processes

Article

Deep Transfer Learning for Approximate Model Predictive Control

Samuel Arce Munoz, Junho Park, Cristina M. Stewart, Adam M. Martin and John D. Hedengren *

Department of Chemical Engineering, Brigham Young University, Provo, UT 84602, USA
* Correspondence: john_hedengren@byu.edu

Abstract: Transfer learning is a machine learning technique that takes a pre-trained model that has already been trained on a related task, and adapts it for use on a new, related task. This is particularly useful in the context of model predictive control (MPC), where deep transfer learning is used to improve the training of the MPC by leveraging the knowledge gained from related controllers. One way in which transfer learning is applied in the context of MPC is by using a pre-trained deep learning model of the MPC, and then fine-tuning the controller training for a new process automation task. This is similar to how an equipment operator quickly learns to manually control a new processing unit because of related skills learned from controlling the prior unit. This reduces the amount of data required to train the approximate MPC controller, and also improves the performance on the target system. Additionally, learning the MPC actions alleviates the computational burden of online optimization calculations, although this approach is limited to learning from systems where an MPC has already been developed. The paper reviews approximate MPC formulations with a case study that illustrates the use of neural networks and transfer learning to create a multiple-input multiple-output (MIMO) approximate MPC. The performance of the resulting controller is similar to that of a controller trained on an existing MPC, but it requires less than a quarter of the target system data for training. The main contributions of this paper are a summary survey of approximate MPC formulations and a motivating case study that includes a discussion of future development work in this area. The case study presents an example of using neural networks and transfer learning to create a MIMO approximate MPC and discusses the potential for further research and development in this area. Overall, the goal of this paper is to provide an overview of the current state of research in approximate MPC, as well as to inspire and guide future work in transfer learning.

Keywords: deep learning; transfer learning; model predictive control; approximate model predictive control

check for updates

Citation: Arce Munoz, S.; Park, J.; Stewart, C.M.; Martin, A.M.; Hedengren, J.D. Deep Transfer Learning for Approximate Model Predictive Control. *Processes* **2023**, *11*, 197. https://doi.org/10.3390/pr11010197

Academic Editor: Xiong Luo

Received: 13 December 2022
Revised: 31 December 2022
Accepted: 4 January 2023
Published: 7 January 2023

1. Introduction

Model predictive control (MPC) requires the development of a dynamical mathematical model and the solution of optimization subroutines in real time. These requirements represent a challenge because industrial processes are often too complex to be modeled from first principles, and first-principles models, when available, can be computationally too expensive to be solved in real time. System identification through statistical methods and historical data can expedite the development of a dynamic model when a first-principles formulation is not available [1]. While a linear model can often be solved in real time, many industrial processes exhibit nonlinear dynamics, and linear MPC becomes insufficient. This has motivated the use of deep neural networks as a non-linear method for process control.

Deep-learning architectures designed to handle sequential data, such as recurrent neural networks (RNNs), long short-term memory (LSTM) networks, and transformers, are responsible for remarkable results in the tasks of natural language processing [2], speech recognition [3], and, more recently, protein folding [4]. These networks are also being used for system identification and model predictive control of evaporation processes [5], pharmaceutical manufacturing [6] and reactor control [7]. Although deep neural networks are capable of capturing complex dynamics, they represent a challenge when they are used to calculate optimal control actions. As noted by Schweidtmann et al. [8], the consideration of complex data-driven models has long been limited to local or stochastic solution approaches [9–11], or fit-for-purpose solutions limited to low-dimensional problems and state-of-the art solvers [12,13]. While these approaches focus on using the deep neural network model as part of the optimization routine, other researchers have focused on using the deep learning models to generate explicit, or approximate MPC formulations [14–17]. These architectures, however, depend on the development of an MPC in the first place so that enough training data are generated.

This study contributes to the development of improved approximate MPC formulations by using deep learning models and transfer learning, a research area in the field of machine learning that aims to transfer the knowledge learned from one problem to a similar problem.

The contributions of this paper are as follows:

- A non-comprehensive survey of the work done on approximate MPC formulations.
- A case study on a multiple-input multiple-output (MIMO) system to show how deep learning and transfer learning can be used to develop an approximate MPC controller.

1.1. Previous Work on Approximate MPC Formulations

Table 1 summarizes a non-comprehensive survey on approximate MPC formulations. A large portion of the research done in this area has focused on decreasing the computational cost of MPC formulations, applied to equivalent systems. Alessio and Bemporad [18], for example, summarize different approaches to solving the MPC optimization problem offline for a given range of operating conditions applied on a system of interest, focusing mostly on piecewise affine functions. Alessio and Bemporad [18] also provide a novel algorithm that reduces the complexity of explicit MPC. In their approach, extensive off-line simulations of closed-loop MPC collect the most recurrent combinations of active constraints and generate critical regions. These critical regions are used to calculate or extrapolate a suboptimal input move in real time.

Table 1. A non-comprehensive survey on explicit MPC, MPC emulators and approximate MPC.

Articles	MPC Informed for Equivalent Systems	Equivalent System with Neural Network	Transfer Learning for Similar Systems	Transfer Learning for Unique Systems
Explicit MPC: 329 Articles, 22 with Neural Networks or Deep Learning				
Tøndel et al. [19]	x			
Alessio and Bemporad [18]	x			
Csekő et al. [20]	x	x		
Grosso et al. [21]	x	x		
Katz et al. [22]	x	x		
Chen et al. [23]	x	x		
MPC Emulation: 357 Articles, 10 with Neural Networks or Deep Learning				
Zheng et al. [24]	x			
Moness and Moustafa [25]	x			
Wang et al. [26]	x	x		
Yan and Wang [27]	x	x		
Novak and Dragicevic [28]	x	x		

Table 1. *Cont.*

Articles	MPC Informed for Equivalent Systems	Equivalent System with Neural Network	Transfer Learning for Similar Systems	Transfer Learning for Unique Systems
Approximate MPC: 24 Articles, 12 with Transfer Learning				
Hofer et al. [29]	x			
Pin et al. [30]	x	x		
Yang et al. [16]	x	x		
Gan et al. [31]	x	x		
Wang et al. [32]		x	x	
Wang and Hong [33]	x	x	x	
Chen et al. [34]	x	x	x	
Han et al. [35]	x	x	x	
Chen et al. [36]	x	x	x	
This study	x	x	x	x

Grosso et al. [21] explore the use of neural networks to analyze complex drinking water networks (DWNs). In their approach, the authors provide a novel quasi-explicit MPC formulation, where heavy computational tasks are replaced by non-linear modules using neural networks, and the prediction horizon and weight matrices of the MPC are continuously adjusted based on a trade-off between risk and economic cost. This allows for the effective scaling of MPC to large, multi-objective applications. In the work of Tøndel et al. [19], polyhedral partitions of the state space are used as a novel approach to improve the efficiency of explicit solutions, avoiding unnecessary partitioning and QP problem solving. Similarly, Csekő et al. [20] propose the use of radial basis function (RBF)-based neural networks as an approximation of the MPC controller and compare that to a Kalman filter applied to an explicit MPC, showing that the neural network approximation can provide fast control but at the expense of deteriorated performance. Additionally, Katz et al. [22] show a novel approach where deep neural networks approximate the non-linear parts of a multiparametric mixed integer linear programming (mpMILP), making the formulation tractable while maintaining the linear aspects components of the optimization problem, such as box constraints.

Other examples include the work of Moness and Moustafa [25], where a switched model predictive controller (SMPC) is used in a physical wind turbine emulator, and the work of Zhu et al. [37], where an improved finite-set MPC is proposed to achieve simplified control structure and faster dynamic response in an islanded alternating current microgrid. In both cases, the novel use of simplified MPC architectures that rely on switching between control strategies from a pool of prediction models result in faster controller response. Similarly, Novak and Dragicevic [28] use a finite-set MPC formulation for power electronics, but with the introduction of a supervised learning approach to imitate the controller. This contribution further alleviates the computational burden of finite-set MPC formulations by introducing an approximate MPC that is independent of the non-linearity of the system.

Examples of deep learning-based approximations of MPC include the work of Wang et al. [26], where a machine learning (ML)-based emulation of an MPC is used for modular multilevel converters, and the work of Yan and Wang [27], where recurrent neural networks (RNNs) and a collective neurodynamic optimization approach are used to create non-linear model predictive control (NMPC) without linearization. The novel neurodynamic approach presented by [27] further augments the use of deep learning-based MPC formulations by improving the starting points of each RNN using the information of global and local best known solutions and a particle swarm optimization approach. This approach effectively combines global search and precise local search capabilities, and converges to global optimal solutions in simulation studies.Other examples include the work of Pin et al. [30], where a NN is used to capture the non-linearities of a discrete-time NMPC model with hard constraints for similar performance in offline control. Similarly, Hofer et al. [29] manipulate an approximate MPC to mirror performance of a full MPC while lowering the computational cost of the onboard control of unmanned aerial vehicles. Their novel method uses a continuous time parametrization of the state and input trajectory to derive a compact description of the optimal control problem, which outperforms a linear quadratic regulation approach in simulation studies.

Regarding the use of transfer learning for approximate control formulations, an offline approximate MPC from Gan et al. [31] controls a hybrid energy ship using a random forest model, performing nearly as well as the MPC policy used to train the model. Han et al. [35] build a knowledge-data-driven MPC using fuzzy NN for waste-water treatment with transfer learning mechanisms to adapt to similar systems. Their novel transfer learning mechanism improves the control performance by integrating state information of the current model with historical model data to offset the data shortage in the learning process. This transfer of knowledge of model parameters has the potential to benefit other water-related applications where deep learning has shown promising results but collecting data is difficult, such as intermittent streamflow predictions in arid regions [38–40].

Similarly, Wang and Hong [33] offer a comprehensive review of building control using reinforcement learning and transfer learning to imitate the performance of MPC. This contributions are valuable in applications where MPC has proven very useful but the expertise required for each application limits its scalability [41]. In a successful example, Chen et al. [34] inform control of a room in Shanghai with limited data using layers from a NN model trained intensively on a room in Beijing. In their approach, a deep neural network with more than 40,000 trainable parameters was pre-trained with multi-year data from the room in Beijing and re-trained with only 15 days of data from the room in Shanghai, resulting in high-quality predictions of the air temperature and relative humidity of the room in Shangai. In other publications, Yang et al. [16] train a RNN with a nonlinear autoregressive exogenous model structure on data from test rooms controlled with MPC, reporting significant improvement from original control, Chen et al. [36] captures MPC behavior for building control in a RL model which then informs models for other buildings. The performance of their learning application improves during operation, reinforced with additional data.

As described in this survey, the research performed on approximate MPC formulations has focused primarily on reducing the computational time of solving the optimization problem. An important aspect of this study is that it presents an approximate MPC that not only reduces the computational time of a traditional MPC formulation, but that it is also designed to learn from different representations of knowledge. This is similar to how an operator can quickly learn to manually control a new processing unit due to their prior experience and skills gained from controlling a similar unit. In other words, the operator’s previous experience with a related task enables them to more easily adapt to and perform the new task. This concept is similar to how transfer learning in machine learning allows a model that has been trained on a related task to be adapted and fine-tuned for use on a new, related task, thereby reducing the amount of training data and improving performance. For example, deep learning facilitates learning from complex systems, such as the decision criteria followed by an operator that is manually controlling a process. Additionally, transfer learning facilitates learning from systems with similar dynamics, similar to how an operator can learn to control a new process by using prior knowledge.

1.2. Transfer Learning

Transfer learning is a research area in the field of machine learning that aims to transfer the knowledge learned from one problem to a similar problem. As noted by Yang et al. [42], transfer learning is rooted in AI, psychology, educational theory and cognitive science, and it has been addressed through frameworks, such as analogical problem solving [43,44] and high-level structural similarity in analogical reasoning [45]. Foundational work by Thrun and Pratt [46] introduces the term *learning to learn*, and proposes methods that can change the way machine learning algorithms generalize. These methods are compared to how humans learn to generalize from exposure to a continual stream of learning tasks. According to Thrun and Pratt [46], a machine learning algorithm is capable of *learning to learn* (i.e., generalize) if its performance at each task improves with experience and with the number of tasks, given a family of tasks, training experience for each of these tasks and a family of performance measures.

Transfer learning can be formally defined with the notations introduced by Pan and Yang [47]. A domain $\mathbb{D}$, consists of a feature space $\mathcal{X}$ and marginal probability distribution $\mathbb{P}^X$, where each input instance $\mathbf{x} \in \mathcal{X}$. Given a specific domain, $\mathbb{D} = \{\mathcal{X}, \mathbb{P}^X\}$, a task $\mathbb{T}$ has two components: a label space $\mathcal{Y}$ and a function $f(\cdot)$ (denoted by $\mathbb{T} = \{\mathcal{Y}, f(\cdot)\}$). The function $f(\cdot)$ is a function that can make predictions on unseen instances $\mathbf{x}^*$s.

In a two-domain scenario, there is one source domain $\mathbb{D}_s$ and one target domain $\mathbb{D}_t$, where the *source domain labeled data* are denoted by $\mathcal{D}s = \{(\mathbf{x}_{s_i}, y_{s_i}\}_{i=1}^{n_s}$, where $\mathbf{x}_{s_i} \in \mathcal{X}_s$ is the data instance and $y_{s_i} \in \mathcal{Y}_s$ is the corresponding class label. Similarly, $\mathcal{D}_t = \{(\mathbf{x}_{s_i}, y_{s_i}\}_{i=1}^{n_s}$ is denoted as the *target domain labeled data*, where the input $\mathbf{x}_{t_i} \in \mathcal{X}_t$ is the input, and $y_{t_i} \in \mathcal{Y}_t$.

Transfer learning has shown success in various fields of study. In the work of [48] a pre-trained network was found to best evaluate body posture as an indicator of sleep quality. Utilizing a transfer learning approach, researchers successfully identified four predefined sleep poses during 12 participants' overnight sleep episodes. When assessed against manually evaluated sleep positions, the pre-trained network using transfer learning outperformed other pre-trained networks. While the standard position sensor was estimated to have 88.2% accuracy, their deep learning video scoring stood at 95.1% accuracy.

Transfer learning has also been used to improve swimming pool safety by using computer vision to produce automated surveillance systems that might identify drowning objects, as applied to a motion detection sensor. Alotaibi [49] utilizes a specialized neural network based on a ResNet-50 model. Researchers successfully enhanced the performance of the proposed system, overcoming the limitations of a small available data size. While a DCNN learning model achieved 95% accuracy upon evaluation, the specialized model was found to have 99% accuracy.

In the field of process control, transfer learning has been used as an approximation of a system with complex dynamics where data are scarce. In the work of Chen et al. [34] for example, a neural network model with 42,902 parameters is used to learn the dynamics of natural ventilation in a building, based on multi-year data from a source system. The model is then retrained with only 200 trainable parameters and only 15 days of data from the target system, which has different floor area, building material, and window size. The resulting model shows a reduced mean squared error, almost one order of magnitude smaller than the models that are only trained on source data or target data. A similar approach is presented by Wang et al. [32], where a growing deep belief network (GDBN) network is used to identify the dynamics of a continuous stirred-tank reactor (CSTR). In their work, transfer learning is used to dynamically optimize the structure of a GDP, by using the pre-trained layers of an initialized deep belief network (DBN) to help the GDBN automatically determine its optimal size and achieve high performance in system identification.

Although both examples show promising results on the use of transfer learning for system identification, they still depend on an efficient MPC to control the system. This article shows a case study in which a deep neural network is used to identify the control policy of a source system, similar to the work discussed on deep learning-based explicit MPC, and deep learning-based MPC emulators, and deploy on a target system. Transfer learning, in this case, is used to enhance the performance of the approximate MPC on the target system, without the need for additional steps to identify the dynamics of the new system or solving an optimization subroutine to calculate the controller outputs.

As noted by Tan et al. [50], deep learning models are *data-dependent*, and they require large amounts of data for training and testing. Ideally, the training and test data must be independent and identically distributed (i.i.d) to accurately assess the predictive capabilities of the model. However, in many situations, it is difficult to collect large amounts of high quality data for deep learning. In these situations, transfer learning is attractive because it relaxes the assumption of i.i.d. training and data sets, suggesting that a model can be pre-trained on a training set and fine tuned on a target test set belonging to a different domain.

Tan et al. [50] further categorize transfer learning into the following:

- Instances-based;
- Mapping-based;
- Adversarial-based;
- Network-based.

An *instances-based* transfer learning approach takes instances from a source domain that can be used in a target domain by adjusting the associated weights. This approach has been successful in text classification applications [51].

In a *mapping-based* approach, the two domains are combined in a new data set and fed to a union network. Matasci et al. [52] use this approach to improve image classification in a remote sensing application.

An *adversarial-based* approach finds patterns that are applicable to the source and target domain, while discriminating between the source and target tasks. Li et al. [53] use this approach in a fault diagnosis application, where plant data are scarce but simulation data are easy to generate.

Finally, a *network-based* approach reuses parts of a network that has been pre-trained with source domain data and learns from the target domain data. As an example, Chen et al. [34] use this approach to train an MPC for smart-building control where training data are scarce but a model pre-trained on a similar building is available. This study presents a *network-based* approach, pre-training a neural network with data from a source system, and fine-tuning with data from a target system, as shown in Figure 1.

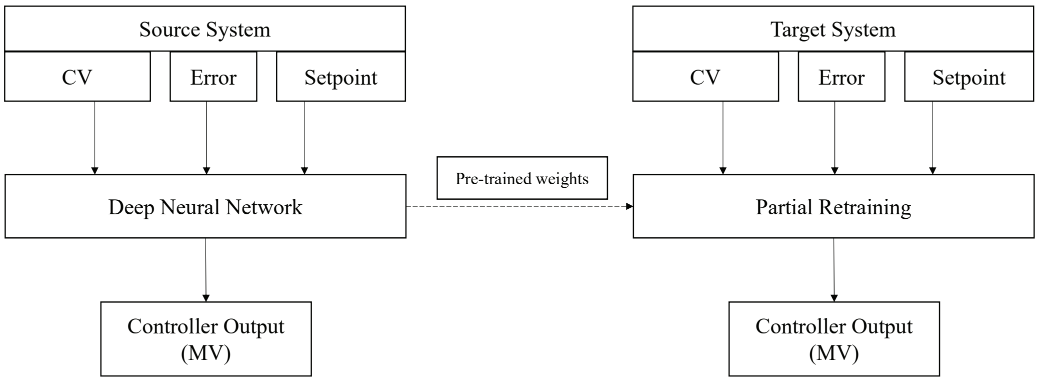

Figure 1. A network-based transfer-learning approach to develop an approximate MPC for a target system using data from a source system.

2. Materials and Methods

The source system in this study is a dual tank arrangement [54] (See Figure 2). The gravity-drained dual tank system uses pump speed and valve position to regulate the liquid level in an upper tank and a lower tank. The upper tank drains to the lower tank, and the lower tank drains to a reservoir. The control system adjusts the pump speed to return the liquid to the tanks and uses a valve to split the flow of liquid between the top and bottom tanks. The valve is adjusted by an MPC that regulates the liquid level in the tanks, ensuring that the tanks remain at a predetermined level. The liquid that flows into the upper tank cascades down to the lower tank. This creates a second-order response between the pump action and lower liquid tank level because of the dynamics of the first tank. The pump speed and valve position are adjusted to maintain the desired liquid levels in both the upper and lower tanks. This system uses the force of gravity and a combination of valve and pump to regulate the liquid level in the upper and lower tanks, ensuring that the tanks remain at their desired levels and that any excess liquid is properly drained to the reservoir. The system benefits from MPC regulation, similar to an application of model predictive control for improved water recovery and throughput stability for tailings reprocessing [55] that adjusts the inlet flow to maintain density and surge tank level.

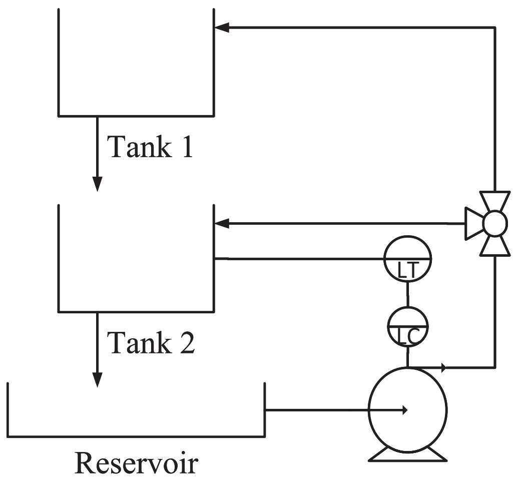

Figure 2. MIMO level control on a dual tank system. Adapted from [54].

The liquid level system is simulated using a first-principles model with Equations (1) and (2), where H_1 and H_2 are the levels (0–100%) of tank 1 and 2 respectively, c_1 and c_2 are inlet valve coefficient and tank outlet coefficient respectively, v corresponds to the valve output, and p corresponds to the pump output, both scaled from 0% to 100%. The same equations are used to generate the predictions used for MPC. Note that this MPC is intentionally formulated as an idealization. While this is not representative of an MPC in practice, it highlights the main goal of this transfer learning exercise, which is to obtain accurate control in the target system. Therefore, the design of the source system is considered part of a feature engineering exercise to gather fit-for-purpose data:

$$\frac{dH_1}{dt} = c_1(1-v)p - c_2\sqrt{H_1} \tag{1}$$

$$\frac{dH_2}{dt} = c_1 v p + c2\sqrt{H_1} - c2\sqrt{H_2} \tag{2}$$

The simulation and MPC were developed using GEKKO [56,57], a dynamic simulation and optimization suite for Python. The MPC developed used a squared error objective function and the level on both tanks are controlled by adjusting the speed of the pump (%) and the valve opening (%).

The target system in this study consists of a small scale temperature control unit [58], where the power input is regulated to heat up a small transistor to a desired temperature, as shown in Figure 3. The temperature control lab implements MPC with an Arduino, two heaters, and two temperature sensors to maintain desired temperature setpoints. The heaters transfer thermal energy to the temperature sensors through conduction and convection while also losing heat to the surroundings with radiative and convective heat transfer. This pocket-sized lab is a hardware benchmark [59] for model identification and controller development for testing new methods [60–62] and reinforcing existing methods in control theory through educational studies [58,63,64].

The dynamics of this system are described by Equations (3) and (4), where m, C_p and T are the mass, heat capacity and temperature of the transistor, U is an overall heat transfer coefficient, A is the surface area of the transistor, and T_∞ is the ambient temperature, with Q_1 being the power input to the transistor:

$$mC_p\frac{dT_1}{dt} = UA(T_\infty - T_1) + Q_1 \tag{3}$$

$$mC_p\frac{dT_2}{dt} = UA(T_\infty - T_2) + Q_2 \quad (4)$$

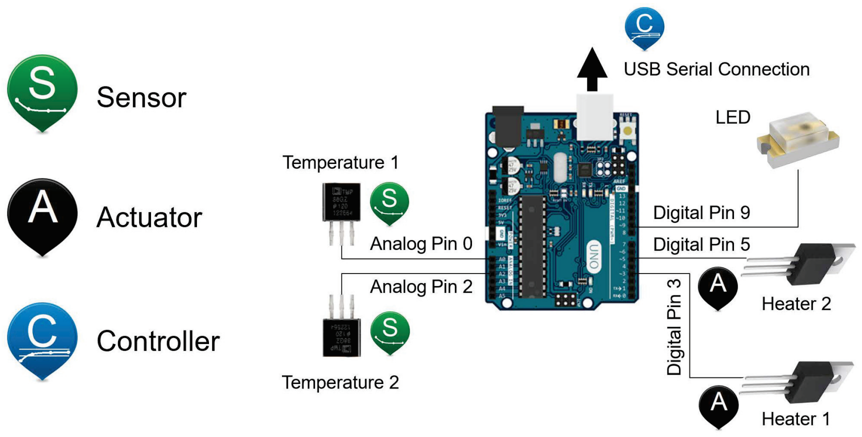

Figure 3. Small-scale temperature control unit [65].

An LSTM network developed in TensorFlow [66] is used to learn the control actions of the source and target systems by taking as inputs the measured quantities ($Y1$ and $Y2$), the setpoints ($Y1_{sp}$ and $Y2_{sp}$) and the controller error ($Y1 - Y1_{sp}$ and $Y2 - Y2_{sp}$). The outputs of the network ($\hat{U}1$ and $\hat{U}2$) are approximate control actions for the actuators. The architecture of the LSTM network is shown in Table 2.

Table 2. LSTM model summary.

Layer	Output Dimension	Number of Parameters
LSTM	$Batch \times 5 \times 100$	42,800
Droput	$Batch \times 5 \times 100$	0
LSTM	$Batch \times 5 \times 100$	80,400
Droput	$Batch \times 5 \times 100$	0
LSTM	$Batch \times 100$	80,400
Droput	$Batch \times 100$	0
Dense	$Batch \times 2$	202
Total trainable parameters		203,802

Three scenarios are analyzed in this study: In the first scenario, the LSTM network learns the control policy from a target system with a linear MPC (2500 points). This serves as a reference to evaluate the performance of an approximate MPC model derived from source system data. In this case, the inputs to the network are the measured temperature of each sensor ($T1$ and $T2$), the setpoints ($T1_{sp}$ and $T2_{sp}$) and the controller error ($T1 - T1_{sp}$ and $T2 - T2_{sp}$). The outputs of the network correspond to approximate power inputs to each transistor ($U1$ and $U2$). The linear MPC uses an autoregressive model with exogenous inputs (ARX), and both the development of the model and the solution of the optimization subroutines are handled with GEKKO [56]. The temperature setpoints (°C) and time between setpoint changes (seconds) are selected at random from $U(40, 60)$ and $U(120, 360)$, respectively.

In the second scenario, 50,000 points of data are gathered from the source system, where the setpoints (%) are selected at random from $U(20, 80)$ for each tank level, and the time between set point changes (seconds) are selected at random from $U(120, 360)$. An LSTM network is then used to learn the control policy from 50,000 points of data from the source system, taking as inputs the measured level from each tank ($H1$ and $H2$), the setpoints ($H1_{sp}$ and $H2_{sp}$) and the controller error ($H1 - H1_{sp}$ and $H2 - H2_{sp}$). The resulting model is then deployed as an approximate MPC to the target system.

In the third scenario, the same model developed in the second scenario is used as a starting point to be re-trained using 600 points of data from the target system. This is achieved by saving the weights of the model developed in the second scenario and retraining all weights for 300 epochs. The resulting model is then deployed as approximate controller to the target system in an online test.

In all scenarios, the deployed models are tested on the same set of setpoint changes. The temperature setpoints (°C) and time between setpoint changes (seconds) are initially selected at random from $U(30, 70)$ and $U(120, 360)$ respectively. The mean squared error (MSE) between the measured temperature and the temperature setpoint is used as a metric to compare the performance of each controller. An overview of the transfer learning process is shown in Figure 4.

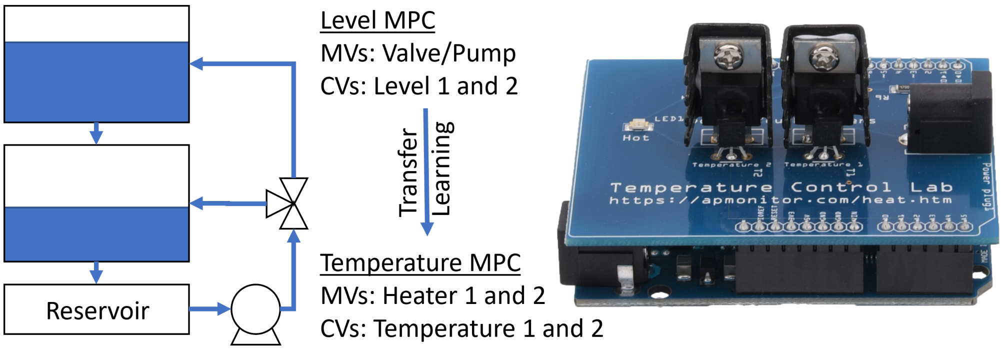

Figure 4. Overview of the transfer learning between liquid level MPC and temperature MPC.

3. Results and Discussion

Table 3 shows the MSE for the three scenarios described in Section 2. The MSE for a linear MPC is also included as reference. All training data are generated with simulated temperature response, while final tests are performed with the Arduino hardware in closed-loop MPC mode.

Table 3. Mean squared error (measured temperature vs. temperature setpoint).

Model	Linear MPC	LSTM Emulator	LSTM Transfer	LSTM Transfer Re-Train
MSE	27.75	102.52	131.82	102.43

In the first scenario, an LSTM network learns the control policy from the target system, which is controlled by a linear MPC. Figure 5 shows the fit of the model in a training and a validation set after 300 epochs. The model is then deployed as an approximate MPC to the target system in an online test. Figure 6 shows the performance of the approximate MPC, which follows the general behavior of the original linear MPC but has an MSE that is more than three times higher. This example of deep learning-based approximate MPC is the least complex of the three scenarios presented; it requires large amounts of training data, and fine tuning of the parameters of the neural network (e.g., learning rate, batch size, number of layers and nodes), but these can be adjusted by monitoring the learning metrics during training.

In the second scenario, an LSTM is trained with 50,000 points of data from the source system and deployed to the target system. Figure 7 shows the resulting performance of the approximate controller. Although the MSE is larger than that of the first scenario, the controller is able to maintain the general behavior from the MPC that was controlling the source system.

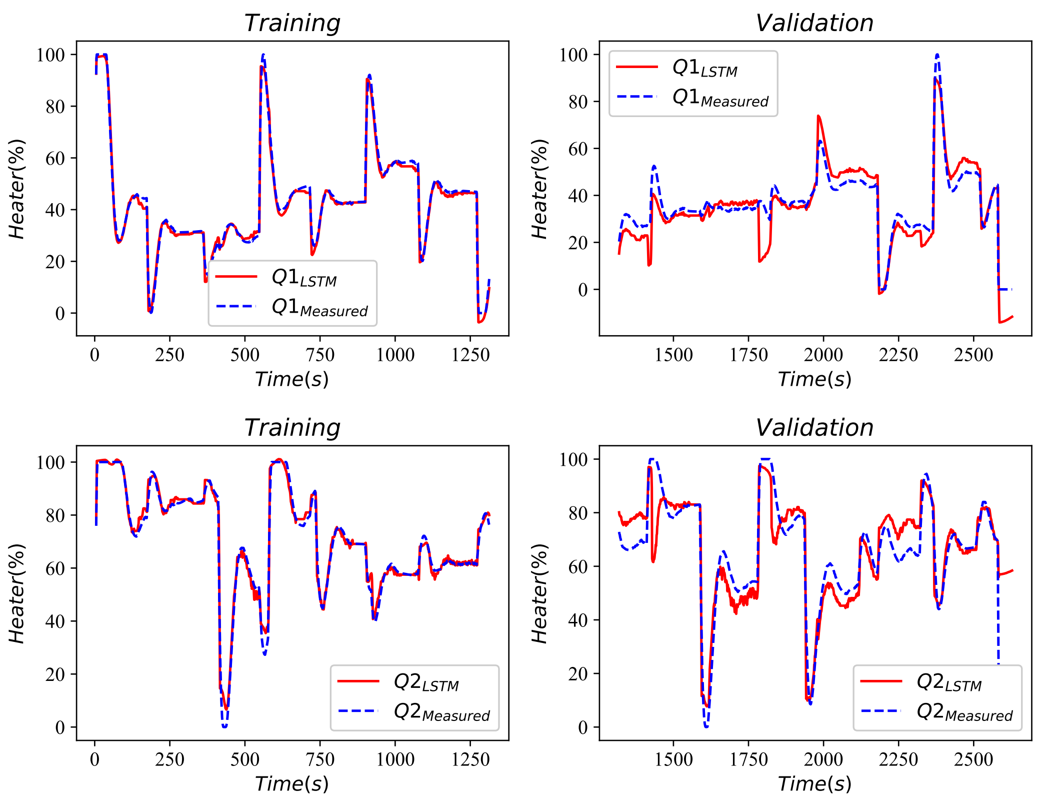

Figure 5. Training and validation fit from training an LSTM to learn the control actions of the target system. The model is trained on 2600 points of data for 300 epochs.

This is the first example of transfer learning in this study, and it shows that it is possible to develop an approximate MPC for a target system by using data from a source system, even if the dynamics and gains are not the same. In this test, the LSTM network is capable of extracting high-level features that are transferable from one system to another, but it is important to note that the selection of inputs to the approximate MPC (Y, Sp, $Error$) is also a key abstraction that helps the LSTM network generalize to other scenarios. The complexity of this technique increases due to the introduction of source data. In this case, the training metrics are affected not only by the parameters of the neural network, but also by the amount and quality of the source data.

In the third scenario, an LSTM is trained with 50,000 points of data from the source system, and partially retrained with 600 points of data from the target system under linear MPC. Figure 8 shows the model fit after training for 300 epochs. The model is then deployed to the target system in an online test as shown in Figure 9. This approximate MPC has an MSE of 99.45, which is similar to the performance in scenario 1, except this is using only a quarter of the data for training.

This is the second example of transfer learning in this study. It presents a framework for developing an approximate MPC by pre-training an LSTM network with source data and fine tuning with target data as shown in Algorithm 1.

Note that for transfer learning to offer improved control of the target system, the controller on the source system must exhibit improved performance over the initial control of the target system, as measured by the MSE. Additionally, this framework takes into account the number of points from the target data as the only tuning parameter to control the learning. In this case, 600 points was found by trial and error to be a satisfactory quantity. The effect of additional parameters such as learning rate, or batch size, along with optimization algorithms to find the optimal tuning parameters that minimize the MSE on the deployed controller should be explored in future publications.

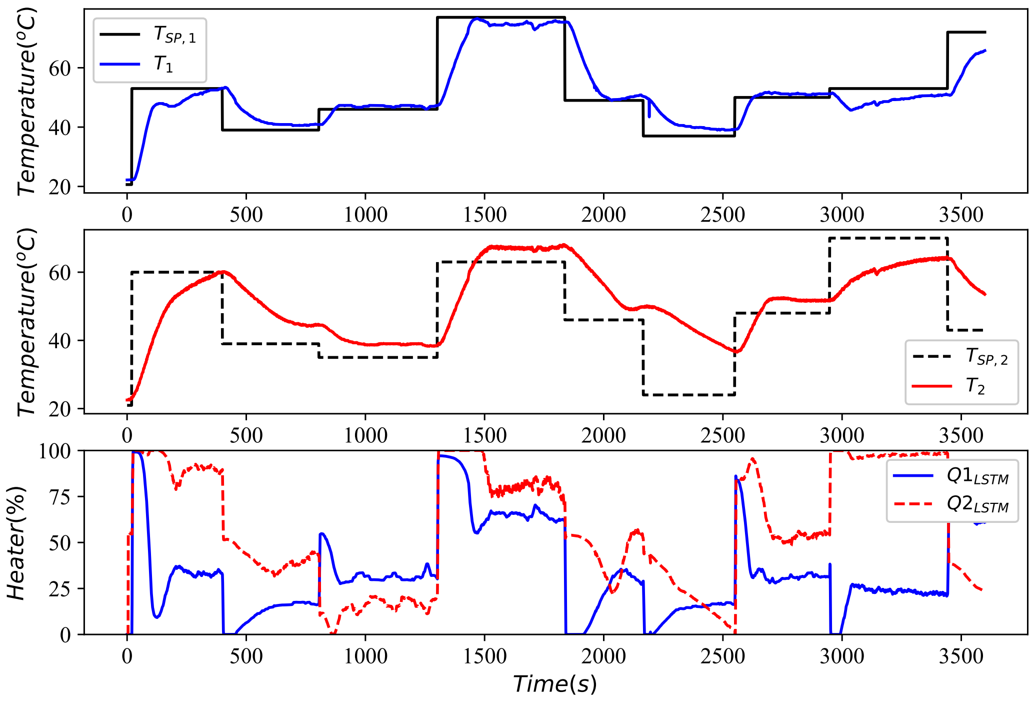

Figure 6. Scenario 1. Temperature and heater output of an approximate MPC control deployed online to a small scale temperature control (target system). The approximate MPC control is trained using only target data.

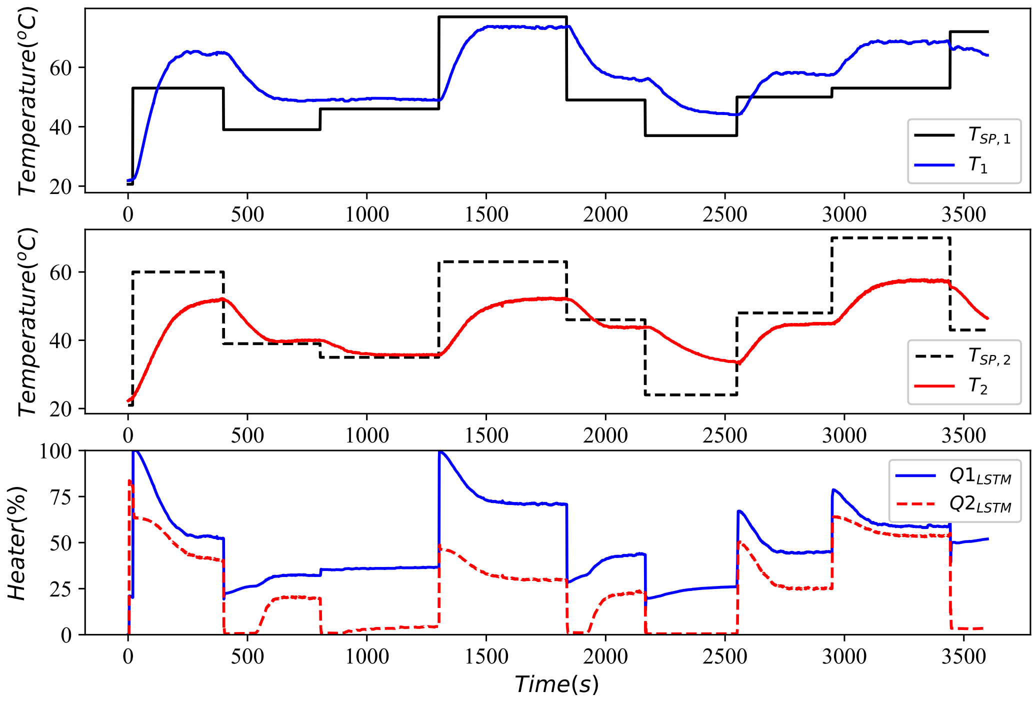

Figure 7. Scenario 2. Temperature and heater output of an approximate MPC control deployed online to a small-scale temperature control (target system). The approximate MPC control is trained using only source data.

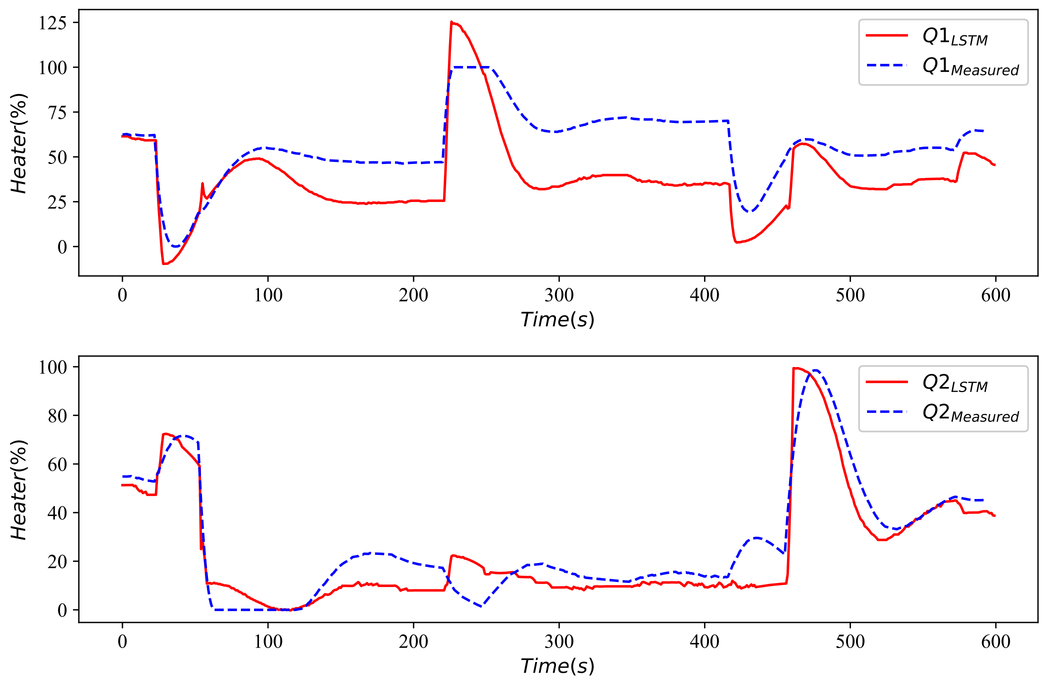

Figure 8. Validation fit after an LSTM network trained on source domain data is re-trained with 600 points of target domain data for 300 epochs.

Algorithm 1 A framework for approximate MPC using LSTM networks and transfer learning.

```
1: $CV$, $SP$ and $Error$ data are collected from a target system.
2: The control performance is measured by $MSE_{Initial}$
3: $CV$, $SP$ and $Error$ data are collected from a source system.
4: An LSTM network is pre-trained with data points from the source system.
5: The LSTM network is fine tuned with data points from the target system.
6: The performance of the fine tuned LSTM network is measured by the $MSE_{Approx.MPC}$
   after being deployed.
7: while $MSE_{Approx.MPC} > MSE_{Initial}$ do
8:     Fine tune the controller with additional points from the target system.
9: end while
```

Scenario 3 shows an improvement of 28% compared to deploying to the target system without retraining (scenario 2). It is important to note that the training data used to develop the approximate MPC from source data used random changes in the setpoint, with $U(20, 80)$, while the training data from the target system that were used to partially retrain the approximate MPC used random changes in the setpoint, with $U(40, 60)$. Figure 9 shows that at around 1500 seconds, the approximate MPC extrapolates to meet a setpoint past the range of the target domain data. This is also observed in Figure 7, except the MSE is larger in that case. The complexity of this technique increases even further, and the performance of the learning metrics is now affected by the amount of quality of both the source data and the target data.

Additional future work is needed to remove the persistent offset through the use of state estimation or control variable bias in the approximate MPC and transfer learning framework [67]. The state estimation removes the persistent offset between the setpoint and process variable and would further improve the MSE results. It is purposefully left out of this study to isolate the benefit of transfer learning of only the controller.

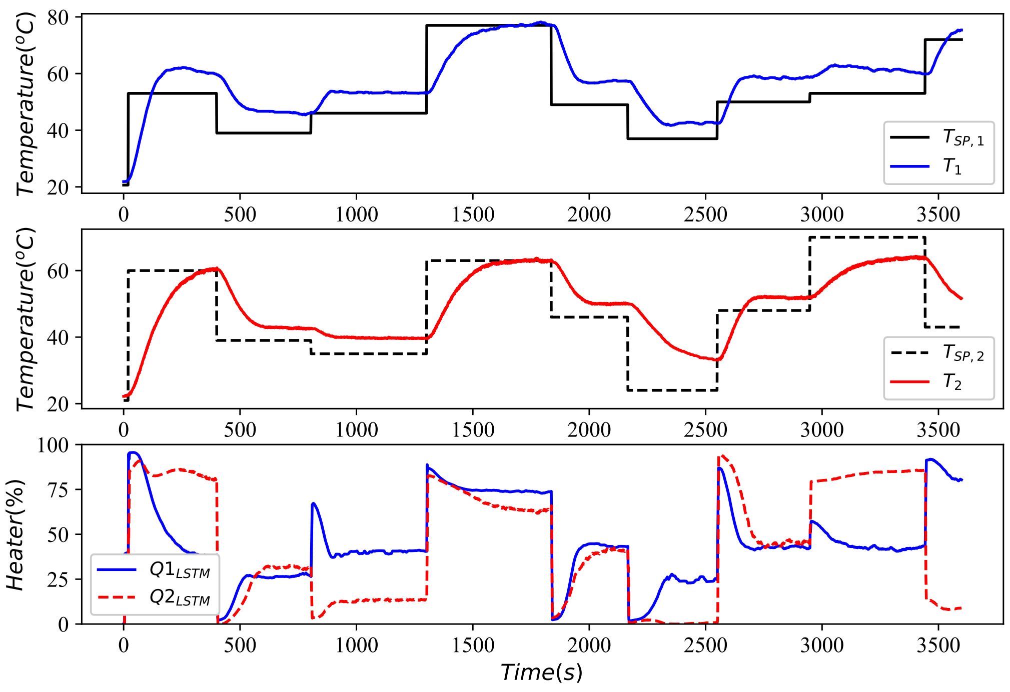

Figure 9. Scenario 3. Temperature and heater output of an approximate MPC control deployed online to a small scale temperature control (target system). The approximate MPC control is trained using 50,000 points of source data and fine tuned with 600 points of target data.

4. Conclusions

This study explored the use of LSTM networks to create an approximate MPC that can control a MIMO system. A deep learning-based control trained with data from a linear MPC maintains the behavior of the source MPC, but the MSE is 102.52 compared to 22.75 from linear MPC. Additionally, a framework for approximate MPC using LSTM networks and transfer learning was demonstrated by training an approximate MPC control on closed-loop data from a MIMO level control system and deploying to a small-scale MIMO temperature control unit. An approximate MPC trained on source domain data alone can control the temperature unit within 23% error (MSE is 131.82) compared to the approximate MPC trained on target domain data (MSE is 102.52). However, when the approximate MPC trained on source domain data is partially retrained on a small portion of the target domain data (600 points), the performance improves, and it is within 1% error (MSE is 102.43) compared to the approximate MPC trained on target domain data (MSE is 102.52). Furthermore, the transfer-learning based controller developed in this study can help extrapolate in some instances to setpoint changes outside the domain of the target data. While this study shows promising results using an LSTM network, future work should explore the use of other network architectures, such as Transformers, which have been successfully used for similar transfer learning tasks in the field of natural language processing [68] and process automation [69]. More work is needed on transfer learning for state estimation and methods to remove the potential offset from approximate MPC and other persistent errors from incomplete learning. Additionally, methods to quantify, generalize, and optimize the transfer of learning for approximate MPC formulations should be developed.

Author Contributions: Conceptualization, methodology, validation and formal analysis: S.A.M.; software, J.P. and S.A.M.; writing—review and editing, S.A.M., A.M.M., C.M.S., J.P. and J.D.H.; visualization, S.A.M. and A.M.M.; supervision, J.D.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Reliable Controls Corporation.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable

Acknowledgments: The authors acknowledge the funding and technical support provided by Reliable Controls Corporation.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ljung, L. Perspectives on system identification. *Annu. Rev. Control* **2010**, *34*, 1–12.
2. Otter, D.W.; Medina, J.R.; Kalita, J.K. A Survey of the Usages of Deep Learning for Natural Language Processing. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 604–624. [CrossRef]
3. Zhang, Z.; Geiger, J.; Pohjalainen, J.; Mousa, A.E.D.; Jin, W.; Schuller, B. Deep Learning for Environmentally Robust Speech Recognition: An Overview of Recent Developments. *ACM Trans. Intell. Syst. Technol.* **2018**, *9*, 49:1–49:28. [CrossRef]
4. Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ronneberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapenko, A.; et al. Highly accurate protein structure prediction with AlphaFold. *Nature* **2021**, *596*, 583–589. [CrossRef]
5. Al Seyab, R.; Cao, Y. Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation. *J. Process Control* **2008**, *18*, 568–581. [CrossRef]
6. Wong, W.C.; Chee, E.; Li, J.; Wang, X. Recurrent Neural Network-Based Model Predictive Control for Continuous Pharmaceutical Manufacturing. *Mathematics* **2018**, *6*, 242. [CrossRef]
7. Schwedersky, B.B.; Flesch, R.C.C.; Dangui, H.A.S. Nonlinear MIMO system identification with echo-state networks. *J. Control Autom. Electr. Syst.* **2022**, *33*, 743–754. [CrossRef]
8. Schweidtmann, A.M.; Esche, E.; Fischer, A.; Kloft, M.; Repke, J.U.; Sager, S.; Mitsos, A. Machine Learning in Chemical Engineering: A Perspective. *Chem. Ing. Tech.* **2021**, *93*, 2029–2039. [CrossRef]
9. Henao, C.A.; Maravelias, C.T. Surrogate-based superstructure optimization framework. *AICHE J.* **2011**, *57*, 1216–1232. [CrossRef]
10. Fahmi, I.; Cremaschi, S. Process synthesis of biodiesel production plant using artificial neural networks as the surrogate models. *Comput. Chem. Eng.* **2012**, *46*, 105–123. [CrossRef]
11. Boukouvala, F.; Hasan, M.M.F.; Floudas, C.A. Global optimization of general constrained grey-box models: New method and its application to constrained PDEs for pressure swing adsorption. *J. Glob. Optim.* **2017**, *67*, 3–42. [CrossRef]
12. Wilson, Z.T.; Sahinidis, N.V. The ALAMO approach to machine learning. *Comput. Chem. Eng.* **2017**, *106*, 785–795. [CrossRef]
13. Zhang, Q.; Grossmann, I.E.; Sundaramoorthy, A.; Pinto, J.M. Data-driven construction of convex region surrogate models. *Optim. Eng.* **2016**, *17*, 289–332. [CrossRef]
14. Zomorodi, H.; Landers, R.G. Extrusion based additive manufacturing using Explicit Model Predictive Control. In Proceedings of the 2016 American Control Conference (ACC), Boston, MA, USA, 6–8 July 2016; pp. 1747–1752. [CrossRef]
15. Mu, J.; Rees, D. Approximate model predictive control for gas turbine engines. In Proceedings of the 2004 American Control Conference, Boston, MA, USA, 30 June–2 July 2004; Volume 6, pp. 5704–5709. [CrossRef]
16. Yang, S.; Wan, M.P.; Chen, W.; Ng, B.F.; Dubey, S. Experiment study of machine-learning-based approximate model predictive control for energy-efficient building control. *Appl. Energy* **2021**, *288*, 116648. [CrossRef]
17. Pon Kumar, S.S.; Tulsyan, A.; Gopaluni, B.; Loewen, P. A Deep Learning Architecture for Predictive Control. *IFAC-PapersOnLine* **2018**, *51*, 512–517. [CrossRef]
18. Alessio, A.; Bemporad, A. A survey on explicit model predictive control. In *Nonlinear Model Predictive Control*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 345–369.
19. Tøndel, P.; Johansen, T.A.; Bemporad, A. An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica* **2003**, *39*, 489–497. [CrossRef]
20. Csekő, L.H.; Kvasnica, M.; Lantos, B. Explicit MPC-based RBF neural network controller design with discrete-time actual Kalman filter for semiactive suspension. *IEEE Trans. Control Syst. Technol.* **2015**, *23*, 1736–1753. [CrossRef]
21. Grosso, J.M.; Ocampo-Martínez, C.; Puig, V. Learning-based tuning of supervisory model predictive control for drinking water networks. *Eng. Appl. Artif. Intell.* **2013**, *26*, 1741–1750. [CrossRef]
22. Katz, J.; Pappas, I.; Avraamidou, S.; Pistikopoulos, E.N. Integrating deep learning models and multiparametric programming. *Comput. Chem. Eng.* **2020**, *136*, 106801. [CrossRef]
23. Chen, J.; Chen, Y.; Tong, L.; Peng, L.; Kang, Y. A backpropagation neural network-based explicit model predictive control for DC–DC converters with high switching frequency. *IEEE J. Emerg. Sel. Top. Power Electron.* **2020**, *8*, 2124–2142. [CrossRef]
24. Zheng, C.; Dragičević, T.; Blaabjerg, F. Model predictive control-based virtual inertia emulator for an islanded alternating current microgrid. *IEEE Trans. Ind. Electron.* **2020**, *68*, 7167–7177. [CrossRef]
25. Moness, M.; Moustafa, A.M. Real-time switched model predictive control for a cyber-physical wind turbine emulator. *IEEE Trans. Ind. Inform.* **2019**, *16*, 3807–3817. [CrossRef]
26. Wang, S.; Dragicevic, T.; Gontijo, G.F.; Chaudhary, S.K.; Teodorescu, R. Machine learning emulation of model predictive control for modular multilevel converters. *IEEE Trans. Ind. Electron.* **2020**, *68*, 11628–11634. [CrossRef]
27. Yan, Z.; Wang, J. Nonlinear model predictive control based on collective neurodynamic optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 840–850. [CrossRef] [PubMed]

28. Novak, M.; Dragicevic, T. Supervised imitation learning of finite-set model predictive control systems for power electronics. *IEEE Trans. Ind. Electron.* **2020**, *68*, 1717–1723. [CrossRef]
29. Hofer, M.; Muehlebach, M.; D'Andrea, R. Application of an approximate model predictive control scheme on an unmanned aerial vehicle. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 2952–2957.
30. Pin, G.; Filippo, M.; Pellegrino, F.; Fenu, G.; Parisini, T. Approximate model predictive control laws for constrained nonlinear discrete-time systems: Analysis and offline design. *Int. J. Control* **2013**, *86*, 804–820. [CrossRef]
31. Gan, M.; Hou, H.; Wu, X.; Liu, B.; Yang, Y.; Xie, C. Machine learning algorithm selection for real-time energy management of hybrid energy ship. *Energy Rep.* **2022**, *8*, 1096–1102. [CrossRef]
32. Wang, G.; Jia, Q.S.; Qiao, J.; Bi, J.; Zhou, M. Deep learning-based model predictive control for continuous stirred-tank reactor system. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 3643–3652. [CrossRef]
33. Wang, Z.; Hong, T. Reinforcement learning for building controls: The opportunities and challenges. *Appl. Energy* **2020**, *269*, 115036. [CrossRef]
34. Chen, Y.; Tong, Z.; Zheng, Y.; Samuelson, H.; Norford, L. Transfer learning with deep neural networks for model predictive control of HVAC and natural ventilation in smart buildings. *J. Clean. Prod.* **2020**, *254*, 119866. [CrossRef]
35. Han, H.; Liu, Z.; Liu, H.; Qiao, J. Knowledge-data-driven model predictive control for a class of nonlinear systems. *IEEE Trans. Syst. Man, Cybern. Syst.* **2019**, *51*, 4492–4504. [CrossRef]
36. Chen, B.; Cai, Z.; Bergés, M. Gnu-rl: A precocial reinforcement learning solution for building hvac control using a differentiable mpc policy. In Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, New York, NY, USA, 13–14 November 2019; pp. 316–325.
37. Zhu, L.; Takami, G.; Kawahara, M.; Kanokogi, H.; Matsubara, T. Alleviating parameter-tuning burden in reinforcement learning for large-scale process control. *Comput. Chem. Eng.* **2022**, *158*, 107658. [CrossRef]
38. Koycegiz, C.; Buyukyildiz, M. Calibration of SWAT and Two Data-Driven Models for a Data-Scarce Mountainous Headwater in Semi-Arid Konya Closed Basin. *Water* **2019**, *11*, 147. [CrossRef]
39. Liu, Q.; Liu, Y.; Niu, J.; Gui, D.; Hu, B.X. Prediction of the Irrigation Area Carrying Capacity in the Tarim River Basin under Climate Change. *Agriculture* **2022**, *12*, 657. [CrossRef]
40. Forghanparast, F.; Mohammadi, G. Using Deep Learning Algorithms for Intermittent Streamflow Prediction in the Headwaters of the Colorado River, Texas. *Water* **2022**, *14*, 2972. [CrossRef]
41. Privara, S.; Široky, J.; Ferkl, L.; Cigler, J. Model predictive control of a building heating system: The first experience. *Energy Build.* **2011**, *43*, 564–572. [CrossRef]
42. Yang, Q.; Zhang, Y.; Dai, W.; Pan, S.J. *Transfer Learning*; Cambridge University Press: Cambridge, UK, 2020. [CrossRef]
43. Carbonell, J.G. A computational model of analogical problem solving. In Proceedings of the IJCAI—7th International Joint Conference on Artificial Intelligence, Vancouver, BC, Canada, 24–28 August 1981; Volume 81, pp. 147–152.
44. Winston, P.H. Learning and reasoning by analogy. *Commun. ACM* **1980**, *23*, 689–703. [CrossRef]
45. Forbus, K.D.; Gentner, D.; Markman, A.B.; Ferguson, R.W. Analogy just looks like high level perception: Why a domain-general approach to analogical mapping is right. *J. Exp. Theor. Artif. Intell.* **1998**, *10*, 231–257. [CrossRef]
46. Thrun, S.; Pratt, L. Learning to learn: Introduction and overview. In *Learning to Learn*; Springer: New York, NY, USA, 1998; pp. 3–17.
47. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
48. Mohammadi, S.M.; Enshaeifar, S.; Hilton, A.; Dijk, D.J.; Wells, K. Transfer Learning for Clinical Sleep Pose Detection Using a Single 2D IR Camera. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2021**, *29*, 290–299. [CrossRef]
49. Alotaibi, A. Automated and Intelligent System for Monitoring Swimming Pool Safety Based on the IoT and Transfer Learning. *Electronics* **2020**, *9*, 2082. [CrossRef]
50. Tan, C.; Sun, F.; Kong, T.; Zhang, W.; Yang, C.; Liu, C. A Survey on Deep Transfer Learning. In *Artificial Neural Networks and Machine Learning—ICANN 2018*; Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2018; pp. 270–279. [CrossRef]
51. Wan, C.; Pan, R.; Li, J. Bi-weighting domain adaptation for cross-language text classification. In Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence Bi-Weighting Domain Adaptation for Cross-Language Text Classification, Barcelona, Spain, 16–22 July 2011.
52. Matasci, G.; Volpi, M.; Kanevski, M.; Bruzzone, L.; Tuia, D. Semisupervised Transfer Component Analysis for Domain Adaptation in Remote Sensing Image Classification. *IEEE Trans. Geosci. Remote Sens.* **2015**, *53*, 3550–3564. [CrossRef]
53. Li, W.; Gu, S.; Zhang, X.; Chen, T. Transfer learning for process fault diagnosis: Knowledge transfer from simulation to physical processes. *Comput. Chem. Eng.* **2020**, *139*, 106904. [CrossRef]
54. Hedengren, J.D. Level Regulation with MPC. 2018. Available online: https://apmonitor.com/do/index.php/Main/LevelControl (accessed on 15 November 2021).
55. Burchell, J.; le Roux, J.; Craig, I. Nonlinear model predictive control for improved water recovery and throughput stability for tailings reprocessing. *Control Eng. Pract.* **2023**, *131*, 105385. [CrossRef]
56. Beal, L.; Hill, D.; Martin, R.; Hedengren, J. GEKKO Optimization Suite. *Processes* **2018**, *6*, 106. [CrossRef]

57. Hedengren, J.D.; Shishavan, R.A.; Powell, K.M.; Edgar, T.F. Nonlinear modeling, estimation and predictive control in APMonitor. *Comput. Chem. Eng.* **2014**, *70*, 133–148. [CrossRef]
58. Oliveira, P.M.; Hedengren, J.D. An APMonitor Temperature Lab PID Control Experiment for Undergraduate Students. In Proceedings of the 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 10–13 September 2019. [CrossRef]
59. Park, J.; Martin, R.A.; Kelly, J.D.; Hedengren, J.D. Benchmark temperature microcontroller for process dynamics and control. *Comput. Chem. Eng.* **2020**, *135*, 106736. [CrossRef]
60. Mejía, C.; Salazar, E.; Camacho, O. A comparative experimental evaluation of various Smith predictor approaches for a thermal process with large dead time. *Alex. Eng. J.* **2022**, *61*, 9377–9394. [CrossRef]
61. Yerolla, R.; Besta, C.S. Development of tuning free SISO PID controllers for First Order Plus Time Delay (FOPTD) and First Order Lag Plus Integral Plus Time Delay model (FOLIPD) systems based on partial model matching and experimental verification. *Results Control Optim.* **2021**, *5*, 100070. [CrossRef]
62. Sharma, S.; Padhy, P.K. An indirect approach for online identification of continuous time-delay systems. *Int. J. Numer. Model. Electron. Netw. Devices Fields* **2022**, *35*, e2947. [CrossRef]
63. de Moura Oliveira, P.B.; Hedengren, J.D.; Solteiro Pires, E.J. Swarm-Based Design of Proportional Integral and Derivative Controllers Using a Compromise Cost Function: An Arduino Temperature Laboratory Case Study. *Algorithms* **2020**, *13*, 315. [CrossRef]
64. de Moura Oliveira, P.; Hedengren, J.D.; Rossiter, J. Introducing Digital Controllers to Undergraduate Students using the TCLab Arduino Kit. *IFAC-PapersOnLine* **2020**, *53*, 17524–17529. [CrossRef]
65. Hedengren, J.D. Temperature Control Lab. 2021. Available online: http://apmonitor.com/pdc/index.php/Main/ArduinoTemperatureControl (accessed on 15 November 2021).
66. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available online: https://www.tensorflow.org/ (accessed on 13 November 2022).
67. Hedengren, J.D.; Eaton, A.N. Overview of estimation methods for industrial dynamic systems. *Optim. Eng.* **2017**, *18*, 155–178. [CrossRef]
68. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.
69. Park, J. Hybrid Machine Learning and Physics-Based Modeling Approaches for Process Control and Optimization. Ph.D. Thesis, Brigham Young University, Provo, UT, USA, 2022.