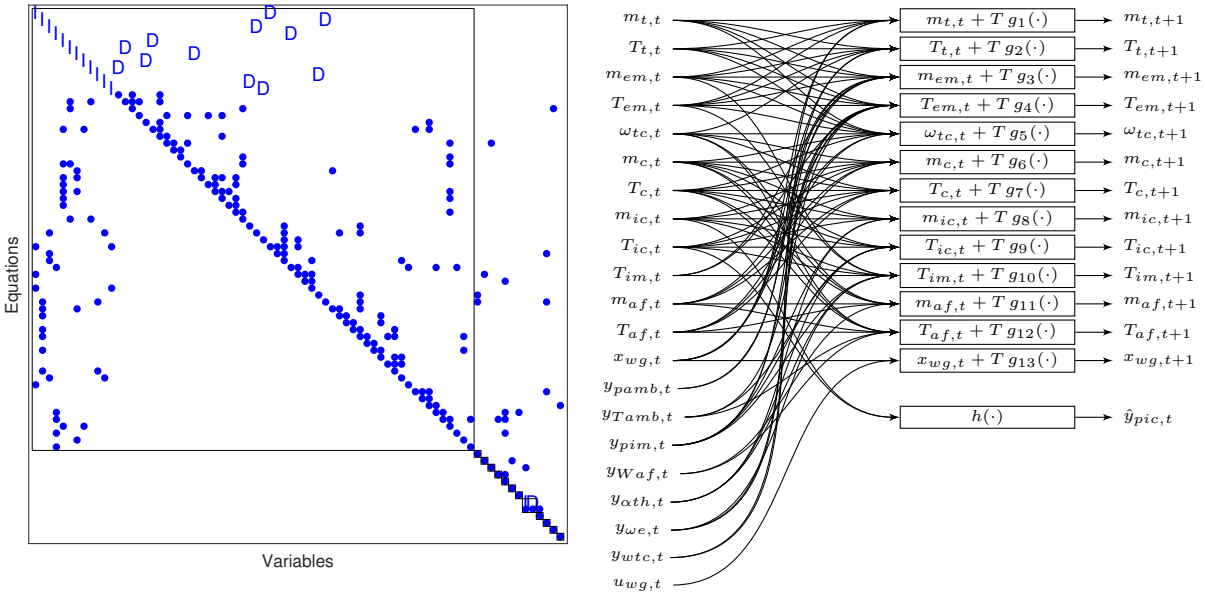

Figure 5: Left: The matching of MSO_{27} that is used to derive the computational graph. Starting from the lowest row, each unknown variable on the diagonal can be computed based on previously computed unknown variables and state variables. Right: Schematic of the resulting grey-box RNN model that is used to estimate the sensor output $\hat{y}_{pic,t}$.

4.3. Residual Generation

Based on the structural model, a set of 144 MSO sets are identified (Frisk et al., 2017). A causality analysis is performed on each of the 144 candidate MSO sets that identified 17 MSO sets that can be used to generate computational graphs with integral causality. Out of these 17 MSO sets, 21 different computational graphs with integral causality are generated. The subset of computational graphs where the sensor equations modeling y_{pim}, y_{pic}, or y_{waf}, are used as residual equation, is used to generate grey-box RNN models. The other computational graphs have residual equations based on sensors measuring slowly varying states, such as temperatures, that do not show sufficient excitation in training data, see Figure 4, and are not considered in this case study.

The computational graph of MSO_{27}, which is MSO set number 27, is derived from the matching shown to the left in Figure 5 going down-to-up where each variable in the diagonal can be computed based on the variables marked on the same row. Rows with I on the diagonal denotes computation of state variables by integrating their derivatives. The resulting grey-box RNN model is shown to the right in Figure 5. The substructures in the grey-box RNN model representing the non-linear functions $g_i(\cdot)$ and $h(\cdot)$ are modeled using a three fully connected layer structure and a scalar output. The dimension of the input layer is determined from the computational graph. Different activation functions and number of nodes in each layer have been evaluated where ReLU and 256 nodes in each layer gave the overall best results and are therefore used in all grey-box RNN models in this case study.

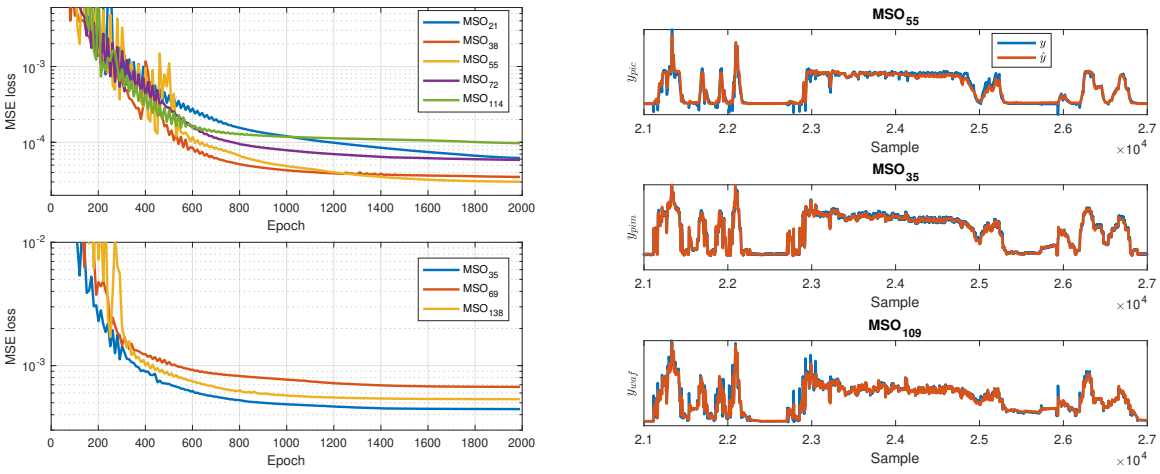

Figure 6: Left: The upper plot shows the loss after each epoch when training neural networks predicting y_{pic} and the lower plot shows the loss when training neural networks predicting y_{pim}. Right: The prediction performance of three of the grey-box RNN models.

4.4. Training

Each neural network is trained using only fault-free data where the time series data is partitioned into a set of 19 equally sized batches of 600 samples covering different operating conditions. Training is done in Python, using PyTorch (Paszke et al., 2017), by minimizing the mean square prediction error $\frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2$. The initial values of the state variables are unknown and are set to some reasonable reference value that is used for all batches. The optimization algorithm ADAM (Diederik and Kingma, 2015) is run for 2000 epochs for each model with an adaptive learning rate starting at $5 * 10^{-4}$ which is reduced every 10th epoch by 3%. The loss after each epoch is shown in Figure 6 for grey-box RNN models predicting $\hat{y}_{pic}$ and grey-box RNN models predicting $\hat{y}_{pim}$.

4.5. Evaluation

Predictions from three grey-box RNN models are shown in the right plot in Figure 6. It is visible that the different grey-box RNN models capture the general dynamic behavior of the different sensor signals. Similar prediction performance is achieved for the remaining grey-box RNN models. The residual outputs have a small bias in the datasets that are likely to be caused by incorrect initial conditions of the state variables. If it is assumed that there are no faults affecting the residual output when a scenario begins, the residual bias is removed by subtracting the median of the residual output computed from a short time interval in the beginning of each dataset.

Fault detection performance using the RNN models for residual generation is evaluated using data from different fault realizations. The left plot in Figure 7 shows four of the residuals plotted as histograms comparing different fault scenarios and nominal operation. The Receiver Operating Characteristics (ROC) curve is computed for different residuals and fault magnitudes. The right plot in Figure 7 shows detection performance by plotting the area under the curve (AUC) as a function of fault size. Multiplicative sensor faults are evaluated for magnitudes in the range [-20%, 20%]. The AUC values are normalized as $2 \cdot (\text{AUC} - 0.5)$ where a value close to zero means that the fault is not

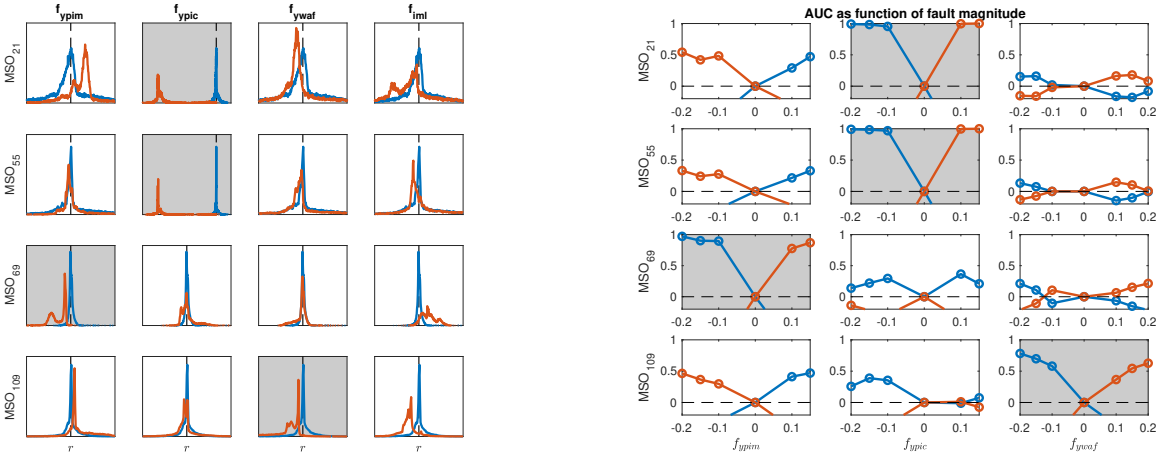

Figure 7: Left: Histograms of residual outputs during different fault scenarios. The blue curve represents the residual output during fault-free case and the red curve when a fault is present. Right: Normalized AUC as a function of fault size. The highlighted subfigures show scenarios when a sensor fault is affecting the sensor signal that is predicted by the grey-box RNN.

affecting the residual output. Cases when the sensor fault is affecting the predicted sensor signal in each corresponding residual are highlighted in gray. It is visible that these faults are easiest to detect by each residual. Some faults have little or no impact on the residual outputs, e.g., the residuals based on MSO_{21}, MSO_{55}, and MSO_{69} are not sensitive to fault f_{ywaf} since AUC is almost zero for all fault magnitudes.

5. Concluding Remarks

This paper describes how techniques used in, e.g., model-based fault diagnosis, are useful for data-driven modeling using physical insights. The proposed methodology can be used to automatically construct grey-box RNN models from structural models to model nonlinear dynamic systems. An advantage of the proposed method is that the neural network structure is derived based on physical insights about the system and is able to capture the causal relations between signals when training data from different faults is limited. The results from the engine case study show that the generated grey-box RNN models can capture the dynamic behavior of the engine and diagnose unknown faults. As future work, other relevant applications should be investigated are variable selection to predict a certain sensor output, and design of predictive models that are robust to disturbances even though the models are trained using only nominal training data.

References

C. Aggarwal. Neural networks and deep learning. *Springer*, 10:978–3, 2018.

I. Arsie, C. Pianese, and M. Sorrentino. A procedure to enhance identification of recurrent neural networks for simulating air–fuel ratio dynamics in si engines. *Engineering Applications of Artificial Intelligence*, 19(1):65–77, 2006.

U. Ascher and L. Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations*, volume 61. Siam, 1998.

M. Bidarvatan, V. Thakkar, M. Shahbakhti, B. Bahri, and A. Aziz. Grey-box modeling of hcci engines. *Applied Thermal Engineering*, 70(1):397–409, 2014.

S. Brunton, J. Proctor, and J N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113 (15):3932–3937, 2016.

J.P. Cassar and M. Staroswiecki. A structural approach for the design of failure detection and identification systems. In *IFAC Conference on Control of Industrial Systems (Control for the Future of the Youth), Belfort, France, 20-22 May*, volume 30, pages 841 – 846, 1997.

T. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

A. Choudhary, J. Lindner, E. Holliday, S. Miller, S. Sinha, and W. Ditto. Physics-enhanced neural networks learn order and chaos. *Physical Review E*, 101(6):062207, 2020.

J. Diederik and P. Kingma. Adam: A method for stochastic optimization. In *3rd international conference for learning representations, San Diego*, 2015.

L. Dong, L. Shulin, and H. Zhang. A method of anomaly detection and fault diagnosis with online adaptive learning under small training samples. *Pattern Recognition*, 64:374–385, 2017.

A. Dulmage and N. Mendelsohn. Coverings of bipartite graphs. *Canadian Journal of Mathematics*, 10:517–534, 1958.

L. Eriksson. Modeling and control of turbocharged si and di engines. *OGST-Revue de l'IFP*, 62(4): 523–538, 2007.

L. Eriksson, S. Frei, C. Onder, and L. Guzzella. Control and optimization of turbo charged spark ignited engines. In *IFAC World Congress*, 2002.

E. Frisk, A. Bregon, J. Aslund, M. Krysander, B. Pulido, and G. Biswas. Diagnosability analysis considering causal interpretations for differential constraints. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 42(5):1216–1229, 2012.

E. Frisk, M. Krysander, and D. Jung. A toolbox for analysis and design of model based diagnosis systems for large scale models. *IFAC-PapersOnLine*, 50(1):3287–3293, 2017.

R. Hofmann, V. Halmschlager, M. Koller, G. Scharinger-Urschitz, F. Birkelbach, and H. Walter. Comparison of a physical and a data-driven model of a packed bed regenerator for industrial applications. *Journal of Energy Storage*, 23:558–578, 2019.

D. Jung. Isolation and localization of unknown faults using neural network-based residuals. In *Proceedings of the Annual Conference of the PHM Society*, volume 11, 2019.

D. Jung, K. Ng, E. Frisk, and M. Krysander. Combining model-based diagnosis and data-driven anomaly classifiers for fault isolation. *Control Engineering Practice*, 80:146–156, 2018.

M. Krysander, J. Åslund, and M. Nyberg. An efficient algorithm for finding minimal overconstrained subsystems for model-based diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 38(1):197–206, 2007.

Y. Lei, B. Yang, X. Jiang, F. Jia, N. Li, and A. Nandi. Applications of machine learning to machine fault diagnosis: A review and roadmap. *Mechanical Systems and Signal Processing*, 138:106587, 2020.

X. Li, Q. Ding, and J. Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11, 2018.

L. Lu, Y. Tan, D. Oetomo, I. Mareels, E. Zhao, and S. An. On model-guided neural networks for system identification. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 610–616, 2019.

Y. Lu, A. Zhong, Q. Li, and B. Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, pages 3276–3285, 2018.

K. Ng, E. Frisk, and M. Krysander. Design and selection of additional residuals to enhance fault isolation of a turbocharged spark ignited engine system. In *7th International Conference on Control, Decision and Information Technologies (CoDIT'20)*. IEEE, 2020.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

G. Pizzuto and M. Mistry. Physics-penalised regularisation for learning dynamics models with contact. In *Learning for Dynamics and Control*, pages 611–622. PMLR, 2021.

B. Pulido and C. González. Possible conflicts: a compilation technique for consistency-based diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(5): 2192–2206, 2004.

B. Pulido, J. Zamarreño, A. Merino, and A. Bregon. State space neural networks and model-decomposition methods for fault diagnosis of complex industrial systems. *Engineering Applications of Artificial Intelligence*, 79:67–86, 2019.

S. Qin. Survey on data-driven industrial process monitoring and diagnosis. *Annual reviews in control*, 36(2):220–234, 2012.

R. Rahimilarki and Z. Gao. Grey-box model identification and fault detection of wind turbines using artificial neural networks. In *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, pages 647–652. IEEE, 2018.

C. Sankavaram, A. Kodali, K. Pattipati, and S. Singh. Incremental classifiers for data-driven fault diagnosis applied to automotive systems. *IEEE Access*, 3:407–419, 2015.

R. Wang, D. Maddix, C. Faloutsos, Y. Wang, and R. Yu. Bridging physics-based and data-driven modeling for learning dynamical systems. In *Learning for Dynamics and Control*, pages 385–398. PMLR, 2021.

Z. Wu, J. Li, M. Cai, Y. Lin, and W. Zhang. On membership of black-box or white-box of artificial neural network models. In *2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, pages 1400–1404, 2016.

Z. Xu and J. Saleh. Machine learning for reliability engineering and safety applications: Review of current status and future opportunities. *Reliability Engineering & System Safety*, page 107530, 2021.

J. Zamarreno, P. Vega, L. Garcıa, and M. Francisco. State-space neural network for modelling, prediction and control. *Control Engineering Practice*, 8(9):1063–1075, 2000.