A predictive model for Nafion-Based IPMC Soft Actuators

Bachelor's Project Thesis

Ruben Langius, s2583992, r.g.langius@student.rug.nl,
Supervisors: prof. dr. R. Carloni, R. D'Anniballe, V. Krishna

Abstract: This research focuses on the development of a predictive model for a Nafion IPMC soft actuator. Nafion-117 is a synthetic polymer that is often researched for potential implementations in soft-robotics. For this research an actuator was made of Nafion-117, which was modelled with the help of a neural network. Before fabricating the final test samples, key variables that affected the actuator's performance as well as the optimum build technique were defined by rigorous testing. The neural network was built based on a feed-forward model. To train this neural network, a data set consisting of 80.000 force measurements of 2 test samples was created. The network was trained and optimised on this data set, after which the resulting network was tested on a separate data set that was collected using a separate third test sample. After training, the validation set returned a root mean square error of 0.042. Prediction on the test sample resulted in a root mean square error of 0.034. Therefore, it can be concluded that this model generalises for Nafion IPMC actuators.

1 Introduction

Ionic polymer-metal composites (IPMCs) are electro-active polymers that can be used as actuators in the field of soft robotics. Properties such as large deformations, low voltage, softness and self-sensing make IPMC a material that is useful for soft actuation [1]. One of the drawbacks of IPMCs is their need for hydration. Dehydrated samples offer less actuation and stability which makes it difficult to predict their actuation.

There have been multiple attempts to model the performance of an IPMC [2][3]. However, despite the research that has been conducted, there is currently no model that can accurately predict the displacement of an IPMC actuator.

The main focus of this paper is to develop a neural network model that is trained on measurements of the actuation of a Nafion-117 based IPMC. In order to create the data set, IPMC actuators must be fabricated.
The IPMCs has been fabricated by a chemical process that transforms the polymer into a polymer-metal composite. Using absorption/reduction cycles, both sides of the polymer have been plated with metal atoms such as platinum, gold or silver [4]. This plating process offers good adhesion between the metal particles and the membrane of the polymer. The result of this process is an even coverage of the metal plating across the membrane surface.

The central research question is 'Can the actuation of a Nafion-117 based IPMC be predicted with a neural network model'. To model a soft actuator correctly, a good data set is important. The data set needs to reflect every possible combination of input variables in order for the neural network to differentiate what the relations are. The quality of the data set also depends on the performance of the fabricated samples. Reproducible measurements reduce side effects in the data which again helps with the accuracy of the model. One of the last steps in building a successful model is fine-tuning the training parameters of the neural network. After tuning the neural network this results in a model of soft actuators that can be used to predict IPMC actuators.

2 Fabrication of the IPMC soft actuator

Several production steps are necessary to be able to start data collection for the neural network. Therefore, this section represents the work on fabricating the samples and the preliminary tests that were done in preparation of the measurements.

2.1 Fabrication

Most IPMC actuators are fabricated by a chemical process. This study uses the chemical method as described by De Luca et al. [5]. This method uses four steps to create the composite sandwich construction of the Nafion polymer and the electrodes:

1. Treating the surface of the Nafion. Both washing and roughening of the surface is necessary to increase the surface area and the inter-facial connection between the polymer and the metal electrodes.

2. Submersing the Nafion in a salt solution that contains the metal ions. This allows for absorption of the metal ions, in our case platinum, into the polymer.

3. In the third step the reduction of the absorbed metal ions to a metallic state nano-particle is established. The nano-particles are located on the surface of the Nafion and are the primary electrode.

4. The fourth and final step is the creation of the secondary plating. This plating increases the performance by thickening the electrodes and therefore lowering the resistance.

After fabrication the Nafion sheets are cut down to form samples of 65 millimetres in length and 5 millimetres wide. During preliminary tests it was observed that the surface roughening influenced the deflection movement during the dehydration phase. Several samples from the first batch of fabricated samples had different roughening patterns as the roughening was done by hand using sandpaper. This resulted in samples that are represented by sample A and B in figure 2.1.
The preliminary tests showed that the deflection during the dehydration of the IPMC seemed to be perpendicular to the roughening direction in those samples. Therefore the decision was made to use roughening in the longitudinal direction (C in figure 2.1) for fabricated samples in the future. These samples would deflect and slightly bend into a U-profile along the length of the sample, which would cause the sample to stiffen over its dehydration period. If a roughening pattern similar to sample B were used, the sample would not retain its shape over the dehydration period.

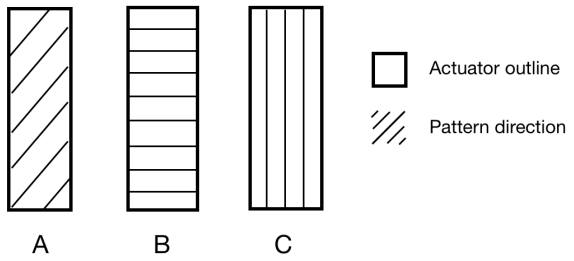

Figure 2.1: Top view of Nafion samples with different roughening patterns.

2.2 Data collection

Using the results from the preliminary tests, the requirements for the measurement setup were clear. The decision was made to measure force output over time in relation to:

- The voltage supplied to the actuator.
- The displacement of the actuator.
- Surface area of the actuator.

This resulted in a setup that could obtain data for 4 variables, by measuring the force over time in multiple iterations, with each iteration different settings for displacement and voltage.
With most variables fixed the only sensor that was needed was a load cell that logs the force measurements multiple times per second. With these requirements set out, it was clear what a potential setup would look like, as displayed in figure 2.2.

This setup has similarities with the setup for force-deflection characterisation as presented by

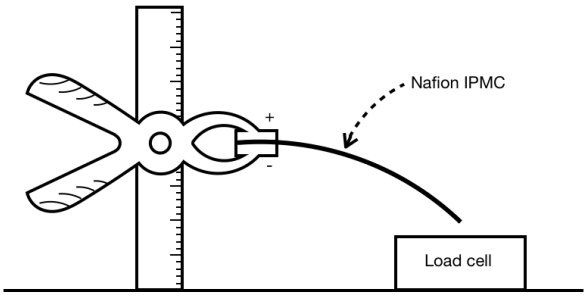

Figure 2.2: Cross section of a schematic setup that fulfils the initial requirements.

Carloni [6]. The key differences are that in this study force over time measurements were needed, and the same set of measurement needed to be repeated for multiple samples with different dimensions. Repeating the measurements for different samples introduces 2 possible problems.
The performance of a model built with deep learning is closely related to the amount of data it is trained on. The more data the better the performance [7]. However, operating an all manual setup measuring all that data with different samples would cost valuable time.
The second problem is the risk that human error would introduce variances across the samples and the different measurements. For each measurement the operator would need to dismount the sample, adjust the displacement, hydrate the sample for a set amount of time, remount the sample and start a new measurement. Even if the operator takes great care of detail it would be easy to influence the measurements, especially while repeating them hundreds of times.

To reduce the workload and improve measurement quality the Instron E1000 test instrument [8] is used. With the Instron control system it is possible to automate some of the operator's tasks such as timing the hydration period and adjusting the displacement.
Another helpful feature is automated data collection using sensors attached to the machine, this automatically logs all variable data with a 10Hz frequency.
To use the Instron instrument, a mount was required to measure the force produced by each sample. A clamp with variable clamping strength was already present. However, it was designed to clamp the material in a vertical direction while mounted to the Instron machine, whereas a horizontal direction was required. The solution was to 3D print a custom mount that transforms the vertical mount of the linear motor into a horizontal mounting position for the clamp. In figure 2.3 the complete setup is displayed with the custom mount displayed in red.

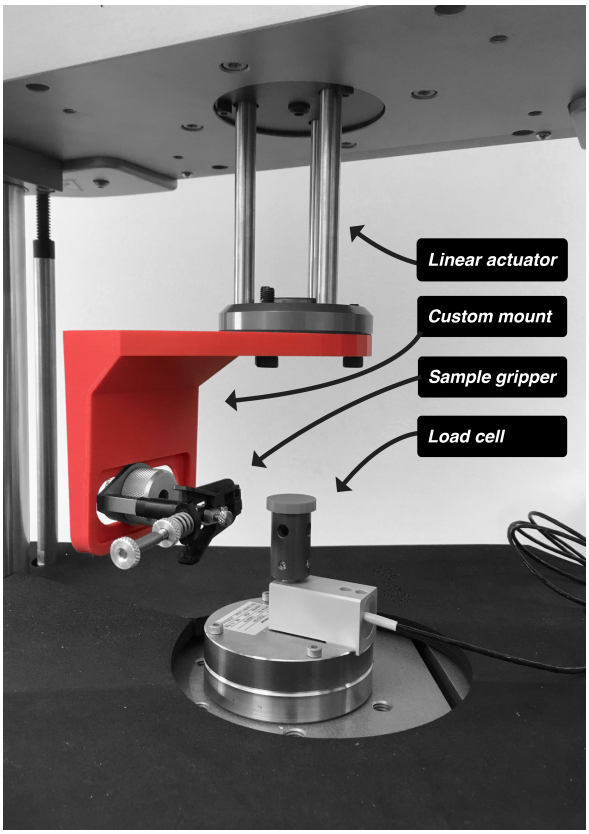

Figure 2.3: Full setup of the Instron E1000 test instrument with the custom mount in red.

As it can be seen in figure 2.3, the mount connects the gripper to the Instron machine so that the test samples can be tested against the load cell, which is mounted on the floor bed of the Instron machine. The height of the gripper in relation to the load cell is controlled by the linear actuator of the Instron machine.
Three working samples were fabricated with two of them having different dimensions. In the ex-

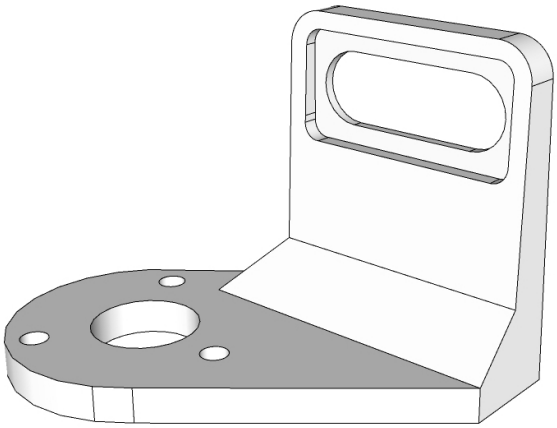

Figure 2.4: 3D rendering of the mount to connect a gripper to the Instron machine.

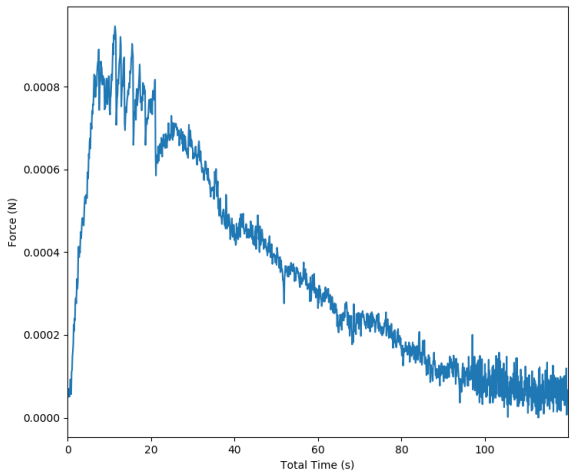

Figure 3.1: Results of sample 1, 14mm displacement and 2 volts applied.

ploratory tests it was established that voltages of 2, 3 and 4 volts were the most interesting to use, similar to what was established in previous research [4]. The setup was able to take accurate measurements for a displacement range of 0 to 20 mm. With increments of 2 mm this results in a total of 33 measurements for each sample. Using 3 samples and considering the necessary hydration periods between each measurement, this leads to a turnaround time of 2 weeks for fabrication and data collection.

3 Method

3.1 Data pre-processing

Figure 3.1 shows an example of the data produced from a measurement run, with a few notable features. Between 0 to 10 seconds there is an increase in the force that the sample is applying on the load cell, which is a key feature of how an IPMC actuates. The samples are not able to apply the maximum force instantly. After the maximum force is reached, the dehydration slope starts. In this slope, there is more noise compared to the start of the measurement. The noise in the measurements is created by the dehydration process in the samples, which causes small visible vibrations in the sample. To eliminate this noise, filtering was applied. It was decided to use a moving mean filter to filter out the noise, which removes peaks in a short time-frame and replace them with the mean of the surrounding points in time. The resulting filtered data allows for better prediction of a single point in time, as the neural network will not learn based on noise and is therefore less likely to output noise.

The moving mean filter was applied with an increasing time window that has a linear relation to time. The rate of filtering over time increases, which offsets the effect of the increasing noise by the dehydration vibrations.
In figure 3.2 the results of filtering are displayed. The amount of filtering was carefully minimized, since there is always the risk of losing data and therefore a loss of performance in the predictive model.
After filtering the data was scaled so that every variable fits in a domain of 0 to 1. This eliminates the need for the neural network to learn the conversion between every variable. For example, in the measurements time has a domain of 0 to 120 seconds, while force has a domain of 0 to 0.005 Newton. By scaling all variables to a domain of 0 to 1 the neural network does not need to do the conversion, and can focus on optimising the weights to create an optimal prediction.

After data processing we created the training and validation subset, using a standard 80 / 20

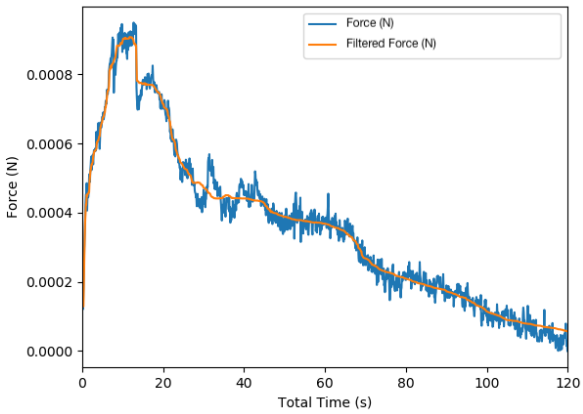

Figure 3.2: Moving mean filtering on the measured force.

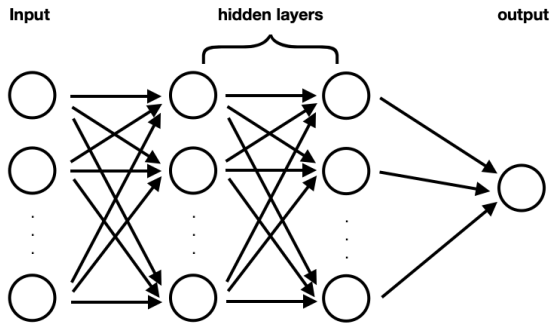

Figure 3.3: Multilayer perceptron.

percent split. The decision was made to split the data by complete measurement runs instead of individual data points. This ensured that the data in the validation set was missing in the training set and that the validation set could be used for its intended use, cross-validation.

3.2 Neural network design

The neural network is made on top of the framework Pytorch. Pytorch is a neural network framework built on top of the Torch engine. It is well suited for this research project due to its dynamic structure and relatively readable source code [9]. Another advantage of Pytorch is that users are able to use Python. Python is a well established programming language for data science purposes [10], which guarantees additional support of other data science packages as well as maintainability in the future.
Pytorch abstracts away a lot of the low level code that is necessary to run deep learning models. This high level approach results in faster development and a code base that is easier to be maintained by people with different backgrounds.

With Pytorch a feed-forward neural network is implemented. This feed forward model uses the multilayer perceptron (MLP) architecture. In this architecture all the nodes are fully connected across multiple hidden layers as pictured by figure 3.3.
A MLP network works with the following procedure. First an entry is read from the input data, then a forward pass is performed (loss function), next, gradients are computed for each of neurons (a backward pass), and finally the gradients are used to transform the weights of each neuron. To optimise a MLP network, the number of neurons and hidden layers can be altered. Another possibility is to use different activation and loss functions.

3.3 Training and testing

Pytorch has 2 main optimization algorithms for regression models. The first algorithm is stochastic gradient descent (SGD) [11]. The second method implements the Adam algorithm presented by Kingma and Lei Ba [12].
The Adam algorithm was originally introduced for its benefits in computation time, memory usage, less manual tweaking of parameters and faster convergence. After testing both optimisation algorithms, the Adam algorithm was chosen since it showed better performance in less time.

With Adam handling the batch gradient descent of the neural network, the focus was moved to the hyper-parameters of the network itself. Due to time limitations it was not possible to create an algorithm that would use a process similar to simulated annealing that would find the optimum system parameters. Instead we used the default values:

- 100 neurons for each hidden layer
- 2 hidden layers

- Sigmoid activation function
- MSE loss function

With these values a base performance for the neural network was obtained, after which the performance of the network could be optimised. The first parameter that was optimised is the number of neuron in each hidden layer. By increasing the number of neurons with steps of 100 hundred neurons it was found that the error of the network was lowest between 200 and 300. With increments of 10 neurons, the ideal number of neurons was found to be 250.

This process of using big interval steps followed by smaller, was repeated for all the other hyper-parameters, which resulted in a neural network with 250 neurons, 2 hidden layers, a reLU activation function and a smooth L1 loss function. During optimisation it was found that the network didn't improve after 20 epochs, future training cycles are therefore limited to 25 epochs, this delivers optimal performance in a time efficient manner.

4 Results

The network was trained for 25 epochs on the training data. From the 3 samples, 2 samples had different dimensions, which were used in training. The other sample is used as a test sample. In training a standard 80 to 20 percent training and validation split was used. This means that from the 66 collected measurements, 14 were used for cross-validation and the remaining 52 were used for training. Figure 4.1 shows the resulting loss graph generated during training.
Figure 4.1 shows a split between the validation and training data that starts at the eight epoch. This is a sign of overfitting on the training data, since the loss gains are not reflected in the validation data. Therefore, it can be argued that the best model is achieved on the eight epoch of training. To increase performance, dropout is used, which is a very effective method to preventing overfitting with the Adam optimisation algorithm [12]. With dropout, each training epoch a random selection of neurons in the neural network is selected and disabled from the network. By disabling these neurons, extra noise was created for the neural network, which decreases overfitting on the training data.

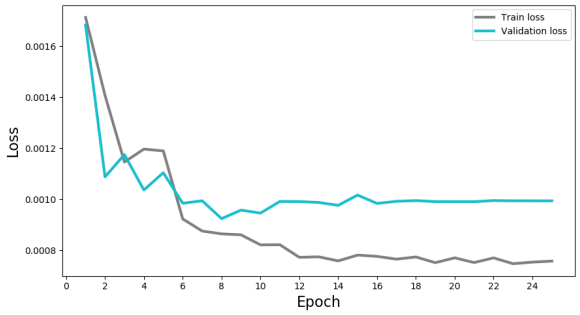

Figure 4.1: Loss function for both training and validation data as a result of epochs.

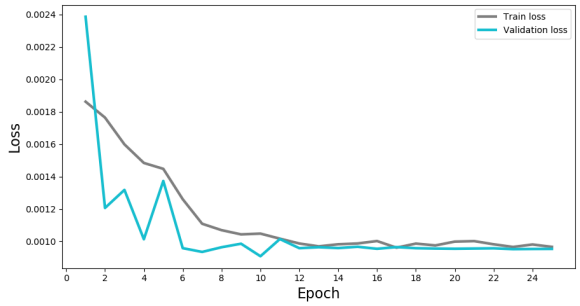

Figure 4.2: Loss of validation and training data with the dropout rate at 20 percent.

In figure 4.1 a dropout rate of 5 percent is used. When increased to 20 percent, overfitting is prevented as can be seen in figure 4.2. Not only is overfitting prevented, the model's performance is also increased with a lowest loss on the validation set on the tenth epoch.
The model's performance is measured using the root mean square error (RMSE), by comparing the measured forces in the validation data with the predicted forces from the model. The performance of this model is defined by its RMSE since we are trying to predict a number. In models that try to classify images or labels we can define accuracy with a simple true or false. In regression this doesn't work since the accuracy is defined by the difference between the prediction and target.
However, the output value of the RMSE can

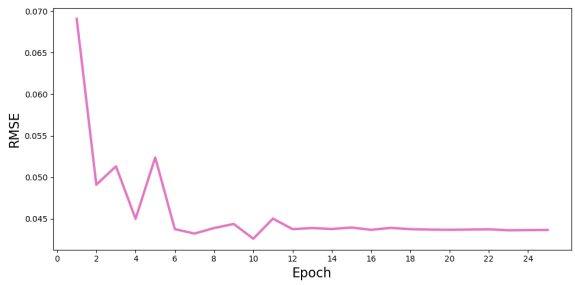

Figure 4.3: RMSE of the model on the validation data.

be compared with the domain of the value. In figure 4.3 it can be seen that the tenth epoch has the lowest RMSE of 0.042. To put that into perspective, the forces are scaled down to a range with a minimum of zero and a maximum of 1. Therefore a RMSE of 0.042 is only 4.2 percent of the total force range.

After training the model is also tested on the test sample. The third sample that was not used in training is from the same fabrication batch and has the same dimensions of one of the training samples. Therefore, a well performing model should be able to predict the force of this third sample. The model reached a RMSE of 0.034 on the test sample, which is compared to 0.042, 19 percent lower then the validation data. Therefore it can be concluded that the prediction model did indeed generalise for Nafion IPMCs. Another way of displaying the results is by overlapping the prediction by the neural network model with the actual measurements. In figure 4.4, the predicted force readings are displayed in orange, with the measurements rendered in blue.

5 Discussion

Previously, IPMC actuators were often characterised by researching one parameter and its influence on the output [2][3]. In this research the relationship between the different input variables and the output is investigated by building a predictive model. The model does not necessarily characterise the material in a mathematical way from which the individual relationships can be extrapolated. Neither does the performance of the predictive model reflect the accuracy of the relationships it created. For that reason this model build on a neural network is different from previous modelling attempts [5].
The new approach proposed int his research has the benefit that the model is easily transferable to different types of IPMCs, which allows for testing and comparing performance and stability between different fabrication methods [13] or control systems [14].

Combining our current methods with closed loop feedback control systems [14] would be an interesting subject for future research. Closed loop feedback would dramatically reduce the minimal time span for predicting. The current model predicts 120 seconds. With a closed loop feedback option this time can be reduced to only fill in the blanks between the feedback loops. The reduced complexity would certainly have to result in better accuracy and might just be the way forward to get IPMCs ready for real life implementations.

Since the predictive model was finished, several improvements have come up that are worth mentioning for future research. The first possible improvement is the most obvious one, it is simply to gather more data. This would benefit the model since it would reduce the split between the validation set and the test set. The validation set is too small with just 14 measurement runs, causing the performance of the validation set to be more sensitive to outliers.

Another improvement that is linked to data gathering, is the logging frequency. In the current setup the data is logged using the Instron test instrument with a frequency of 10 Hz. This frequency creates overlap between the individual data-points, therefore we need to split our data into training and validation set based on separate measurements, in order to have valid cross-validation. With new data collected in the future it would be possible to lower the logging frequency since it is not useful in the current implementation, and it would give us the opportunity to split the data into training and validations set by individual data-points.

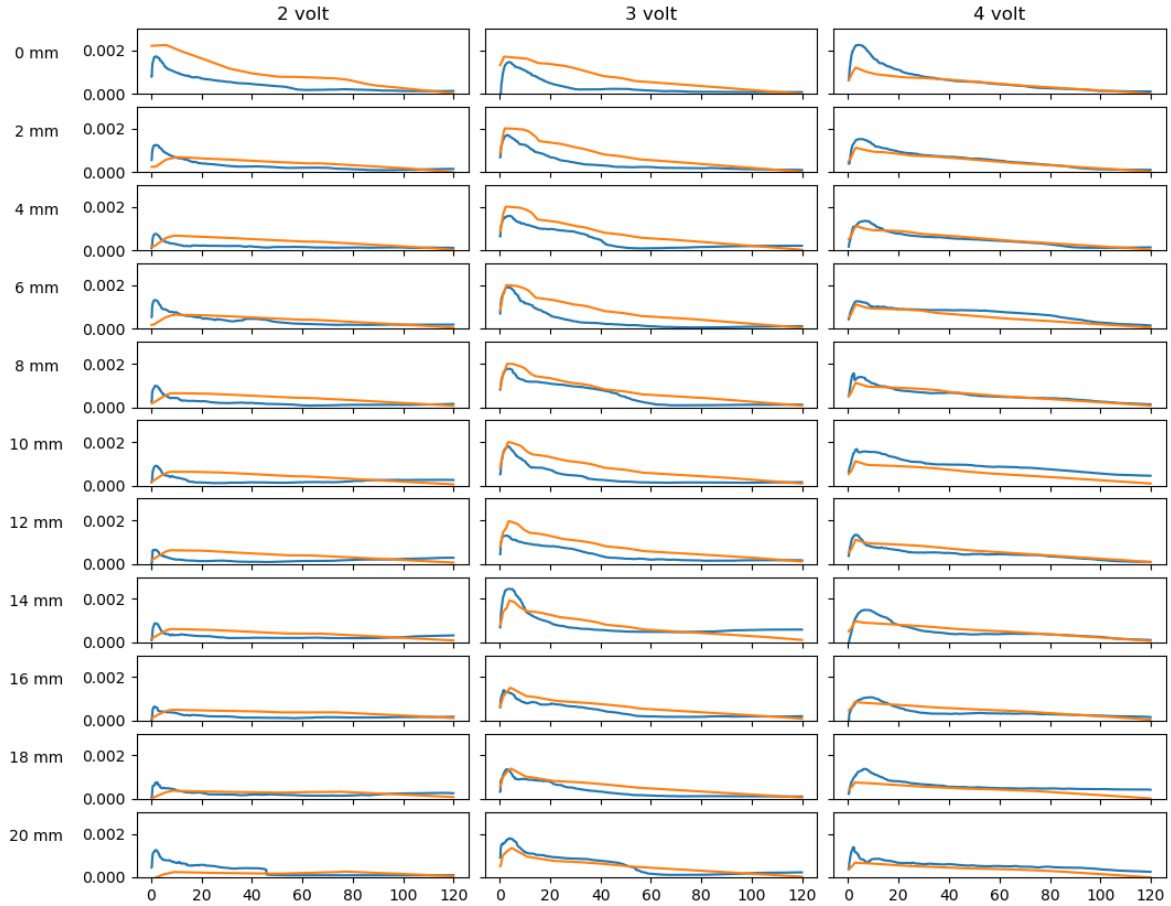

Figure 4.4: Representing all the measurements of the test sample. Each graph represents a force over time measurement with a specific distance and volt setting. The blue line is the measurement itself and the orange is the prediction by the model.

In this research project the decision to apply filtering on the force measurements was made, since the aim was to predict a single point in time. Therefore, it made sense to filter the force measurement to prevent the model form predicting noise. However, it is also possible to approach it the other way around. In this case you let the neural network learn everything, including the noise, and use filtering by not predicting a single point in time but multiple. Subsequently, filtering could be applied on these predictions.

It would be interesting to see if there is a performance difference between filtering before or after training. However, the real benefit of this alternative approach is that there is no risk of losing data that holds valuable information. A neural network can learn features from data that is just noise for the human eye, and sometimes noise is even added to increase the performance of the model [15]. It would therefore be useful to test this alternative approach to see whether it has any effect.

6 Conclusion

In this research 3 IPMC actuators have been fabricated that were used to model the production of force over time using a neural network. The measurement setup consisted of an Instron measurement machine that not only eliminated human error but also increased efficiency. This setup was used for 99 measurements runs in total. Each run consists of the force output over time for a specific voltage and distance. Each sample was tested with 2,3 and 4 volts with displacements between 0 and 20 mm. The resulting data set covers the force output over time of each sample across the specified domains of voltage and distance.

The neural network that was created is setup to handle multivariate regression problems like the Nafion IPMC force prediction. The neural network is built on the Pytorch platform and the optimum values for the key hyper parameters are 2 hidden layers with 250 neurons each, reLU activation function, 20 percent dropout and a smooth L1 loss function. With this network both the SGD and Adam [11][12] optimisation algorithms were tried. These algorithms optimise the weight adjustments during learning. The Adam algorithm performed better than SGD on this data-set. Its automatic and individual updates on the parameters allows for minimal setup work without losing performance. Moreover, the Adam algorithm outperformed the SGD algorithm with every possible parameter setup that was tested. Not only did it outperform SGD, it also converged much faster, resulting in less training time.

The final neural network was trained and validated on 2 out of 3 fabricated IPMCs. The 66 measurement runs across these 2 samples were split into a training and test set. The training set consisting of 80 percent of the data, was trained for 25 epochs, and validated on the other 20 percent. After training a root mean square error (RMSE) of 0.042 was achieved on the validation set.
The neural network was also tested using a third sample. This third sample is fabricated and measured identically to the other 2. The RMSE of the test sample is 0.034, which is 0.008 lower compared to the RMSE in the validation data. The smaller RMSE of the test sample compared to the validation data is the smaller size of the validation set. Since the validation set only contains 20 percent of the data, any outlier or other variance has a bigger impact on the RMSE of that set. The test set contains 33 measurement runs, therefore the RMSE on this set will be less susceptible for outliers or small variances.

Since performance is equal for both trained and non-trained samples, the conclusion is that the neural network model generalises for Nafion IPMC actuators. Technically this model could be implemented in existing robots or applications that use Nafion IPMCs.

References

[1] Mohsen Shahinpoor and Kwang J Kim. "Ionic polymer-metal composites: IV. Industrial and medical applications". In: *Smart Materials and Structures* 14.1 (Feb. 2005), pp. 197–214.

[2] Yoseph Bar-Cohen et al. "Characterization of the electromechanical properties of ionomeric polymer-metal composite (IPMC)". In: ed. by Yoseph Bar-Cohen. Vol. 4695. International Society for Optics and Photonics, July 2002, pp. 286–293.

[3] Kwangmok Jung, Jaedo Nam, and Hyoukryeol Choi. "Investigations on actuation characteristics of IPMC artificial muscle actuator". In: *Sensors and Actuators A: Physical* 107.2 (2003), pp. 183–192.

[4] Mohsen Shahinpoor and Kwang J Kim. "Ionic polymer-metal composites: I. Fundamentals". In: *Smart Materials and Structures* 10.4 (Aug. 2001), pp. 819–833.

[5] Viviana De Luca et al. "Ionic electroactive polymer metal composites: Fabricating, modeling, and applications of postsilicon smart devices". In: *Journal of Polymer Science Part B: Polymer Physics* 51.9 (May 2013), pp. 699–734.

[6] Raffaella Carloni et al. "A Variable Stiffness Joint With Electrospun P(VDF-TrFE-CTFE) Variable Stiffness Springs". In: *IEEE Robotics and Automation Letters* 3.2 (Apr. 2018), pp. 973–978.

[7] Chen Sun et al. *Revisiting Unreasonable Effectiveness of Data in Deep Learning Era.* 2017.

[8] *https://www.instron.us/products/testing-systems/dynamic-and-fatigue-systems/electropuls-systems.*

[9] Nikhil Ketkar. "Introduction to PyTorch". In: *Deep Learning with Python.* Berkeley, CA: Apress, 2017, pp. 195–208.

[10] Travis E. Oliphant. "Python for Scientific Computing". In: *Computing in Science & Engineering* 9.3 (2007), pp. 10–20.

[11] Ilya Sutskever et al. *On the importance of initialization and momentum in deep learning.* Tech. rep. 2013.

[12] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: (Dec. 2014).

[13] Hirohisa Tamagawa et al. "Bending control of Nafion-based electroactive polymer actuator coated with multi-walled carbon nanotubes". In: *Sensors and Actuators B: Chemical* 156.1 (Aug. 2011), pp. 375–382.

[14] Il-Seok Park et al. "Physical Principles of Ionic Polymerfffdfffdfffd Metal Composites as Electroactive Actuators and Sensors". In: *MRS Bulletin* 33.3 (Mar. 2008), pp. 190–195.

[15] G. Peter Zhang. "A neural network ensemble method with jittered training data for time series forecasting". In: *Information Sciences* 177.23 (Dec. 2007), pp. 5329–5346.