Feasibility Study of Transfer Learning on LSTM Recurrent Neural Networks for Fiber Manufacturing Commercialization

by

Nilay Sawant

B.S., University of Illinois Urbana-Champaign (2020)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Engineering in Advanced Manufacturing and Design

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2022

Author .....
Department of Mechanical Engineering
August 5, 2022

Certified by .....
Brian W. Anthony
Principal Research Scientist
Thesis Supervisor

Accepted by .....
Nicolas Hadjiconstantinou
Chairman, Department Committee on Graduate Theses

Feasibility Study of Transfer Learning on LSTM Recurrent Neural Networks for Fiber Manufacturing Commercialization

by

Nilay Sawant

Submitted to the Department of Mechanical Engineering
on August 5, 2022, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Advanced Manufacturing and Design

Abstract

This thesis explores business pathways to commercialize Device Realization Lab's technology that uses deep reinforcement learning for optical fiber manufacturing control systems. A viable business solution is proposed based on feedback from venture capital investors. The solution comprises developing cloud-based software that can generate digital twins for fiber manufacturing companies. These digital twins can serve as anomaly detectors and suggest optimal input parameters that reduce production variation and tolerance, improving quality and decreasing scrap rate. Efforts to define a minimum viable product (MVP) for this business solution began with the creation of a long short-term memory recurrent neural network (LSTM RNN) model for a desktop fiber extrusion system that mimics the fiber extrusion process on the manufacturing floor. Transfer learning on the LSTM RNN was then implemented to explore the feasibility of reusing a well-developed machine learning (ML) model for a fiber material (e.g. glass fiber) to construct an ML model for a separate fiber material (e.g. nylon fiber) for which a relatively low amount of data is available. The study found that applying transfer learning reduced the mean squared error of the new fiber material model by over 40% compared to developing the model without transfer learning. This thesis strives to reveal the innovative applications of the technology that can benefit the fiber manufacturing field and defines an MVP that can be shared with venture capital investors as a first step toward commercializing this technology.

Thesis Supervisor: Brian W. Anthony
Title: Principal Research Scientist

Acknowledgments

I would like to thank my thesis advisor, Brian Anthony, who guided my thesis journey. I admire Brian's passion for exploring, and the opportunities I have gained from the Device Realization Lab have made me a better engineer, researcher, and entrepreneur. In addition, I would like to thank my fellow lab members for creating a lively environment in the lab.

I want to also thank Jose Pacheco and Professor David Hardt, who gave me the opportunity to be a part of the Master of Engineering in Advanced Manufacturing and Design program. I would also like to acknowledge all my classmates from MIT and UIUC who not only challenged me but also supported me in my young adult years. The entire MEng Cohort has made the past year a very memorable one. In addition, I thank my best friend, Anirudh, for his sincerely dependable support.

I would like to thank my loving family. My older sister, Manali, has always been a great role model for me growing up. My dad, Sanjay, taught me the importance of hard work, and I will never forget his quote, "there is no substitute for hard work." Finally, I would like to thank my mom, Sandhya, for her endless love that has made me the man I am today.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Device Realization Lab (DRL) at MIT develops novel control systems to improve the optical fiber manufacturing process. These improved control systems implement machine learning to reduce the tolerance of optical fibers to 125 microns ± 1 micron, allowing for simple assembly of the fiber and connector and eliminating several preparation steps during fiber and connector assembly.

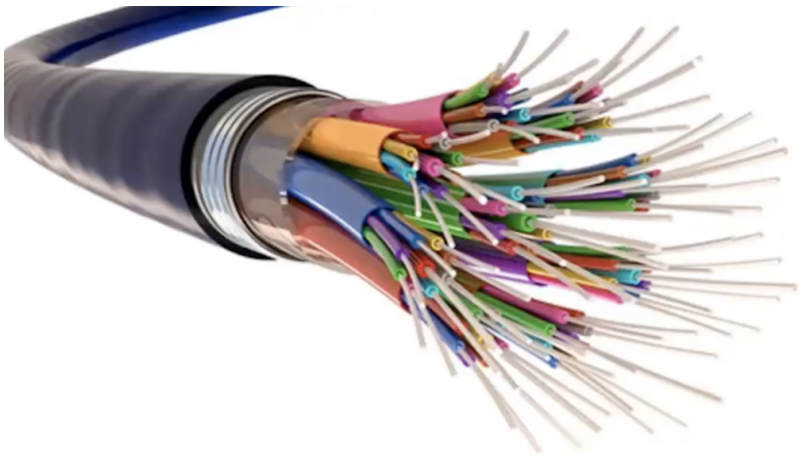


Figure 1-1: Image of optical fiber within cable [5]

The current manufacturing process has a tolerance greater ± 1 micron and thus requires post-processing steps such as sanding to unite the fiber to the connector. By removing this step, the manufacturer can save tens of millions of dollars on the cable assembly.

Figure 1-2: Image of cable with connectors [22]

Apart from the production of optical fibers, programmable logic controllers (PLCs) are also critical in the manufacturing of other products including semiconductors, automobiles, glass, paper, textile, cement, food and beverages, and pharmaceuticals.

This thesis will focus on exploring the use of improved machine learning based (ML-based) control systems for the commercialization of synthetic fiber manufacturing. Synthetic fibers are man-made (commonly extruded) fibers manufactured through chemical synthesis and include glass, polyester, carbon, and aramid fibers (e.g. Kevlar, Vectran, and Nylon).

1.1 Motivation

Although the manufacturing industry is currently accustomed to PLCs, they are time-consuming and labor-intensive. Results from isolated trial and error experiments are used to generate simple, intuitive models that inform the selection of the next tuning parameter. Because this process is repeated until the desired performance is achieved, it can result in production downtime caused by multiple design and debugging iterations. Better controllers are needed to change this status quo of manufacturing processes. Firstly, next-generation controllers can reduce production variation and tolerance, improving quality and decreasing scrap rate. Secondly, they can also control for slight variations in material properties between raw material batches to produce consistent quality products. Thirdly, next-generation controllers are less affected by breaks in the production process (i.e. machine is shut down) and

are capable of quick recovery to optimal quality.

1.1.1 Past Work

A Data-Driven Approach to System Dynamics Modeling and Control Design by George C. Chen explains the technology DRL has built to improve feedback control [2]. DRL has:

- Architected, implemented, and trained long short-term memory (LSTM) neural networks to model the process and obtain the correlation between inputs and outputs for the optical fiber extrusion manufacturing process

- Employed a system identification process using statistical analysis models

- Built a closed-loop simulation of the fiber extrusion system

Kim et al. showed how deep reinforcement learning algorithms can learn and control systems by applying the deep reinforcement learning framework on a compact fiber drawing system as an example [14]. Kim's ML-based control system "is trained and tested on a real physical fiber drawing process with stochastic behavior and nonlinear delayed dynamics,... predictively regulates the diameter to track dynamically varying reference trajectories,... [and] does not require prior analytical or numerical models of the system" [14]. Model-free Tracking Control of an Optical Fiber Drawing Process using Deep Reinforcement Learning by Sangwoon Kim further explains the actor-critic approach that was developed for the ML-based control system [13].

The technology and approach can accelerate the labor-intensive and repetitive tuning process needed to optimize controllers. Although some manufacturing companies now collect large quantities of controller and sensor data, this data is generally not used to build improved control systems. Companies continue to rely on traditional control strategies largely around PID controllers.

1.1.2 Objective

This thesis explores three business pathways to commercialize the current technology built by DRL and proposes a viable business solution based on feedback from venture capital investors. An LSTM recurrent neural network (RNN) model was developed for a desktop fiber extrusion system that mimics the fiber extrusion process on the manufacturing floor. Transfer learning on the LSTM RNN was then implemented to explore the feasibility of the proposed business solution and understand the limits of transfer learning for this application. This thesis strives to define a lean minimum viable product (MVP) and develop a prototype for this business solution that can be shared with venture capital investors and be presented to prospective clients for feedback.

1.2 Optical Fiber Manufacturing Process

Optical fiber cables provide long-distance telecommunications and transfer data at high speeds using thin flexible glass fibers that carry light.

Optical fiber manufacturing consists of two main processes: the manufacturing of the glass preform and the extrusion of the preform into a fiber. The cylindrical glass preform is manufactured by using chemical vapor deposition and sintering of silicon. The preform is then extruded into fiber and spooled in a large fiber drawing tower.

When the fiber is extruded, the preform goes through a feed mechanism. The furnace heats the preform, which allows via gravity to form into a thin fiber. Using helium gas, the fiber is cooled. The fiber is pulled and spooled through the drawing pulley and winding drum. 1-3 shows the components of the optical fiber extrusion process.

Sensors monitor parameters including furnace temperature and velocity and tension of the drawing pulley. The outputs such as the fiber diameter are also measured.

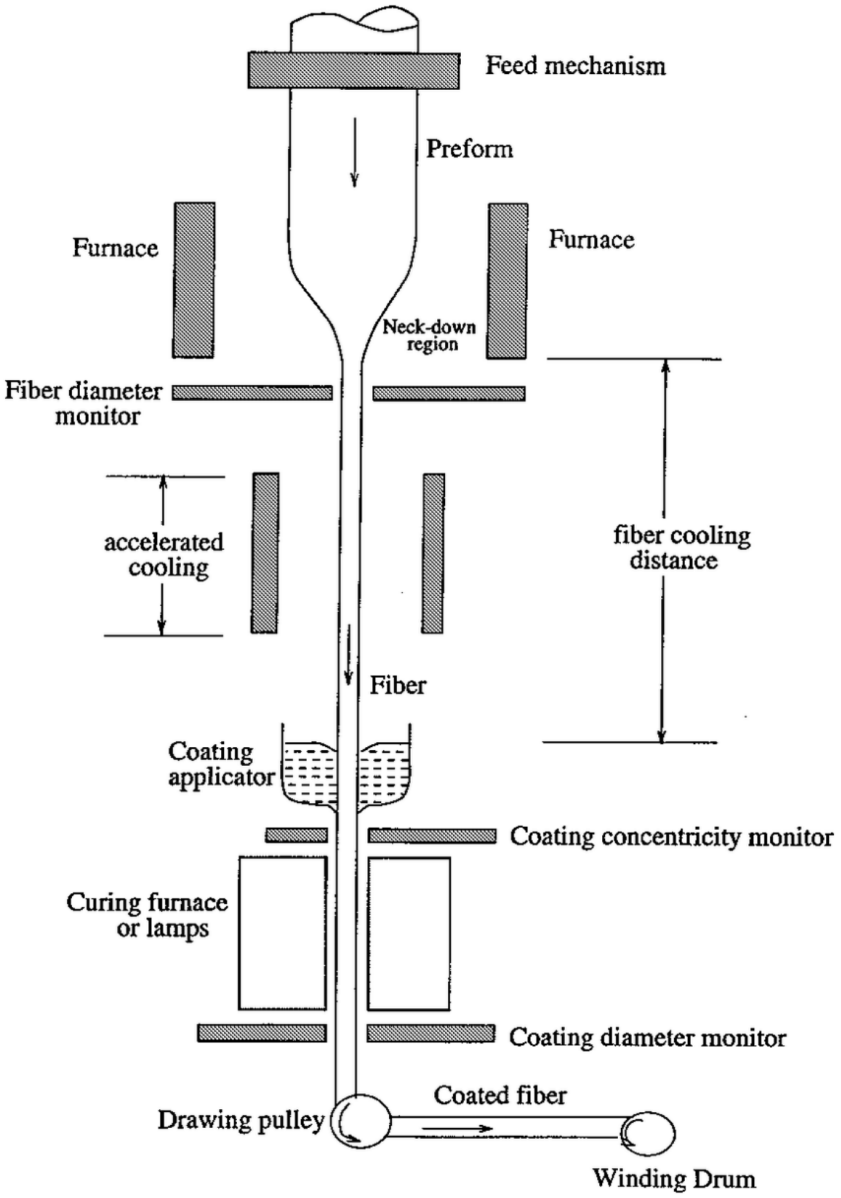


Figure 1-3: Manufacturing process of optical fiber extrusion [3]

1.3 Programmable Logic Controllers

1.3.1 Background

Programmable Logic Controllers are industrial computers that are designed to receive inputs (sensor data) and control outputs (equipment such as motors). PLCs come in a wide range of prices and are used in many systems such as robotics, manufacturing machines, and assembly lines.

Figure 1-4: Allen-Bradley CompactLogix 5480 PLC in the DRL

Engineers program PLCs using control systems that often include feedback loops in which the system's output is used as an input for the next operation.

1.3.2 Control Systems

A control system is defined as an assembly of subsystems and processes that produce a desired output and performance for a given input [18]. As an example, control systems can allow for large equipment such as an elevator to be moved with precision. 1-5 shows the input command and the measures of performance (steady state response and steady-state error). The transient response, in this case, would affect the passengers' comfort and time when using the elevator.

Nise states that the four main reasons control systems are built are power amplification, remote control, convenience of input form, and compensation for disturbances. Proportional integral derivative (PID) controllers are often used to improve control systems. Nicholas Minorsky developed the theory behind PID controllers for the automatic steering of ships [18].

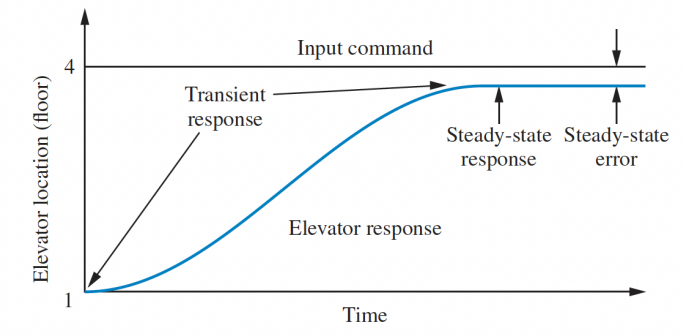


Figure 1-5: Graph of elevator location vs. time with an input command of 4th floor [18]

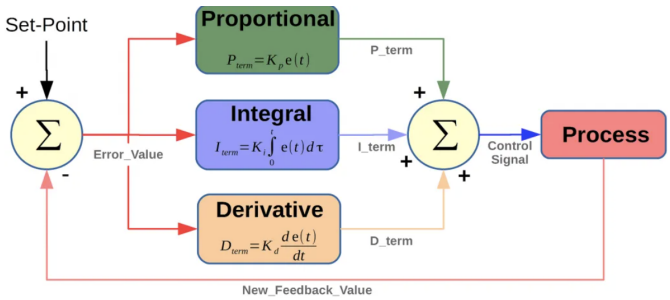


Figure 1-6: Breakdown of a PID control system [15]

PID controllers use error signals (via a feedback loop) and sum three mathematical operations to produce a control signal that advances the system to the desired output [15]. The proportional term multiplies the error by a constant or gain, Kp. The integral term consists of a constant and the integral of the error with respect to time. The derivative term consists of a constant and the derivative of the error with respect to time.

$$u = \underbrace{K_p e}_{\text{Proportional Term}} + \underbrace{K_i \int_0^t e\,dt}_{\text{Integral Term}} + \underbrace{K_d \frac{d}{dt} e}_{\text{Differential Term}}$$

Figure 1-7: Mathematical equation in continuous time used to represent a PID Controller [15]

The proportional term, integral term, and derivative term are often used to reduce

the rise time, steady-state error, and overshooting/fluctuation, respectively [15].

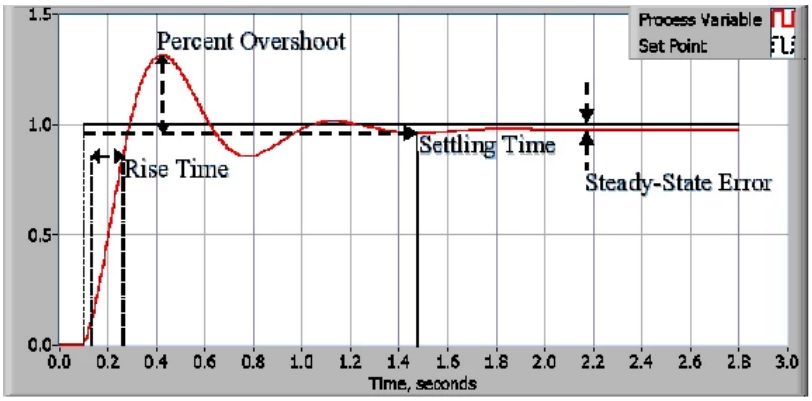


Figure 1-8: Graph labeling rise time, steady-state error and percent overshoot [9]

As an example, 1-9 shows a simplified block diagram of how a PLC and PID control system would be used in optical fiber manufacturing. The process variable (fiber diameter) can be graphed over time to get a graph similar to 1-8 and the proportional, integral, and derivative terms can be tuned to minimize the rise time, steady-state error, and overshooting.

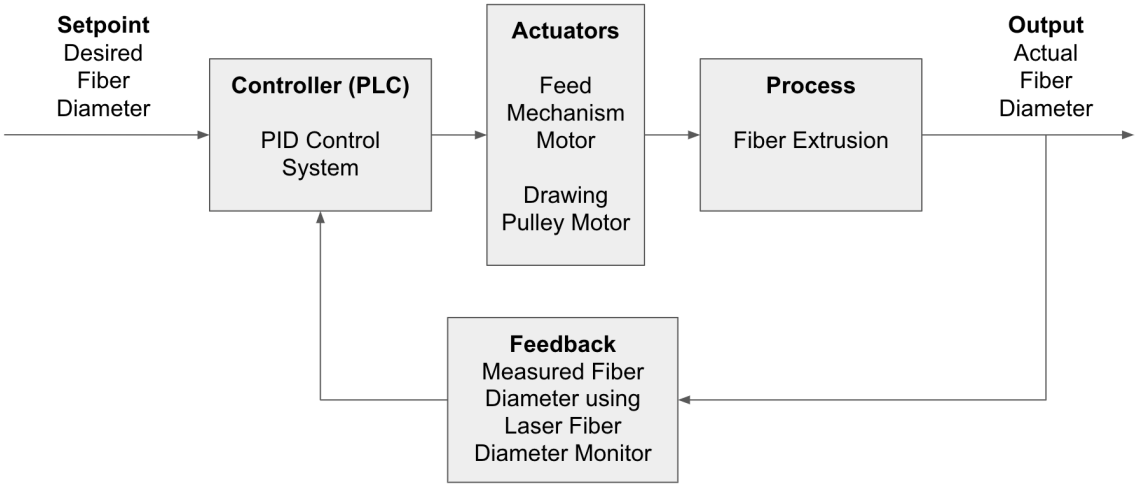


Figure 1-9: Example Control System for Fiber Extrusion

Chapter 2

Neural Networks and Transfer Learning

This chapter outlines the relevant theory behind neural networks and transfer learning. However, the chapter is not intended to be a comprehensive explanation. For a comprehensive explanation of machine learning, deep learning and long short-term memory networks refer to Deep learning for anomaly detection in multivariate time series data by Jan Paul Assendorp [1].

Artificial Neural Networks (ANNs) were first introduced in 1944 by Warren McCullough and Walter Pits as an analogy or an approximation to the operation of neural networks in the human brain [7]. Inspired by the human brain, which comprises billions of neurons, ANNs similarly consist of a network of interconnected neurons that transmit information upon stimulation from adjacent neurons [4]. Originally, ANNs have been Feed-Forward Neural Networks, in which the information is flowing in one direction and the output of a given cycle or network depends mainly on the input of the current cycle rather than previous inputs (no concept of input or output history).

2.1 LSTM Recurrent Neural Networks

Recurrent Neural Networks (RNNs) improve upon Feed-Forward Neural Networks by incorporating internal memory and therefore a concept of history or state. RNNs

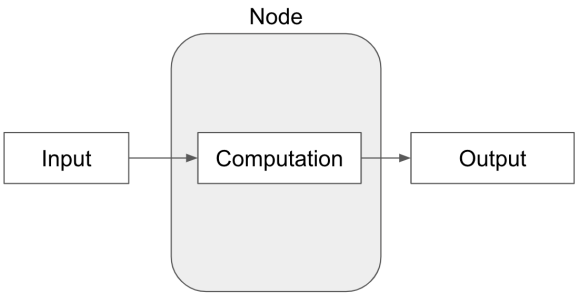


Figure 2-1: Block diagram of a node in an Artificial Neural Network

utilize both the current input and the output from the previous inputs to make a decision. After the output is generated, it is copied and then stored in the recurrent network for use in future computations [17]. RNNs are suited for time series data in which the sequence of data matters and the output of the current timestep depends on the output of previous timesteps.

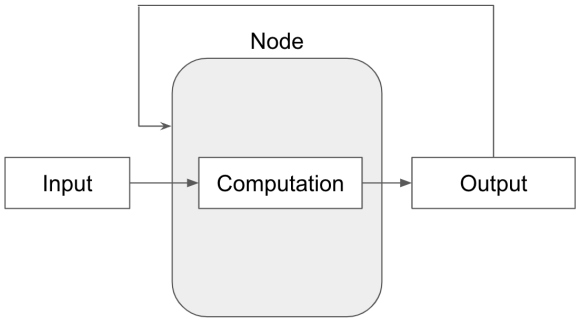


Figure 2-2: Block diagram of a node in a Recurrent Neural Network

However when the output depends on inputs at timesteps far before the current timestep with irrelevant data in between, RNNs have trouble connecting the data and detecting sequences. For these reasons, RNNs suffer from a phenomenon known as the vanishing gradient problem and the long-term dependency problem.

As a result, LSTM RNNs were developed, where a state cell with a set of gates is used to determine when the information stored in the memory gets to be used [16].

The forget gate is used to identify state information that can be forgotten as it is deemed irrelevant. The input gate is used to identify new and relevant information to be updated in the state cell. The output gate is used to determine the information that should be outputed out of all the information that is currently stored in the

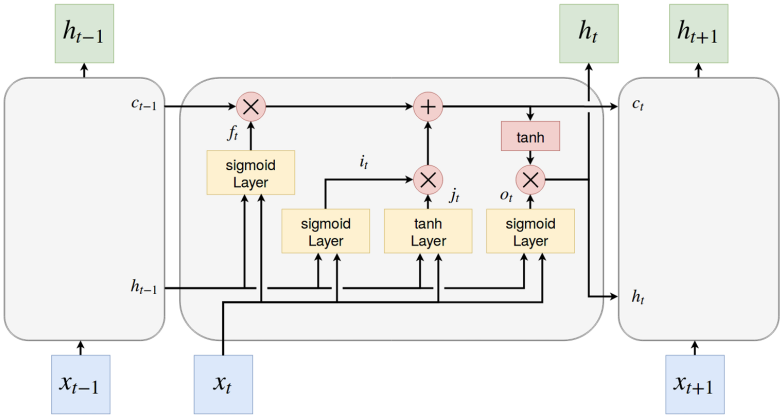


Figure 2-3: Node in an LSTM Recurrent Neural Network at timestep t [1]

state cell. The forget, input, and output gates all range from 0 to 1. A value of 0 for the output gate signifies that no information is passed through while a value of 1 indicates that all the information is passed through. In 2-3, f, i, and o signify the forget, input and output gate while c signifies the memory state of the cell. The input vector of the LSTM node is x and the output vector of the LSTM node is h.

Almost all recent applications and successes attributed to RNNs are based on LSTMs. They have been extensively used in different tasks, including speech recognition, acoustic modeling, trajectory prediction, and correlation analysis [24].

2.2 Transfer Learning

Transfer Learning is a machine learning method that utilizes the knowledge gained from solving one task (i.e. source task) to solve an unrelated, yet similar task (i.e. target task). Transfer Learning can be used when both the source task and target task have the same input and a large amount of data is available to build the source task model.

To accomplish transfer learning, the first n layers and the weights of the source model are transferred to the target model and the target data is used to train and develop the newly replaced layers. This method prevents the need to recreate all the layers in the artificial neural network for the target model.

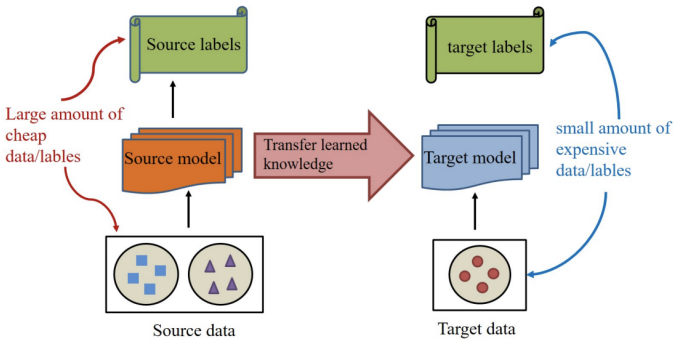


Figure 2-4: A source model with large amounts of generic data can be used to create a target model [21]

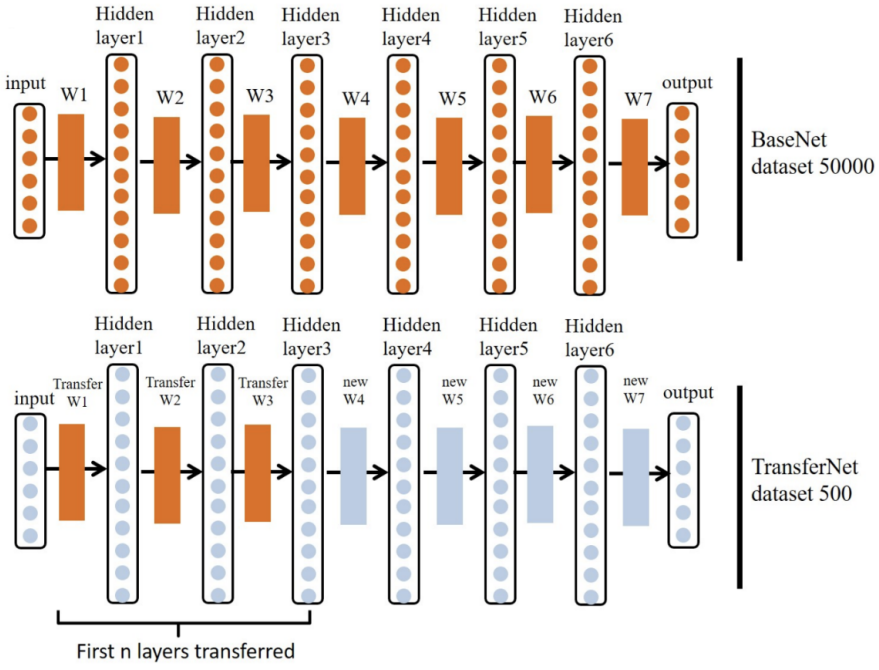


Figure 2-5: Example of the first n = 3 layers being transferred from source (BaseNet) model with dataset size of 50,000 to target (TransferNet) model with dataset size of 500 [21]

Chapter 3

Business Environment and Ecosystem

DRL members met with Duncan C. McCallum, an advisor and ex-venture capitalist, to explore the commercialization of ML-based controllers. McCallum holds Bachelor's and Master's degrees in Engineering from the Massachusetts Institute of Technology, and a Masters in Business Administration degree from Harvard Business School. He co-founded and served as the CEO of three companies and previously worked at Bessemer Venture Partners and Flagship Ventures. He is also an experienced board member, having served on 20 boards.

McCallum recommended creating sales decks and conducting customer interviews to understand the market need. One main concern is whether large PLC manufacturers, such as Seimens and Rockwell, will cooperate with this change in status quo. Another consideration is whether ML-based control systems can be easily implemented within the existing PLCs on manufacturing floors. This is particularly a concern because PLCs were not designed with ML-based control systems in mind. A thorough assessment of the resources required to carry out this integration needs to be conducted. McCallum also advised implementing a familiar revenue model, such as fixed subscriptions. He stated that while it may be beneficial to form commercial partnerships with large companies such as Siemens, it is important to remember to not relinquish ownership prematurely.

Taking account of McCallum's feedback, DRL determined three feasible business pathways for ML-based controllers: commercializing a generic ML-based PLC in

various industries, running a consultancy that develops ML-based control systems, and manufacturing a specific product that uses proprietary ML-based PLCs as the core technology.

These three business pathways were pitched to Manny Stockman, a venture capitalist at Osage University Partners (OUP). Stockman holds a Bachelor's degree in Mechanical Engineering and Materials Science Engineering from Duke University and a Doctor of Philosophy in Mechanical and Aerospace Engineering from Princeton University. He previously worked as a Staff Engineer at Lockheed Martin and is now focused on deep technology and software spinouts at OUP. OUP usually signs one to two deals per year with academic teams in the early stages of product development. While their decision is based on three vectors– product, team, and market– Stockman acknowledged that the process is holistic and that not all three vectors need to be fully developed at the early stage.

3.1 Business Pathways

The following subsections detail three possible business pathways that can be implemented to commercialize ML-based control systems.

3.1.1 Commercializing a Generic ML-based PLC

Under the generic ML-based PLC business pathway, the main goal is to commercialize a generic ML-based PLC that replaces existing PID PLCs on manufacturing floors in multiple industries. As shown by George C. Chen, considerable software engineering is required to develop an ML model for a manufacturing system [2]. Due to the uniqueness of each manufacturing process, software engineers will need to redevelop ML models for each new manufacturing system. Unfortunately, the existing technology does not allow for the straightforward replacement of PID PLCs with ML-based PLCs. A first step towards understanding the feasibility of this business pathway is to assess the resources required to accomplish this substitution.

3.1.2 Running a Consultancy

Under the second business pathway, a consultancy sells engineering services that build turnkey digital twins and ML-based control systems for PLCs allowing companies to outsource the ML-based PLC integration efforts. A digital twin is a computer-generated representation of a material object that is constructed using large amounts of data (i.e. that gathered from various sensors) and can simulate various processes to study outcomes based on differences in input parameters [8]. The services would be sold for an hourly fee or a fixed fee per project. DRL collaborates with various companies to improve their manufacturing processes. For example, through a collaboration with a coffee roasting company, DRL is formulating an ML-based control system that controls inputs such as temperature and airflow to improve the aroma of roasted coffee. DRL is also collaborating with a food packaging company's package forming process to produce aesthetically pleasing containers.

3.1.3 Manufacturing a Specific Product that uses Proprietary ML-based PLCs as the Core Technology

The third business pathway entails mass-production of a certain good that utilizes proprietary ML-based controllers. The importance of strategically choosing the product that is continuously manufactured for this purpose cannot be overstated. If replacing the PID PLC with an ML-based PLC remarkably improves product quality and reduces production costs, the product can be deemed appropriate for mass production. The exit strategy would be to sell the business to a larger company with a greater market share in the field.

3.2 Feedback from Venture Capital

For the generic ML-based PLC business pathway, Stockman cautioned against complicated integrations and excessive combinations of possible solutions as a result of adapting the generic ML model to vastly different manufacturing processes. Stock-

man recommended that the first approach for this business pathway could be to create software that builds digital twins of manufacturing systems for real-time manufacturing monitoring. The software would create an ML model using an initial dataset provided by the client company. As the company generates additional data to feed into the ML model, the software can serve as an anomaly detector and suggest outliers. As the ML model gains access to more data over time, it becomes empowered to suggest input parameters to improve the manufacturing process.

For the second business pathway, Stockman anticipated that consulting may function as a small business. However, he cautioned that the consultant's salaries will expend a substantial portion of the revenue. Stockman and McCallum warned that consulting businesses are difficult to scale and venture capital investors generally neglect to fund consultancies.

Out of the three business pathways, Stockman alluded that although the third business pathway which entails manufacturing a specific product will be the most arduous, it has the potential to make the most revenue. He recommended choosing a high-value product whose value is derived from the manufacturing process. The optimal product is one that is impossible to manufacture without the use of ML-based PLCs.

3.3 Proposed Business Solution

The second business pathway takes the least amount of time and capital to launch but has the lowest revenue potential. The third business pathway has the highest revenue potential but also takes the most amount of time and capital to launch. The pathway of developing software that auto-generates digital twins strikes a balance between the time and capital necessary to launch while having adequate revenue potential.

Stockman's proposition to develop software that auto-generates digital twins is an arduous endeavor due to the considerable number of unrelated manufacturing processes. One solution that alleviates this concern proposes to solely focus on building digital twins for synthetic fiber manufacturing companies. Large amounts of fiber

draw data, collected using the MIT fiber draw tower, would be used to create a digital twin using LSTM RNNs and physics-based models. This digital twin would serve as a generic source model that understands low-level features. Fiber manufacturing companies could then supplement the model with their own data. Transfer learning would be utilized to auto-generate and create a specific digital twin for the fiber manufacturing company. The software will programmatically train and test data to find optimal hyperparameters that minimizes error of the auto-generated ML model. The model accuracy would improve over time with the addition of data. The digital twin can be used for anomaly detection and suggesting optimal input parameters. The proposed business solution would be cloud-based and sold as a monthly subscription. Subscription cloud-based solutions are familiar to companies and venture capital investors are enticed by recurring revenue models.

Chapter 4

Experimental Setup

The potential to realize the proposed business solution using transfer learning is explored through a series of experiments. Data is collected on Fiber Extrusion Device (FrED), a desktop fiber manufacturing system that simulates the fiber extrusion process at a small scale. A large amount of data is collected with Material 1 to develop a generic LSTM RNN source model. A smaller amount of data is collected with Material 2 to develop a new LSTM RNN model. Then using fine-tuning and transfer learning, two additional models for Material 2 (a third and fourth model) are developed using previously developed layers from the Material 1 model. The mean squared error (MSE) is calculated between the predicted data and the test data for all models and compared.

4.1 FrED - Fiber Extrusion Device

David Donghyun Kim has previously detailed how FrED was built [11]. The four main systems of FrED are the extruder, laser micrometer, cooling system, and spool system.

The device uses glue sticks as preform. The stepper gear feeds the preform into the heating element which uses two 40W cartridge heaters. The preform becomes malleable and forms into a thin fiber.

The fiber goes through a laser micrometer which measures the fiber diameter and

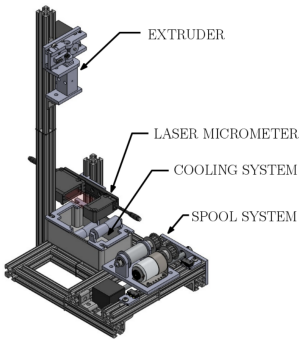


Figure 4-1: Complete assembly of FrED [11]

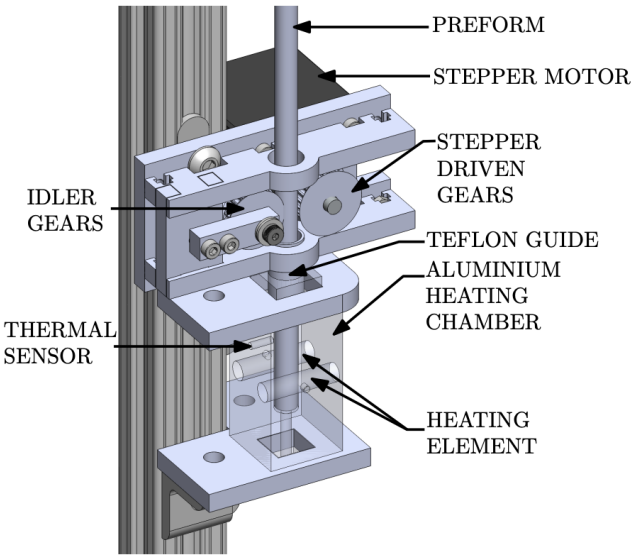


Figure 4-2: Extruder sub-assembly of FrED [11]

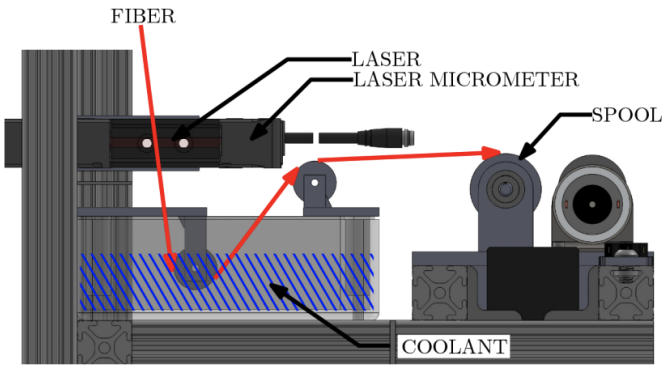


Figure 4-3: Laser micrometer and cooling system sub-assembly of FrED [11]

then into water that serves as a coolant.

The fiber is pulled and spooled through the spooling system. The DC motor rotates the spool. A lead screw connected to a stepper motor allows the spool to move side to side so the fibers can be collected on the entire spool.

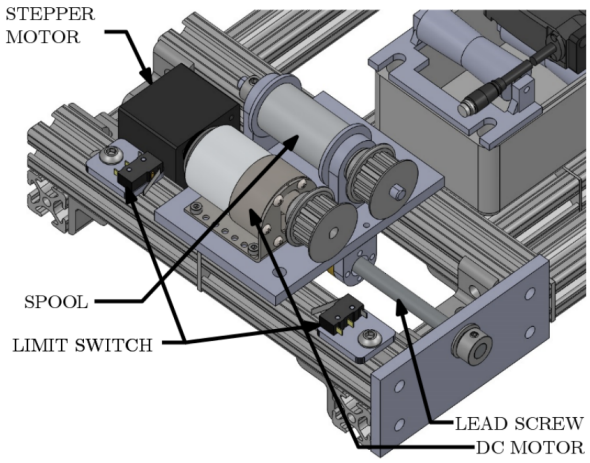


Figure 4-4: Spooling system sub-assembly of FrED [11]

Using Teensy micro-controllers, Arduino, and Robot Operating System (ROS), the motors and temperature of the heating element can be controlled. Data is recorded on a computer interfaced with FrED. Each manufactured spool represents one dataset.

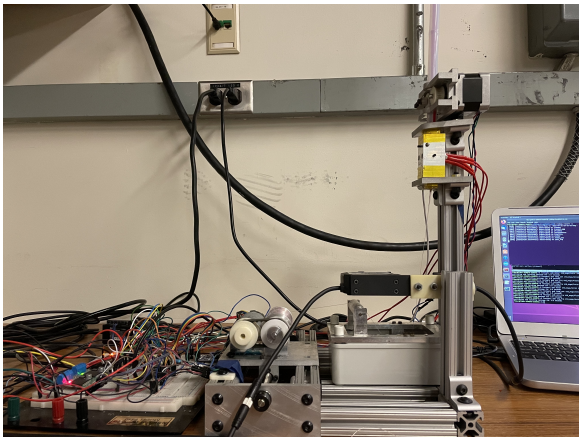


Figure 4-5: Image of FrED

4.2 Materials, Parameters, Data Collection

For Material 1, AdTech generic 0.28 inch diameter all-purpose glue sticks were used as preform. The temperature of the heating element was set to 82°C.

For Material 2, Bettomshin EVA based 0.28 inch diameter glue sticks were used as preform. The temperature of the heating element was set to 77°C.

Continuous data was collected with the inputs being extrusion stepper motor frequency (ext_frq) and spindle motor speed (sdl_spd). The fiber diameter was measured as the output using the laser micrometer. For each diameter measurement, the timestep was logged.

For Material 1, data was collected at ext_frq of 8, 9, 10, 11, 12, 13, 14, 15, 16, and 17. At each extrusion motor frequency, the sdl_spd was run at 65, 70, 80, 90, 100, 110, 120, 130, 135, 140, 145 and 150. In addition, four spools of data were collected at an ext_frq of 13 and sdl_spd of 100.

For Material 2, two spools of data were collected for training. The first spool was set at ext_frq of 13 and the sdl_spd was varied at 65, 70, 80, 90, 100, 110, 120, 130, 140 and 150. The second spool was set at sdl_spd of 100 and the ext_frq was varied at 8, 9, 10, 11, 12, 13, 14, 15, 16, and 17.

Two additional spools were collected for testing. The first spool varied between ext_frq of 8, 13, and 17 and sdl_spd of 65, 100, and 150. The second spool was set at a constant ext_frq of 13 and sdl_spd of 100.

Data was formatted into csv files with columns of timestep, ext_frq, sdl_spd and fiber diameter.

Chapter 5

Developing LSTM RNN Model for FrED

This chapter details the steps taken to develop an LSTM RNN source model for Material 1 fiber data and is split into three sections: Data Visualization, Model Creation, and Model Output.

5.1 Data Visualization

Material 1 data was visualized to detect patterns and correlation between parameters and rule out unsuitable models. Data visualization also helped with determining the shape of the input and the hyperparameters of the model.

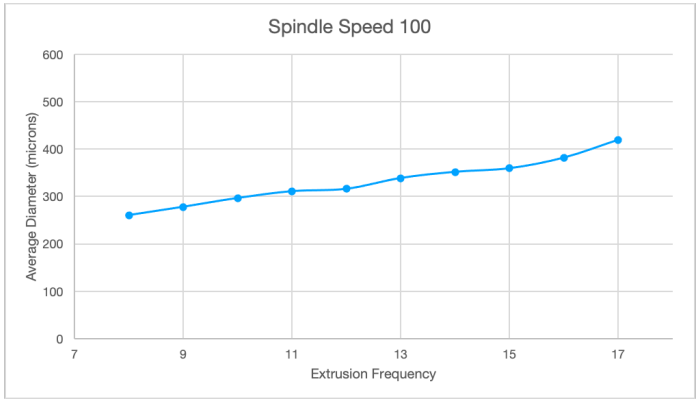


Figure 5-1: Diameter vs. Extrusion Frequency for a constant Spindle Speed

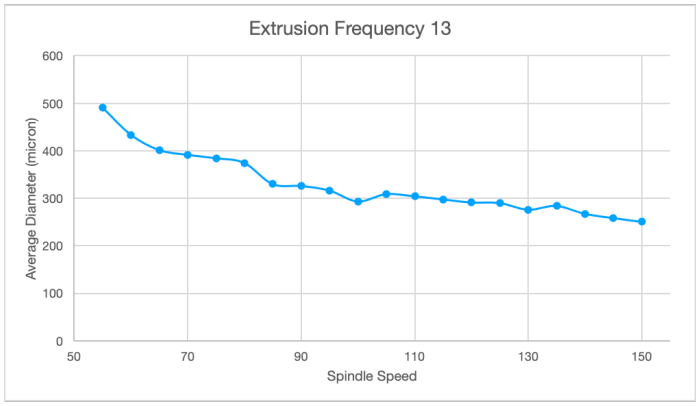


Figure 5-2: Diameter vs. Spindle Speed for a constant Extrusion Frequency

5-1 shows the effect of extrusion frequency on diameter. As extrusion frequency increases, diameter increases. 5-2 shows the effect of spindle speed on diameter. The inverse relationship between spindle speed and diameter can be modeled using physics. Kim previous implemented a theoretical physics-based model that predicts the fiber diameter of FrED [12]. The trend follows the theoretical conservation of mass as shown in Eqn. 5.1 where v_2 is the linear velocity of the fiber at the spool, v_1 is the linear velocity of the fiber at the heating chamber exit (related to ext_frq), A_1 is the cross-sectional area of the heating chamber exit ($20.7mm^2$ for FrED), and A_2 is the cross-sectional area of the fiber at the spool.

$$v_2 = v_1 \cdot \frac{A_1}{A_2} \tag{5.1}$$

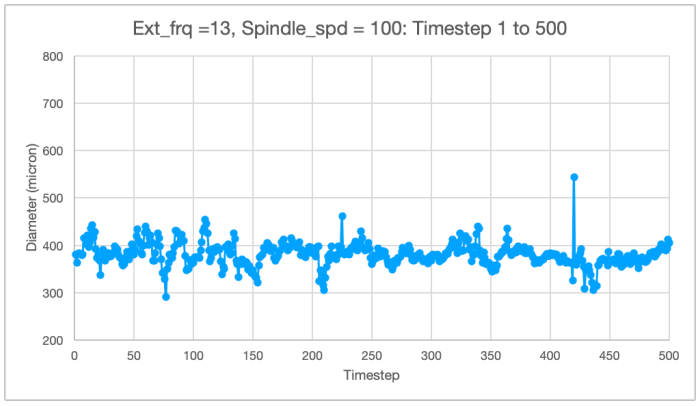


Figure 5-3: Diameter vs. Time from timestep 1 to 500 for constant input parameters

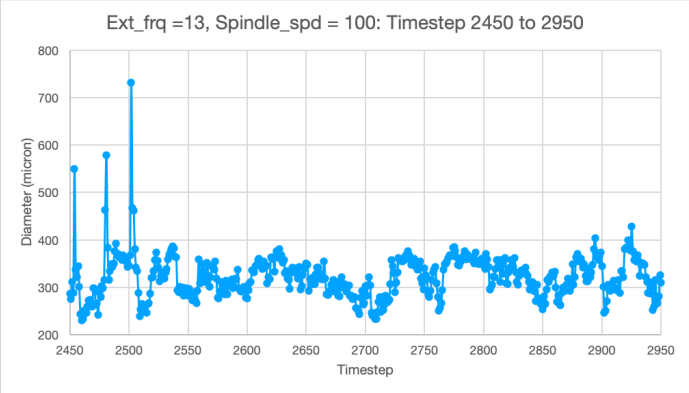


Figure 5-4: Diameter vs. Time from timestep 2450 to 2950 for constant input parameters

Table 5.1: Effect of Time on Diameter Measurement

Timestep Range	**Average Diameter (microns)**
1 to 500	381.0
2450 to 2950	322.3

5-3 and 5-4 show the difference in diameter measurements at different timestep ranges with the same parameters. As time increases the diameter decreases. Table 5.1 shows a 58.7 micron diameter difference between timestep 1 to 500 and timestep 2450 and 2950. The inverse relationship between time and diameter is the result of the spool becoming larger over time, which in turn increases the linear velocity of the fiber at the spool. The fiber gets pulled at a faster velocity despite the system being set at a constant spindle speed (constant angular velocity). Eqn. 5.2 explains the role the diameter of the spool has on v_2 where D_s is the diameter of the spool including the wound up fiber and ω_s is the angular velocity of the spool in revolutions per second. Eqn. 5.2 shows the proportional relationship between D_s and v_2. Combining Eqn. 5.1 and Eqn. 5.2 results in Eqn. 5.3 which calculates the cross-sectional area of the fiber at the spool as the output. An increase in either D_s or ω_s will lead to a decrease in A_2 and thus fiber diameter.

$$v_2 = \pi \cdot D_s \cdot \omega_s \tag{5.2}$$

$$A_2 = \frac{v_1 \cdot A_1}{\pi \cdot D_s \cdot \omega_s} \tag{5.3}$$

Both 5-3 and 5-4 show outliers in the dataset. Due to errors caused by the laser micrometer, data greater than 500 microns are due to mismeasurement. Lines 31-35 in A.1 shows data cleaning. In addition, outliers were manually removed through visual inspection.

5.2 Model Creation

Google Colab was used to develop the model [6]. The complete list of libraries imported are shown from Lines 6 to 15 in A.1 and include pandas, NumPy, TensorFlow, and Keras libraries [20] [19] [23] [10].

An input size (INPUT_SIZE as shown in line 47 of A.1) of 2950 timesteps was chosen as it represents data collected for one spool. Other input sizes such as 500 timesteps were tested. However, as shown in the data visualization, time plays a role in fiber diameter and an input size of 2950 timesteps was chosen to allow the model to detect long-term sequencing. Twelve datasets of 2950 were used for training and three datasets of 2950 were used for testing thus leading to a 80/20 split between training and testing data.

DRL previously tested four LSTM architecture designs for developing its ML model on the optical fiber extrusion process: single layer LSTM, single layer biLSTM, side-by-side LSTM, and deep LSTM [2].

DRL found that the deep LSTM architecture performed the best and thus a deep LSTM network was chosen [2]. The deep LSTM network consists of three 128 node LSTM layers, a fully connected layer of 100 nodes and a final regression layer as shown in lines 79 to 86 of A.1.

Hyperparameters of the model include number of epochs and batch size for training and are shown in lines 89-94 of A.1. Epoch hyperparameter values of 10, 15, 20, and 30 were tested. A batch size (batch_size as shown on line 93 of A.1) of four was chosen for training. This is because twelve datasets are used for training and a batch

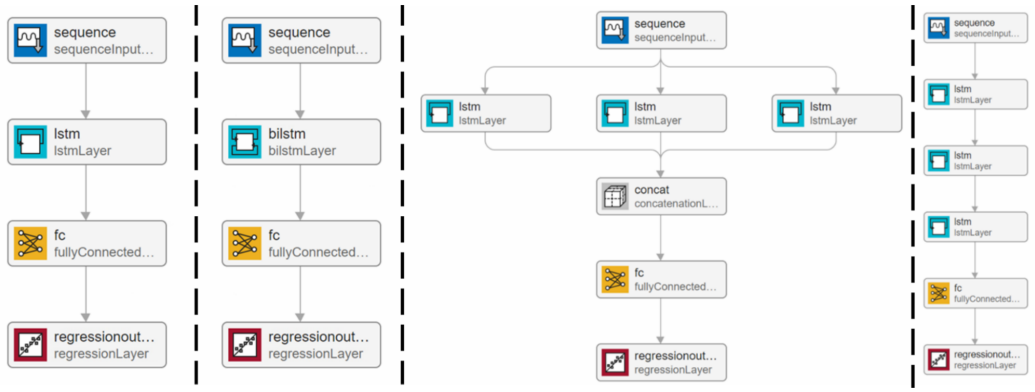
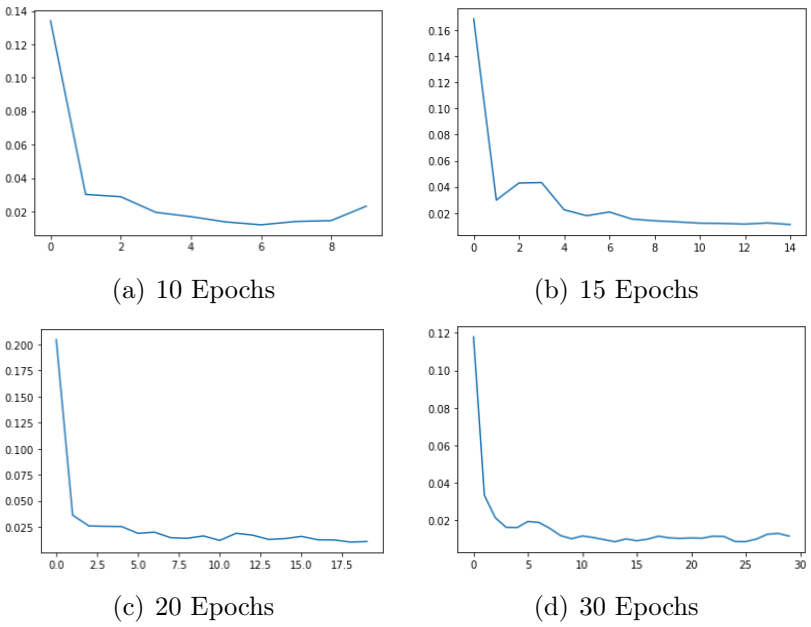


Figure 5-5: From Left to Right: Single Layer LSTM, Single Layer biLSTM, Side-by-side LSTM, Deep LSTM [2]

size of four will allow for training to be done three times per epoch.



(a) 10 Epochs

(b) 15 Epochs

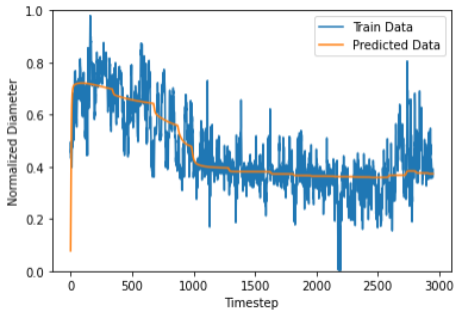
(c) 20 Epochs

(d) 30 Epochs

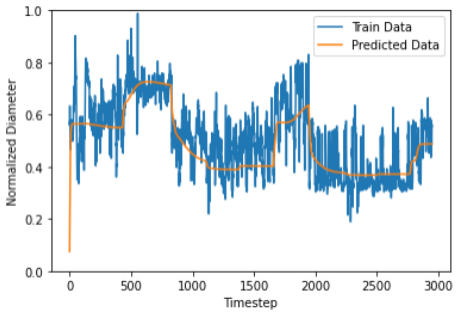
Figure 5-6: Loss (y-axis) vs. Epoch (x-axis) for Various Epoch Hyperparameter Values

5-6(a) shows a high final loss for 10 epochs. Thus, an epoch hyperparameter value of 10 produces an undertrained model. The loss graph for 15, 20 and 30 epochs shows a low final loss. 5-6 shows 15 epochs is the inflection point the minimum loss occurs. Hence, an epoch hyperparameter of 15 was chosen. The final loss for 15 epochs (5-6(b)) was 0.0111.

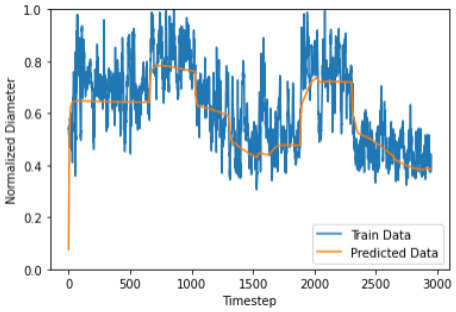
5.3 Model Output



(a) 1st Train Dataset



(b) 2nd Train Dataset

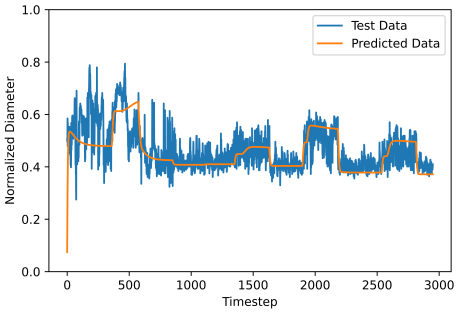


(c) 3rd Train Dataset

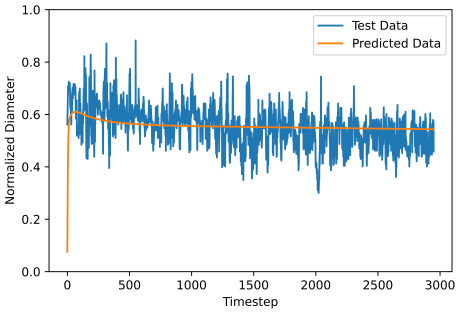
Figure 5-7: Train vs. Predicted for Material 1

Table 5.2: Input Parameters for Respective Training Datasets

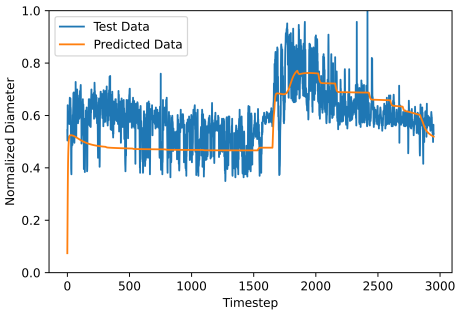
	Train 1 Parameters		Train 2 Parameters		Train 3 Parameters	
Timestep	**Sdl_spd**	**Ext_frq**	**Sdl_spd**	**Ext_frq**	**Sdl_spd**	**Ext_frq**
0	65	11	100	12	100	15
200	65	11	100	12	100	15
400	70	11	100	12	100	15
600	70	11	65	12	100	15
800	80	11	65	12	65	15
1000	100	11	110	12	65	15
1200	100	11	150	12	110	15
1400	110	11	120	12	150	15
1600	110	11	120	12	120	15
1800	120	11	70	12	120	15
2000	135	11	130	12	65	15
2200	140	11	130	12	70	15
2400	145	11	145	12	130	15
2600	150	13	135	12	130	15
2800	120	13	90	12	145	15



(a) 1st Test Dataset



(b) 2nd Test Dataset



(c) 3rd Test Dataset

Figure 5-8: Test vs. Predicted for Material 1

Table 5.3: Input Parameters for Respective Test Datasets

	Test 1 Parameters		Test 2 Parameters		Test 3 Parameters	
Timestep	**Sdl_spd**	**Ext_frq**	**Sdl_spd**	**Ext_frq**	**Sdl_spd**	**Ext_frq**
0	100	10	100	13	130	13
200	100	10	100	13	130	13
400	65	10	100	13	135	13
600	110	10	100	13	140	13
800	110	10	100	13	145	13
1000	150	10	100	13	145	13
1200	140	10	100	13	150	13
1400	90	10	100	13	150	13
1600	90	10	100	13	150	16
1800	145	10	100	13	65	16
2000	70	10	100	13	70	16
2200	134	10	100	13	90	16
2400	135	10	100	13	90	16
2600	80	10	100	13	110	16
2800	80	10	100	13	120	16

5-7 shows the LSTM RNN model run on three of the twelve training datasets. 5.2 shows the input parameters that FrED was set to for the 2950 timesteps. 5-7(a), 5-7(b), and 5-7(c) show that the model is able to predict fluctuations in sdl_spd well for training data but 5-7(a) shows the model could improve on predicting the effect of ext_frq. In 5-7(a) when ext_frq increased from 11 to 13 between timestep 2400 and 2600, the model was not able to accurately predict a large increase in diameter.

5-8 shows the LSTM RNN model run on the three test datasets. 5.3 shows the input parameters that FrED was set to for the 2950 timesteps. 5-8(a) shows that the LSTM RNN model does not overfit the training data as it is able to accurately predict fluctuations in sdl_spd for new test data. 5-8(b) shows that the model is also able to predict the effect of the increasing angular velocity of the spool over time for constant parameters. Both the test data and the model show a decreasing trend over time for diameter. 5-8(c) shows that the model is able to predict Δdiameter caused by a Δext_frq well but can improve on predicting the diameter with a given ext_frq. The MSE between predicted and test data for the model was 0.00622. This chapter shows that LSTM RNNs are able to sufficiently model the input and output parameters of FrED and detect long-term patterns. The next chapter implements transfer learning on the Material 1 model created in this chapter.

Chapter 6

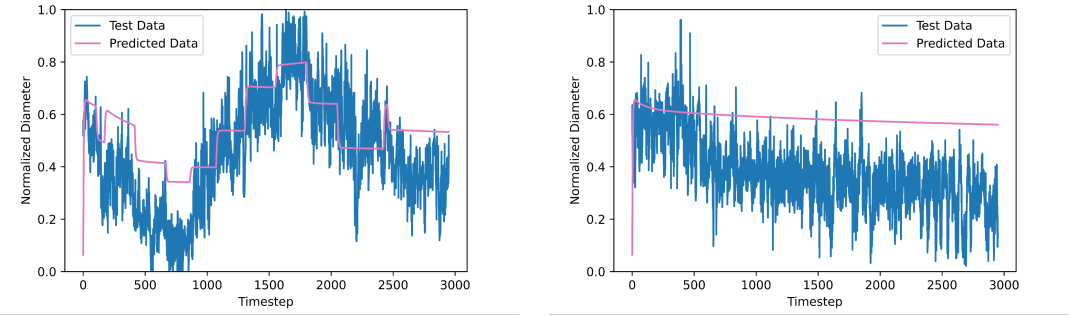
Applying Transfer Learning for Different Material

This chapter details the steps taken to develop four LSTM RNN models that predict Material 2 data. Predicted Data 1 uses the Material 1 model developed in the previous chapter and does not utilize the Material 2 training data. Predicted Data 2 was developed without the use of Material 1 model/training data and only with the relatively small amount of Material 2 training data collected. Predicted Data 3 was developed by fine-tuning the Material 1 model with Material 2 train data. Predicted Data 4 was developed by transferring the first two LSTM layers of the Material 1 model and training the last 3 layers with Material 2 train data. Testing these four LSTM RNN models gives insight on the best method to auto-generate an ML model for a prospective fiber manufacturing company.

6.1 Predicted Data 1 (PD1): Testing Material 1 Model on Material 2 Data

Material 2 data was tested using the Material 1 model to assess the difference between material properties. It is hypothesized that the model will be accurate for materials with similar properties to those of Material 1 and the model will lose accuracy for

materials with dissimilar properties to those of Material 1. The parameters used for both test datasets are shown in 7.2.



(a) 1st Test Dataset (b) 2nd Test Dataset

Figure 6-1: Test vs. PD1

The MSE between predicted and test data was 0.0440. 6-1(a) depicts a significant difference in material properties as the predicted diameter by the model at timestep 750 is much higher than the diameter measurements of the Material 2 test data. At most timesteps in 6-1(b), the predicted diameter is also larger than the Material 2 test data. This could be due to Material 2 producing a smaller diameter than Material 1 on average for the same parameters. The diameter of Material 2 fiber was on average 30.8 microns smaller than that of Material 1 fiber at an extrusion frequency of 13 and spindle speed of 100.

6.2 Predicted Data 2 (PD2): Developing Model with Material 2 Train Data

A new LSTM RNN model was developed for Material 2. Two training datasets on Material 2 were used to develop a LSTM RNN model, which was then tested on two Material 2 test datasets. The use of a small dataset in training the model is representative of a situation in which a prospective client chooses to develop a model on their own. The source code to develop the model is shown in A.2.

The model was run with hyperparameters of batch_size = 1 and epochs = 15. The final loss was 0.0121. The MSE was 0.0292 between test and predicted. The similarly

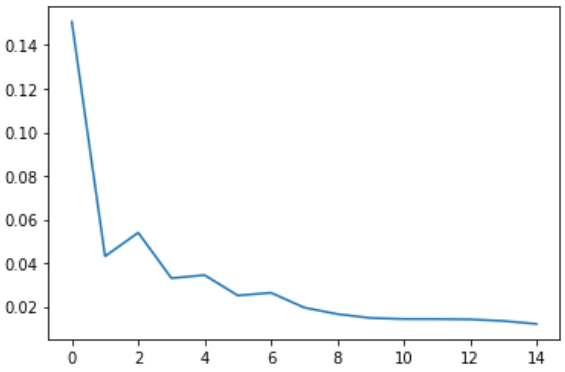
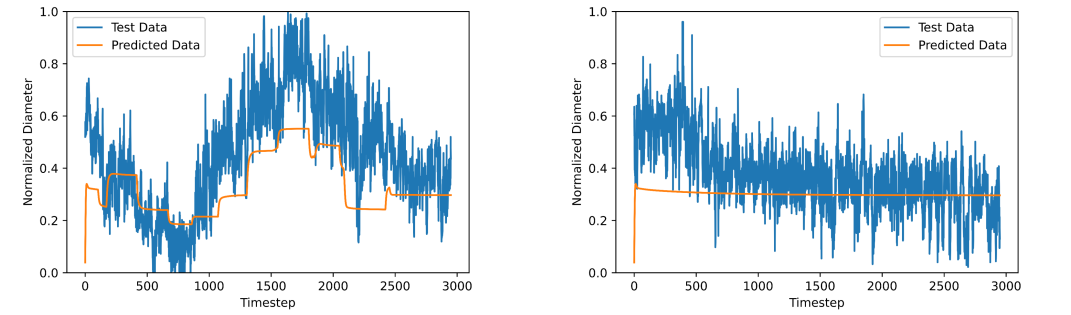


Figure 6-2: Loss (y-axis) vs. Epoch (x-axis) for PD2 Model



(a) 1st Test Dataset

(b) 2nd Test Dataset

Figure 6-3: Test vs. PD2

low final loss of the PD2 model (0.0121 for PD2 and 0.0111 for Material 1 model) shows the model's ability to predict training data well. However, the relatively high MSE of the PD2 model compared to the Material 1 model shows the PD2 model's inability to predict test data well. This shows that the PD2 model might be suffering from overfitting as there is not enough training data given to the model to develop an understanding of the true trend between parameters.

6.3 Predicted Data 3 (PD3): Fine-tuning Material 1 Model with Material 2 Train Data

Fine-tuning is a technique where all layers of a previously trained model are retrained with new data at a low learning rate. Material 1 model was loaded and retrained with

hyperparameters of batch_size = 1 and epochs = 15 as shown in A.3. The final loss was 0.011 and the MSE between test and predicted was 0.0215.

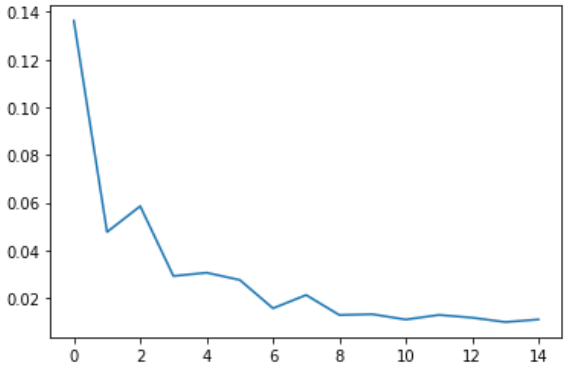
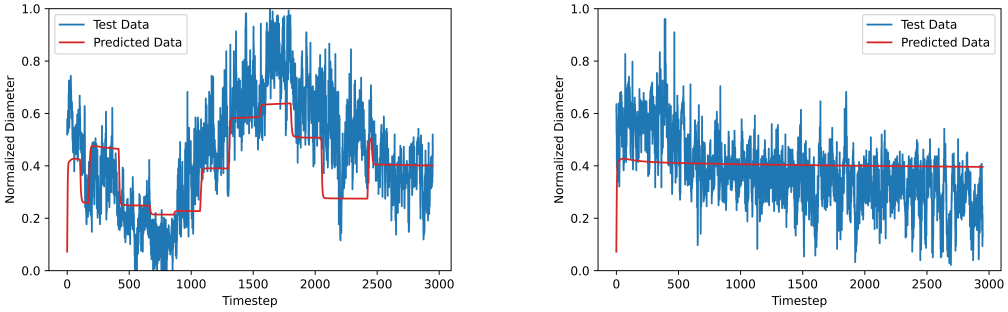


Figure 6-4: Loss (y-axis) vs. Epoch (x-axis) for PD3 Model



(a) 1st Test Dataset

(b) 2nd Test Dataset

Figure 6-5: Test vs. PD3

6.4 Predicted Data 4 (PD4): Developing Model using Transfer Learning

The optimal number of layers needed to be transferred was found to be the first two LSTM layers. Transferring three of the LSTM layers resulted in a higher MSE value. As shown in 6-6, the first two layers of the Material 1 model were transferred. A.4 shows the hyperparameters and the source code of the model. Through experimentation of various hyperparameters, batch_size was set to 1, and number of epochs

was set to 30. The final loss was 0.009 and the MSE between test and predicted was 0.0175.

Layer (type)	Output Shape	Param #	Trainable
lstm (LSTM)	(None, 2950, 128)	67072	N
lstm_1 (LSTM)	(None, 2950, 128)	131584	N
lstm_2 (LSTM)	(None, 2950, 128)	131584	Y
dense (Dense)	(None, 2950, 100)	12900	Y
dense_1 (Dense)	(None, 2950, 1)	101	Y

Total params: 343,241
Trainable params: 144,585
Non-trainable params: 198,656

Figure 6-6: LSTM RNN Layers Transferred

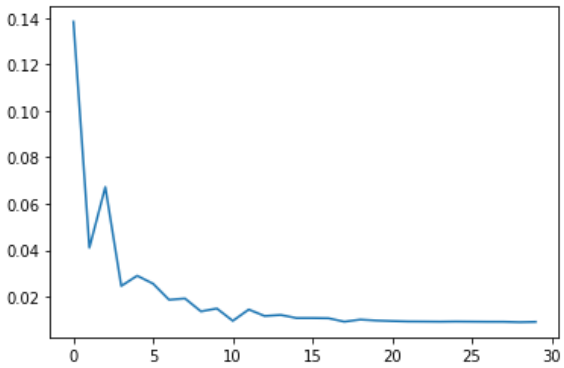
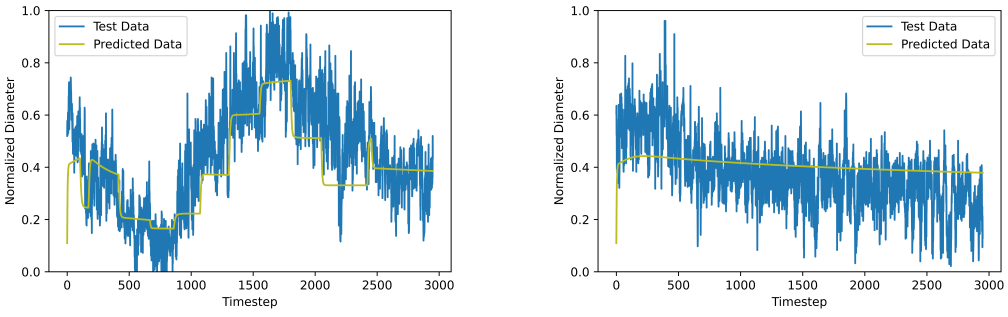


Figure 6-7: Loss (y-axis) vs. Epoch (x-axis) for PD4 Model



(a) 1st Test Dataset

(b) 2nd Test Dataset

Figure 6-8: Test vs. PD4

Chapter 7

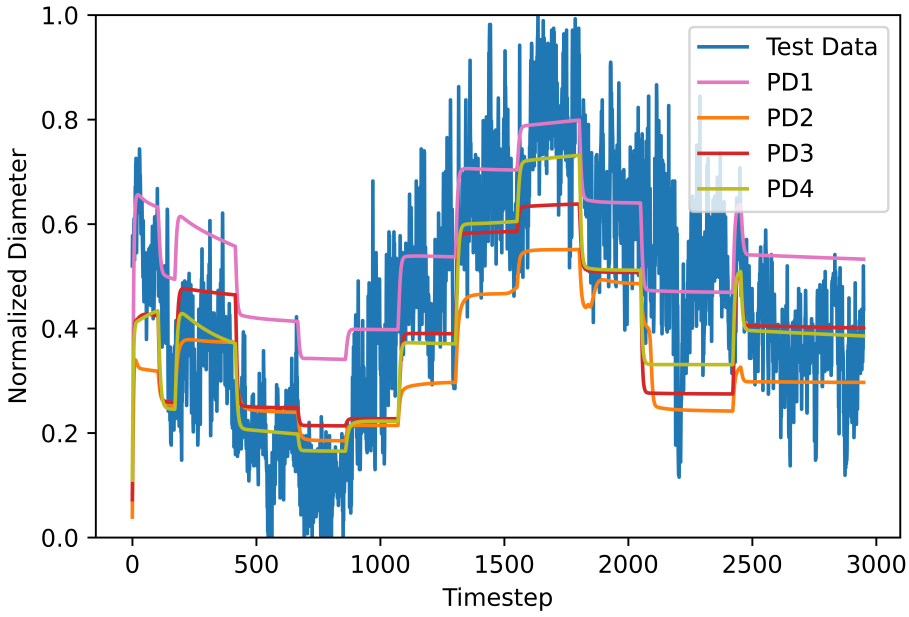
Results

Table 7.1: A Comparison of MSE Values across Models

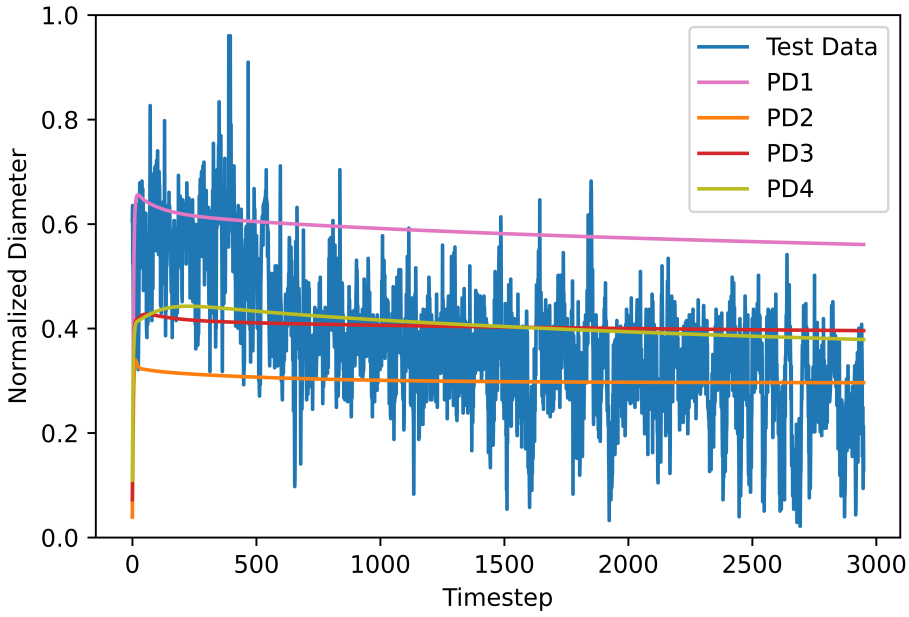
Model	**# of Input Batches**	**Test Data**	**# of Test Batches**	**MSE**	**Graph Label**
Material 1	12	Material 1	3	0.00622	–
Material 1	12	Material 2	2	0.044	PD 1
Material 2	2	Material 2	2	0.0292	PD 2
Fine-tuning	12 + 2	Material 2	2	0.0215	PD 3
Transfer Learning	12 + 2	Material 2	2	0.0175	PD 4

The MSE of PD1 (testing Material 1 model on Material 2 test data) was seven times higher than that of testing Material 1 model on Material 1 test data. PD4 had the lowest MSE value out of all models tested on Material 2 data. PD3, which implemented fine-tuning, showed an MSE value that was 26.4% lower than that of PD2 which does not implement fine-tuning or transfer learning. PD4, which implemented transfer learning, showed an MSE that was 40.4% lower than that of PD2.

For test data 1 as shown in 7-1(a), PD4 can more accurately predict fiber diameter than PD1 at timestep 750 while also better predicting fiber diameter than PD2 at timestep 1700. For test data 2 as shown in 7-1(b), PD4 is able to better predict fiber diameter than both PD1 and PD2 as PD1 generally over-predicts fiber diameter and PD2 generally under-predicts fiber diameter.



(a) 1st Test Dataset



(b) 2nd Test Dataset

Figure 7-1: Test vs. PD1, PD2, PD3, PD4

Table 7.2: Input Parameters for Material 2 Test Datasets

	Test 1 Parameters		Test 2 Parameters	
Timestep	**Sdl_spd**	**Ext_frq**	**Sdl_spd**	**Ext_frq**
0	100	13	100	13
200	65	8	100	13
400	65	8	100	13
600	100	8	100	13
800	150	8	100	13
1000	150	13	100	13
1200	100	13	100	13
1400	65	13	100	13
1600	65	17	100	13
1800	65	17	100	13
2000	100	17	100	13
2200	150	17	100	13
2400	150	17	100	13
2600	100	13	100	13
2800	100	13	100	13

Chapter 8

Conclusion

8.1 Recommendation

It can be inferred that Material 1 is considerably different than Material 2 based on the Material 1 model's failure to accurately predict the fiber diameter of Material 2. The large decrease in MSE values when implementing transfer learning and fine-tuning demonstrates the effectiveness of these techniques. Transfer learning performed significantly better than fine-tuning and thus should be the preferred approach when developing a commercial product for this technology. This study illustrates that a prospective fiber manufacturing company would be able to improve the MSE of their model by 40.4% using the product. Sales decks and the study should be pitched to prospective clients to gauge their interest and understand the utility of the solution to various companies. Such information will provide valuable insight into the optimal pricing structure for the subscription model.

8.2 Future Work

DRL researcher, Shreya Dhar is currently working on improving FrED to make the device easier to use and more functional. Dhar is working on a tension measuring device installed between the cooling system and the spool. The device measures the tension at each timestep that the fiber is spooled. This would enable an additional

input for the LSTM RNN model and could significantly improve the model accuracy. In addition, the improved FrED can manufacture larger spools, significantly increasing the input data size from 2950 timesteps. This in turn would allow the LSTM RNN model to detect long-term patterns and sequencing. DRL researcher, Russel Bradley is implementing a digital microscope and computer vision to optically measure fiber diameter on FrED. The digital microscope can also measure opacity of the fiber, which can serve as a second output in addition to fiber diameter for the LSTM RNN model. Additionally, more power needs to be transmitted to the heating element to allow FrED to extrude materials with high melting points such as Nylon. DRL has previously extruded Nylon on different devices but the heating element subassembly and electronics need to be implemented on the current version of FrED that was used for this thesis. Next, a spinneret system would need to be developed to imitate the aramid manufacturing process more accurately.

Collecting more data with a greater variety of input parameters will help develop a robust generic fiber draw ML-model that can be used with transfer learning to solve specific problems. However, it should be noted that for larger datasets, an effective data cleaning algorithm will need to be developed to remove outliers. DRL Researcher, Mohamed Othman is working on implementing ML-models on a conventional PLC (Allen-Bradley CompactLogix 5480 PLC shown in 1-4) to control the fiber diameter of FrED. Finally, this study and a pitch deck for the proposed business solution need to be presented to venture capital investors for feedback and guidance on the next steps to advance this technology.

The following outlines a provisional plan to launch this technology as a product:

1. Conduct further exploration on the theoretical physics and mechanics of the fiber draw process.

2. Develop a robust source model on MIT's full-scale fiber draw system that implements physics-based modeling with LSTM RNNs.

3. Develop preliminary transfer learning infrastructure that auto-generates ML models.

I Create an algorithm that finds optimal hyperparameters through programmatically training and testing data.

II Integrate software on cloud servers that allow customers to input fiber draw data securely.

III Test software with few beta partners to verify the capability of transfer learning in a real-life setting.

4. Market service to fiber manufacturing companies once the transfer learning infrastructure is shown to work effectively.

Appendix A

Source Code

```
from google.colab import drive
drive.mount('/content/drive')
import os
os.chdir("drive/My Drive/Data/CSVFiles")  #import Data

import pandas as pd
import numpy as np
import lightgbm
from IPython.display import display
from sklearn.model_selection import train_test_split, GroupKFold,
    KFold
from sklearn.metrics import mean_absolute_error
from tensorflow import keras
import tensorflow as tf
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt

one_header = ['ext13-16diam','extfrq9-13diam', '
    mtr100allextfrq2diam', 'mtr100allextfrqdiam','
    mat1ext13mtr100temp82diam','mat1ext13mtr100temp82diam2', '
    mat1ext13mtr100temp82diam3','mat1ext13mtr100temp82diam4']
two_header = ['ext11diam', 'extfrq10-14diam', 'extfrq12diam', '
    extfrq13allmtrspddiam', 'extfrq15diam', 'extfrq17diam', '
    extfrq8diam']
```

```
19
20 dataframes_list = []
21
22
23 for i in range(len(one_header)):
24     temp_df = pd.read_csv(f'{one_header[i]}.csv', header=1)
25     dataframes_list.append(temp_df)
26
27 for i in range(len(two_header)):
28     temp_df = pd.read_csv(f'{two_header[i]}.csv', header=2)
29     dataframes_list.append(temp_df)
30
31 def clean_df(df):
32   df = df.rename(columns=lambda x: x.strip().lower())
33   df = df.drop(columns=["field.htr_pwm", "field.temp_ma", "field.
      encv", "time", "%time", "field.temp_set","field.temp"], errors='
      ignore')
34   df = df.rename(columns = {'field.diam':'diam', 'Motor':'motor', '
      ext_frq':'ext'})
35   df = df[(df['motor'] > 0) & (df['ext'] > 0) & (df['diam'] > 0) &
      (df['diam'] <= 500)]
36
37   df.dropna()
38   return df
39
40 dfs = list(map(clean_df, dataframes_list))
41 test = [dfs[9],  dfs[5], dfs[3]]
42 dfs.pop(9)
43 dfs.pop(6)
44 dfs.pop(3)
45 train = dfs
46
47 INPUT_SIZE = 2950
48 train_data = []
49 test_data = []
50 for df in train:
```

```
  i = 0
  while i + INPUT_SIZE < df.shape[0]:
    train_data.append(df[i:i+INPUT_SIZE])
    i += INPUT_SIZE

for df in test:
  i = 0
  while i + INPUT_SIZE < df.shape[0]:
    test_data.append(df[i:i+INPUT_SIZE])
    i += INPUT_SIZE

train_data = np.array(train_data)
print(train_data.shape )

test_data = np.array(test_data)
print(test_data.shape)

diam_min = np.min(train_data[:, :, 0])
diam_max = np.max(train_data[:, :, 0])
print(diam_min)
print(diam_max)

diam_range = diam_max - diam_min
train_data[:, :, 0] = (train_data[:,:,0] - diam_min)/diam_range
test_data[:, :, 0] = (test_data[:,:,0] - diam_min)/diam_range

train_data.shape

model = keras.models.Sequential([
    keras.layers.Input(shape=(2950,2)),
    keras.layers.LSTM(128, return_sequences=True),
    keras.layers.LSTM(128, return_sequences=True),
    keras.layers.LSTM(128, return_sequences=True),
    keras.layers.Dense(100, activation='relu'),
    keras.layers.Dense(1),
])
```

```
model.compile(optimizer="adam", loss="mse")

history = model.fit(
    x=train_data[:,:,1:],
    y=train_data[:,:,0],
    epochs=15,
    batch_size=4,
)

plt.plot(history.epoch, history.history['loss'], label='total loss')
plt.show()

pred = model.predict(test_data[:, :, 1:])

mse = ((pred.reshape(len(test_data),2950) - test_data[:, :, 0])**2).mean(axis=None)
print(mse)

x = np.arange(stop=2950)
plt.plot(x, test_data[0, :, 0], color ='tab:blue')
plt.plot(x, pred[0, :], color ='tab:orange')
plt.ylim([0, 1])
plt.show()
plt.savefig('test_0.png', dpi=1200)

for i in range(0,2950,200):
  print(f't={i} {test_data[0, i, 1:]}')

x = np.arange(stop=2950)
plt.plot(x, test_data[1, :, 0], color ='tab:blue')
plt.plot(x, pred[1, :], color ='tab:orange')
plt.ylim([0, 1])
f = plt.figure()
f.set_figwidth(15)
f.set_figheight(15)
```

```
plt.show()

for i in range(0,2950,200):
  print(f't={i} {test_data[1, i, 1:]}')

x = np.arange(stop=2950)
plt.plot(x, test_data[2, :, 0], color ='tab:blue')
plt.plot(x, pred[2, :], color ='tab:orange')
plt.ylim([0, 1])
f = plt.figure()
f.set_figwidth(15)
f.set_figheight(15)
plt.show()

for i in range(0,2950,200):
  print(f't={i} {test_data[2, i, 1:]}')

train_pred = model.predict(train_data[:,:,1:])
x = np.arange(stop=2950)
for i in range(10):
  plt.plot(x, train_data[i, :, 0], color ='tab:blue')
  plt.plot(x, train_pred[i, :], color ='tab:orange')
  plt.ylim([0, 1])
  f = plt.figure()
  f.set_figwidth(15)
  f.set_figheight(15)
  plt.show()
```

Source Code A.1: Material 1 LSTM RNN Model

```

from google.colab import drive
drive.mount('/content/drive')

import os
os.chdir("drive/My Drive/Data/CSVFiles/Material2")  #import Data

```

```
import pandas as pd
import numpy as np
import lightgbm
from IPython.display import display
from sklearn.model_selection import train_test_split, GroupKFold, KFold
from sklearn.metrics import mean_absolute_error
from tensorflow import keras
import tensorflow as tf
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt

one_header = ['mat2ext13allmtrspeeddiam','mat2ext8_13_17mtr65_100_150diam', 'mat2ext13mtr100temp77diam', 'mat2mtr100allextfrqdiam']

dataframes_list = []


for i in range(len(one_header)):
    temp_df = pd.read_csv(f'{one_header[i]}.csv', header=1)
    dataframes_list.append(temp_df)

def clean_df(df):
  df = df.rename(columns=lambda x: x.strip().lower())
  df = df.drop(columns=["field.htr_pwm", "field.temp_ma", "field.encv", "time", "%time", "field.temp_set","field.temp"], errors='ignore')
  df = df.rename(columns = {'field.diam':'diam', 'Motor':'motor', 'ext_frq':'ext'})
  df = df[(df['motor'] > 0) & (df['ext'] > 0) & (df['diam'] > 0) & (df['diam'] <= 500)]

  df.dropna()
  return df

```

```
dfs = list(map(clean_df, dataframes_list))
test = [dfs[1], dfs[2]]
dfs.pop(2)
dfs.pop(1)
train = dfs

INPUT_SIZE = 2950
train_data = []
test_data = []
for df in train:
  i = 0
  while i + INPUT_SIZE < df.shape[0]:
    train_data.append(df[i:i+INPUT_SIZE])
    i += INPUT_SIZE

for df in test:
  i = 0
  while i + INPUT_SIZE < df.shape[0]:
    test_data.append(df[i:i+INPUT_SIZE])
    i += INPUT_SIZE

train_data = np.array(train_data)
print(train_data.shape )

test_data = np.array(test_data)
print(test_data.shape)

diam_min = np.min(train_data[:, :, 0])
diam_max = np.max(train_data[:, :, 0])

diam_range = diam_max - diam_min
train_data[:, :, 0] = (train_data[:,:,0] - diam_min)/diam_range
test_data[:, :, 0] = (test_data[:,:,0] - diam_min)/diam_range

model = keras.models.Sequential([
    keras.layers.Input(shape=(2950,2)),
```

```
    keras.layers.LSTM(128, return_sequences=True),
    keras.layers.LSTM(128, return_sequences=True),
    keras.layers.LSTM(128, return_sequences=True),
    keras.layers.Dense(100, activation='relu'),
    keras.layers.Dense(1),
])
model.compile(optimizer="adam", loss="mse")

history = model.fit(
    x=train_data[:,:,1:],
    y=train_data[:,:,0],
    epochs=15,
    batch_size=1,
)

plt.plot(history.epoch, history.history['loss'], label='total loss')
plt.show()

pred = model.predict(test_data[:, :, 1:])

mse = ((pred.reshape(len(test_data),2950) - test_data[:, :, 0])**2).mean(axis=None)
print(mse)

x = np.arange(stop=2950)
plt.plot(x, test_data[0, :, 0], color ='tab:blue', label="Test Data")
plt.plot(x, pred[0, :], color ='tab:orange', label="Predicted Data")
plt.ylim([0, 1])
plt.xlabel("Timestep")
plt.ylabel("Normalized Diameter")
plt.legend()
plt.show()

```

```
for i in range(0,2950,200):
  print(f't={i} {test_data[0, i, 1:]}')

x = np.arange(stop=2950)
plt.plot(x, test_data[1, :, 0], color ='tab:blue', label="Test Data")
plt.plot(x, pred[1, :], color ='tab:orange', label="Predicted Data")
plt.ylim([0, 1])
plt.xlabel("Timestep")
plt.ylabel("Normalized Diameter")
plt.legend()
plt.show()
f = plt.figure()
f.set_figwidth(15)
f.set_figheight(15)


for i in range(0,2950,200):
  print(f't={i} {test_data[1, i, 1:]}')
```

Source Code A.2: Material 2 LSTM RNN Model

```

model = keras.models.load_model('/content/drive/MyDrive/Models/model1')
history = model.fit(
    x=train_data[:,:,1:],
    y=train_data[:,:,0],
    epochs=15,
    batch_size=1,
)
```

Source Code A.3: Replaced Source Code for Fine Tuning LSTM RNN Model

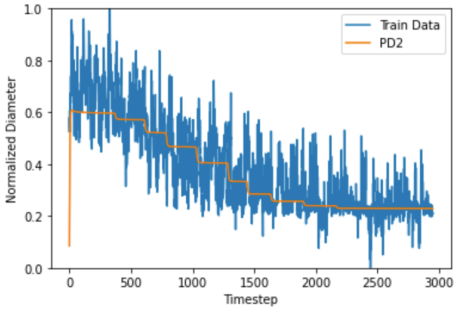
```

model_m1m_tl_lstm = keras.models.load_model('/content/drive/MyDrive/Models/model1')
```

```
model_m1m_tl_lstm.get_layer(name="lstm").trainable = False
model_m1m_tl_lstm.get_layer(name="lstm_1").trainable = False
model_m1m_tl_lstm.get_layer(name="dense").trainable = True
model_m1m_tl_lstm.get_layer(name="dense_1").trainable = True
model_m1m_tl_lstm.get_layer(name="lstm_2").trainable = True
model_m1m_tl_lstm.summary(
    show_trainable=True
)

history = model_m1m_tl_lstm.fit(
    x=train_data[:,:,1:],
    y=train_data[:,:,0],
    epochs=30,
    batch_size=1,
)

pred = model_m1m_tl_lstm.predict(test_data[:, :, 1:])
```

Source Code A.4: Replaced Source Code for Transfer Learning LSTM RNN Model

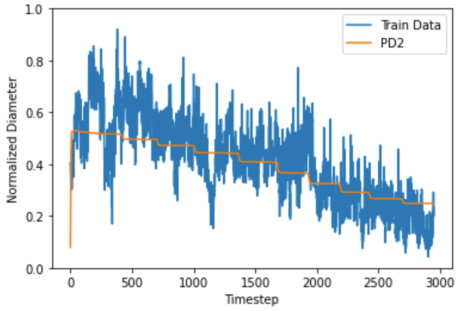
Appendix B

Figures




```
t=timestep [sdl_spd. ext_frq.]
t=0 [65. 13.]
t=200 [65. 13.]
t=400 [70. 13.]
t=600 [70. 13.]
t=800 [90. 13.]
t=1000 [90. 13.]
t=1200 [100.  13.]
t=1400 [110.  13.]
t=1600 [120.  13.]
t=1800 [130.  13.]
t=2000 [140.  13.]
t=2200 [150.  13.]
t=2400 [150.  13.]
t=2600 [150.  13.]
t=2800 [150.  13.]
```

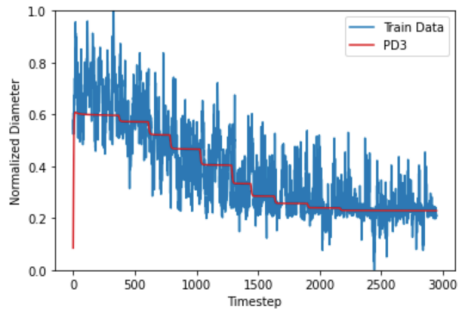
(a) 1st Train Dataset




```
t=timestep [sdl_spd. ext_frq.]
t=0 [100.  17.]
t=200 [100.  17.]
t=400 [100.  17.]
t=600 [100.  16.]
t=800 [100.  15.]
t=1000 [100.  15.]
t=1200 [100.  14.]
t=1400 [100.  13.]
t=1600 [100.  13.]
t=1800 [100.  12.]
t=2000 [100.  11.]
t=2200 [100.  10.]
t=2400 [100.  10.]
t=2600 [100.   9.]
t=2800 [100.   8.]
```

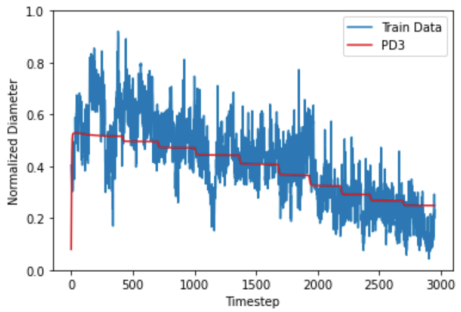
(b) 2nd Train Dataset

Figure B-1: Train vs. Predicted Data 2 for Material 2




```
t=timestep [sdl_spd. ext_frq.]
t=0 [65. 13.]
t=200 [65. 13.]
t=400 [70. 13.]
t=600 [70. 13.]
t=800 [90. 13.]
t=1000 [90. 13.]
t=1200 [100.  13.]
t=1400 [110.  13.]
t=1600 [120.  13.]
t=1800 [130.  13.]
t=2000 [140.  13.]
t=2200 [150.  13.]
t=2400 [150.  13.]
t=2600 [150.  13.]
t=2800 [150.  13.]
```

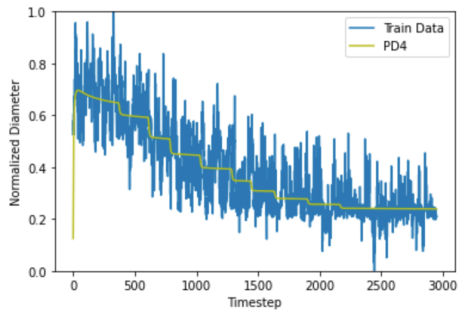
(a) 1st Train Dataset




```
t=timestep [sdl_spd. ext_frq.]
t=0 [100.  17.]
t=200 [100.  17.]
t=400 [100.  17.]
t=600 [100.  16.]
t=800 [100.  15.]
t=1000 [100.  15.]
t=1200 [100.  14.]
t=1400 [100.  13.]
t=1600 [100.  13.]
t=1800 [100.  12.]
t=2000 [100.  11.]
t=2200 [100.  10.]
t=2400 [100.  10.]
t=2600 [100.   9.]
t=2800 [100.   8.]
```

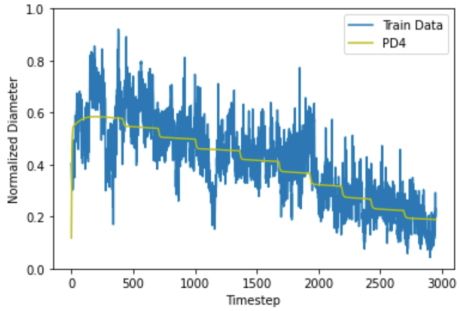
(b) 2nd Train Dataset

Figure B-2: Train vs. Predicted Data 3 for Material 2




```
t=timestep [sdl_spd. ext_frq.]
t=0 [65. 13.]
t=200 [65. 13.]
t=400 [70. 13.]
t=600 [70. 13.]
t=800 [90. 13.]
t=1000 [90. 13.]
t=1200 [100.  13.]
t=1400 [110.  13.]
t=1600 [120.  13.]
t=1800 [130.  13.]
t=2000 [140.  13.]
t=2200 [150.  13.]
t=2400 [150.  13.]
t=2600 [150.  13.]
t=2800 [150.  13.]
```

(a) 1st Train Dataset




```
t=timestep [sdl_spd. ext_frq.]
t=0 [100.  17.]
t=200 [100.  17.]
t=400 [100.  17.]
t=600 [100.  16.]
t=800 [100.  15.]
t=1000 [100.  15.]
t=1200 [100.  14.]
t=1400 [100.  13.]
t=1600 [100.  13.]
t=1800 [100.  12.]
t=2000 [100.  11.]
t=2200 [100.  10.]
t=2400 [100.  10.]
t=2600 [100.   9.]
t=2800 [100.   8.]
```

(b) 2nd Train Dataset

Figure B-3: Train vs. Predicted Data 4 for Material 2

Bibliography

[1] Jan Paul Assendorp. *Deep learning for anomaly detection in multivariate time series data*. PhD thesis, Hochschule für Angewandte Wissenschaften Hamburg, 2017.

[2] George C Chen. A data-driven approach to system dynamics modeling and control design. Master's thesis, Massachusetts Institute of Technology, 2022.

[3] S Roy Choudhury, Yogesh Jaluria, and S HK Lee. Generation of neck-down profile for furnace drawing of optical fiber. Technical report, American Society of Mechanical Engineers, New York, NY (United States), 1995.

[4] Stefania Cristina. Calculus in action: Neural networks. Web, August 2022.

[5] Nitisha Dubey. Leading fiber optic companies in the world. Web, July 2021.

[6] Google. Welcome to colaboratory - google research. Web. research.google.com/colaboratory/.

[7] Larry Hardesty. Explained: neural networks. *MIT News*, 14, 2017.

[8] IBM. What is a digital twin? Web.

[9] National Instruments. Pid theory explained. Web, March 2020.

[10] Keras. Keras api reference. Web. keras.io/api/.

[11] David Donghyun Kim. *Design and development of desktop fiber and fabric manufacturing system for advanced materials*. PhD thesis, Massachusetts Institute of Technology, 2020.

[12] David Donghyun Kim and Brian Anthony. Design and fabrication of desktop fiber manufacturing kit for education. In *Dynamic Systems and Control Conference*, volume 58295, page V003T31A005. American Society of Mechanical Engineers, 2017.

[13] Sangwoon Kim et al. *Model-free tracking control of an optical fiber drawing process using deep reinforcement learning*. PhD thesis, Massachusetts Institute of Technology, 2020.

[14] Sangwoon Kim, David Donghyun Kim, and Brian W Anthony. Dynamic control of a fiber manufacturing process using deep reinforcement learning. *IEEE/ASME Transactions on Mechatronics*, 27(2):1128–1137, 2021.

[15] Microcontrollers Lab. Pid controller implementation using arduino. Web.

[16] Navin Kumar Manaswi, Navin Kumar Manaswi, and Suresh John. *Deep learning with applications using python*. Springer, 2018.

[17] Aditi Mittal. Understanding rnn and lstm. Web, October 2019.

[18] Norman S Nise. *Control systems engineering*. John Wiley & Sons, 2020.

[19] NumPy. Numpy documentation. Web, 2022. numpy.org/doc/stable/.

[20] Pandas. pandas documentation. Web, June 2023. pandas.pydata.org/docs/.

[21] Yurui Qu, Li Jing, Yichen Shen, Min Qiu, and Marin Soljacic. Migrating knowledge between physical scenarios based on artificial neural networks. *ACS Photonics*, 6(5):1168–1174, 2019.

[22] Rfindustries. Fiber optic cable types – multimode and single mode. Web, January 2019.

[23] TensorFlow. Tensorflow core v2.9.1. Web, August 2022. www.tensorflow.org/api_docs/python/tf.

[24] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.