Deep Learning Assignment

MSC IN ARTIFICIAL INTELLIGENCE, NCSR-D

ENERI BONIAKOU (NTUA)

Domain

Plasma, often referred to as the fourth state of matter, is a collection of free charged particles moving in random directions and is electrically neutral. One way to form a plasma is using a generator to drive current through a gas (e.g. O_2, Ar, He, CF_4) confined between two parallel conducting plates/electrodes. Low pressure inductively coupled plasmas (ICPs) are widely used in the semiconductor manufacturing for etching and thin-film processing; about 40-45% of the steps to manufacture semiconductor devices use plasma [1]. Plasma etching is a key technique to create very small patterns on a material's surface in order to form transistors, interconnects, capacitors and more. Accurate prediction of etching rates is crucial for optimizing process parameters such us power and pressure, enabling a comprehensive understanding of how these affect the etching process. These predictions play a key role in enhancing productivity and reducing overall process time.

The case study that is investigated here, is PMMA etching by O_2 and Ar plasmas. Polymethylmethacrylate (PMMA) is a widely used polymer that has seen extensive application over the past 30 years. It is commonly found in products such as airplanes, automobiles, machinery, and electronic devices. Given that its broad range of uses often requires surface modifications, there is significant interest in studying the structural transformations that PMMA undergoes when exposed to UV light, electron or ion beams, and plasma [5].

In order to have the previous predictions, multidimensional physics-based simulations/models (PBMs) should be solved, but are computationally costly, as they solve extensive sets of partial differential equations for mass, energy and momentum balances alongside Maxwell's equations. An optimization algorithm requires multiple iterations of the computationally expensive PBM. Therefore, a model capable of making accurate predictions order of magnitude faster will be very helpful. What is interesting here, is having multiple neural-networks (NNs) for multiple gases with less amount of data required for the training of NNs from the source one. Moving from one gas to another, seems to be a feasible task using NNs as they serve as function approximators and the underlying equations that the PBM solves are the same.

Dataset and Resources

In this work, I have created two PBMs, one for Oxygen gas (O_2) and one for Argon gas (Ar) and the etchant substrate of both of them is Poly(methyl methacrylate) (PMMA) which is used as a resist in semiconductor processing. The O_2 gas consists of 156 reactions whereas the Ar chemistry is simpler, comprising only 40 reactions. These reactions include ionizations, dissociations and attachments. Regarding the run time, a single run of the PBM requires 40-50 min. The hardware configuration used for the runs of the PBM throughout this work consists of an AMD Ryzen Threadripper 3960X with 264GB of RAM. The PBMs have as inputs the power that is applied to the coil in order to drive current through the gas, and the pressure of the pump that is connected at the outlet of the ICP reactor. Figure 1 provides an illustration of the plasma discharge within the reactor and shows the simulation geometry. Table 1 shows the operating conditions of the two PBMs. Each PBM was run 4000 times and the two datasets consists of two inputs (power, pressure) and 10 outputs (etching rate along the radial distance of the PMMA).

Table 1: Operating conditions of the two models that generated the corresponding datasets.

	Power (Watts)	Pump Pressure (Torr)
O_2 Model	500-2000	0.02-0.07
Ar Model	50-500	0.01-0.06

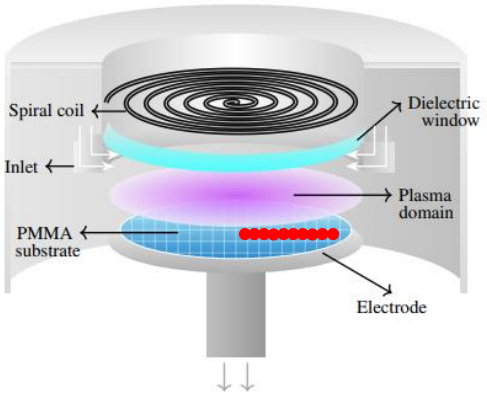

Figure 1: ICP reactor and is main components. The arrows show the gas flow. The power is applied to the spiral coil which creates electromagnetic waves which then induce current in the gas causing its dissociation, through which highly reactive chemical species are formed to start the etching of PMMA. The 10 red dots illustrate the place of the outputs of the PBM and therefore the outputs of the NN. As can be seen, the red dots are points on different radial distances along the substrate (blue disk). Experimental measurements also take place to these points in the real reactor. Measuring along the radial distance, gives us the uniformity of the etching rate.

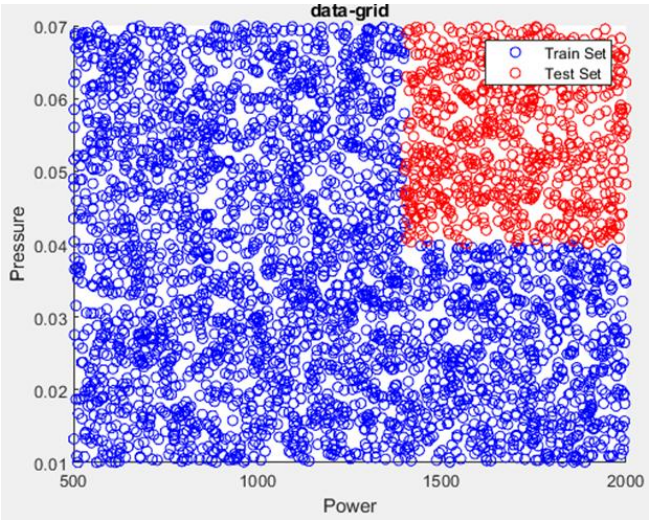

Figure 2: The input pairs used in the PBM runs to create the dataset. The pairs were generated with a random number generator and the testing dataset what chosen to be at the upper left corner – data that will be out of the range that the NN has seen.

The datasets created are saved as csv files with semicolon separator. They are divided as follows; 20% for testing and 80% for training (60 training + 20 validation). In figure 2 the input pairs (power, pressure) are plotted. More information about the PDE's that are solved in the PBMs and the chemistry reaction set behind them can be found from Refs [2], [3]. The hardware configuration used for the NNs training consists of an Intel Core i3-1115G4 with 16GB of RAM. In figures 3 and 4 the 3D plots of the training data are shown.

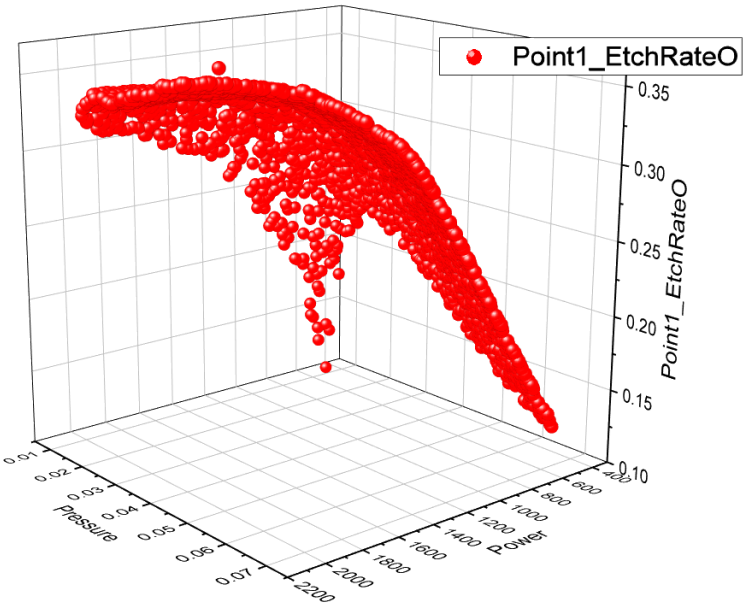

Figure 3: Etching rate (um/min) at the center of the wafer versus the pressure and the coil power for the O_2 plasma.

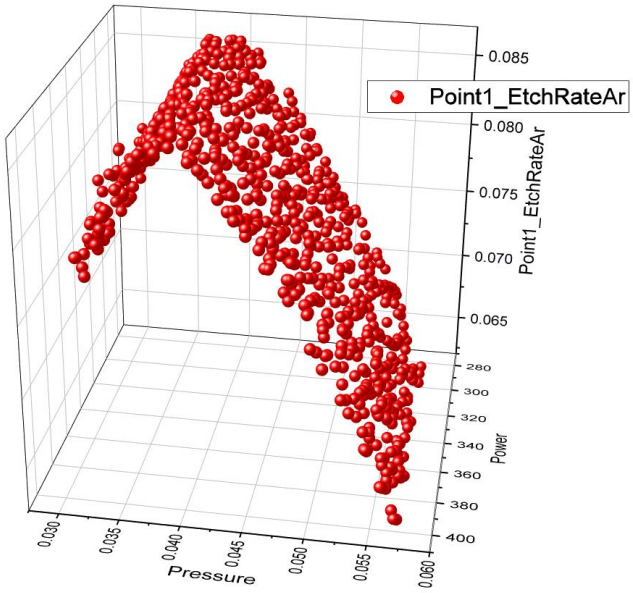

Figure 4: Etching rate (um/min) at the center of the wafer versus the pressure and the coil power for the Ar plasma.

Baseline

In order to have a baseline in the discussion, a simple linear regression can be used in the form of $y = a_1*x_1 + a_2*x_2$, where y will be the vector of the 10 outputs mentioned before and x_1, x_2 the two inputs. For the linear model sklearn linear regression was used. The graphs below show the results of the fitting.

For the Ar dataset the performance is shown in Figure 3 and for the O_2 dataset in Figure 4.

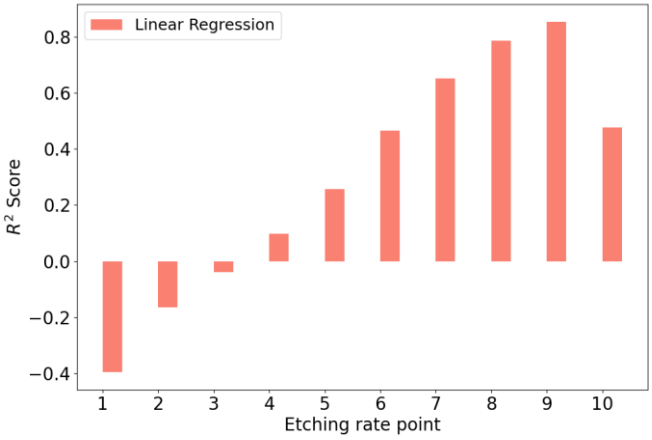

Figure 5: Per output R^2 score of linear regression on the Ar dataset after training. Average **R^2 = 0.299**.

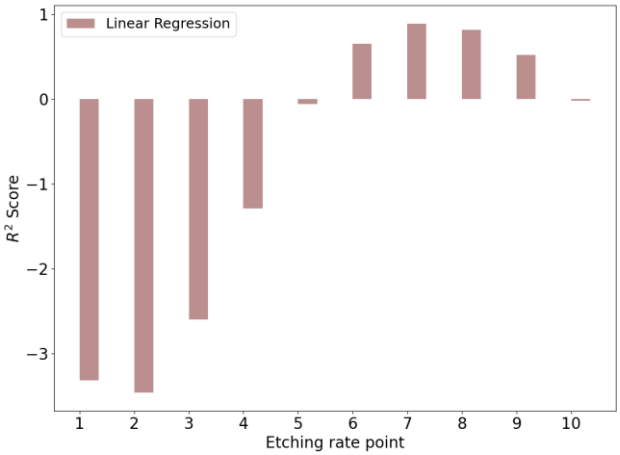

Figure 6: Per output R^2 score of linear regression on the O_2 dataset after training. Average **R^2 = -0.7868**.

Another comparison can also be made with SVR after training it at the train dataset. The results are shown below:

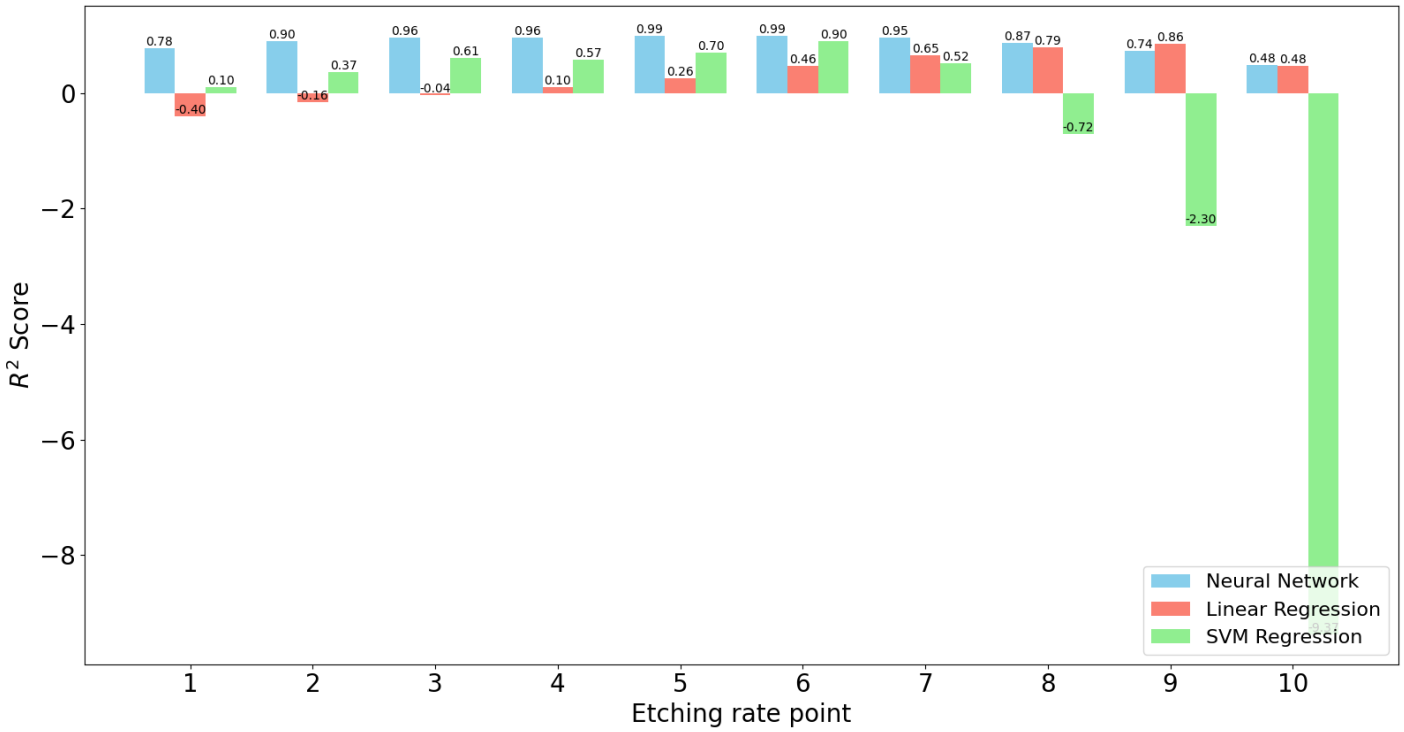

Figure 7: Overall comparison of the models for the Ar dataset. Average SVR R^2 = -0.861.

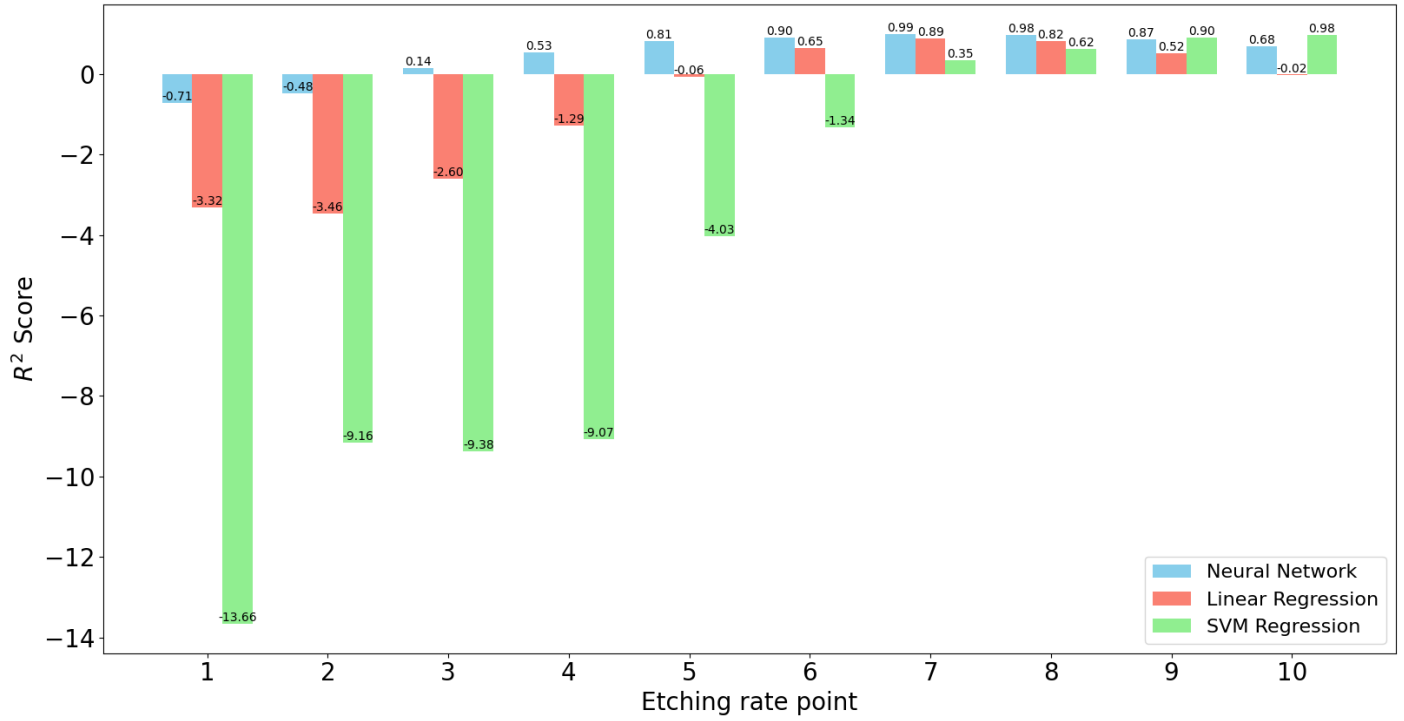

Figure 8: Overall comparison of the models for the O2 dataset. Average SVR R2 = -4.38

Results – Metrics - MLP

The NN's architecture is a simple feedforward model performing regression, with two inputs and ten outputs corresponding to different etching rate values across the wafer as shown in Figure 1. Before training, the optimal hyperparameters of the NN were found using *optuna*. The hyperparameters that were optimized are: learning rate, batch size, weight decay, number of layers and number of neurons/layer. To ensure good generalization, early stopping and L2 regularization technique was applied. The normalization was done with Min-Max with the respective global values. This approach ensures fair normalization for the testing sets and provides unique scaling for both NNs which is essential for transfer learning in regression tasks. In other words, the minimum and the maximum of the whole dataset was used for each of the inputs and the outputs. Regarding the loss function, a weighted mean squared error loss was chosen; the outputs can differ by an order of magnitude (output 1 vs output 10) due to higher gas

species densities occurring at the center of the PMMA substrate (Figure 1)*. This difference in the magnitude made it difficult for the NN to learn all the outputs when they all contributed equally to the loss.

Table 2: Evaluation of the trained models on the respective test sets using Mean Absolute Error (MAE) and the coefficient of determination (R^2). M1 denotes the neural network trained on O_2 plasma data, while M2 denotes the model trained on Ar plasma data.

Metric	M1 (O_2 model)	M2 (Ar model)
Average MAE	2.62×10^{-2}	3.43×10^{-5}
Average R^2	0.4726	0.8629

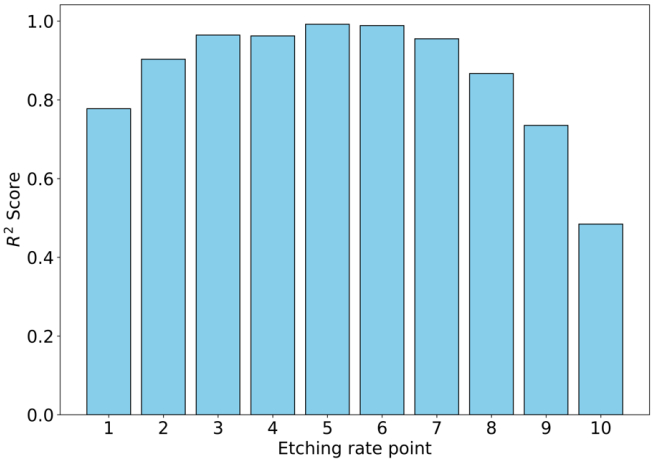

Figure 9: Per output R^2 for the M2 model

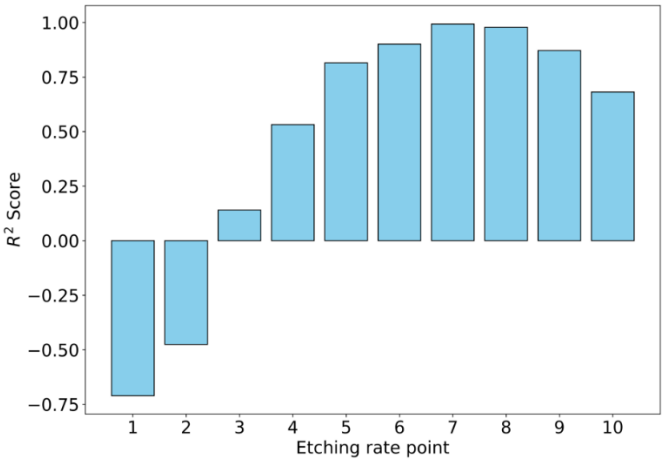

Figure 10: Per output R^2 for the M1 model

Seeing Figures 5, 6 we see that the etching rates of the O_2 plasma are harder to predict than the Ar plasma. Maybe, this is explained by the higher complexity of the O_2 plasma as described in the Domain section. In order to compare the performance of the NN to the baseline we can combine Figures 3, 5 and 4, 6:

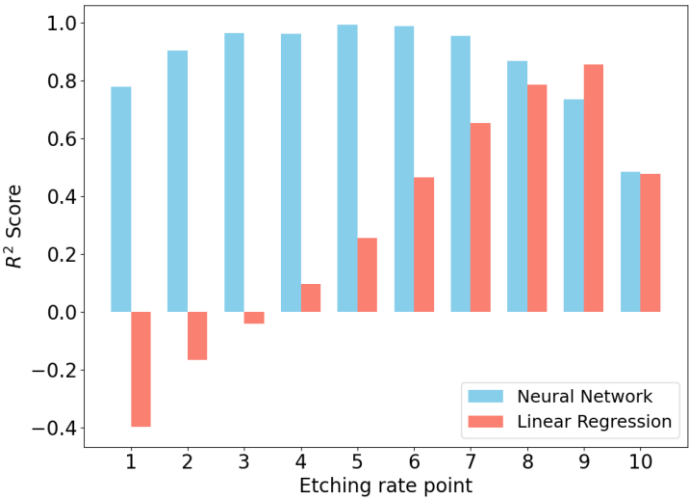

Figure 11: Neural network (M2) vs Linear regression performance of the Ar model.

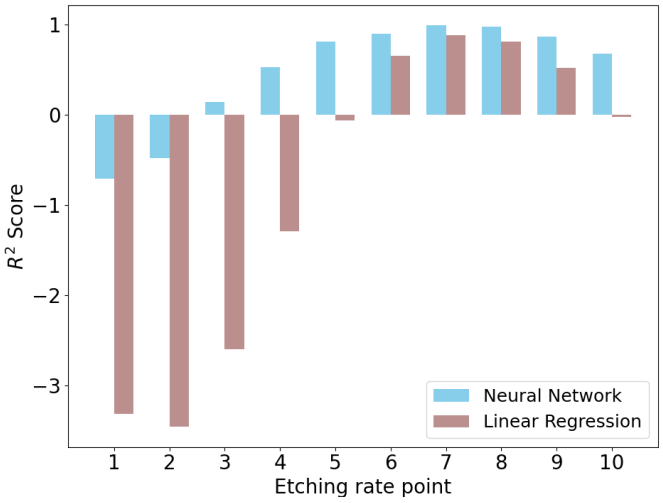

Figure 12: Neural network (M2) vs Linear regression performance of the Ar model.

In both models it can be seen that the NN outperforms the simple linear regression. The overall R^2 score is higher for the NN.

Implement Kolmogorov – Arnold Networks (KANS)

Since KANS are promising alternatives to MLPs, especially on small scale tasks, we tried implementing them in the regression task. One difference from MLPs is that these networks have learnable activation functions and also they do

not have linear weights [4]. They also are used in cases were the data come from some kind of equations. The package that implements KANS comes also from the repository of [4]. For the hyperparameters, the same as the optimal MLP where used (3 layers. Also, the same optimizer (Adam) and the same criterion (MSELoss) where used with the previous MLP. In figure 9 the KAN architecture is shown and in table 3 the metrics.

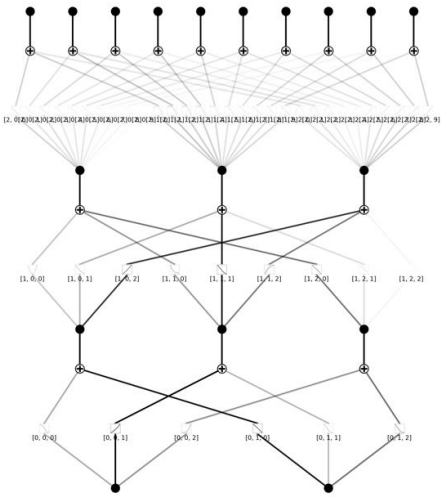

Figure 13: KAN architecture with 3 layers, 2 inputs and 10 outputs.

Metric	M1 (O_2 model)	M2 (Ar model)
Average MAE	0.00418	0.00055
Average R^2	0.77667	0.9844

Table 3: Evaluation of the trained models on the respective test sets using Mean Absolute Error (MAE) and the coefficient of determination (R^2)

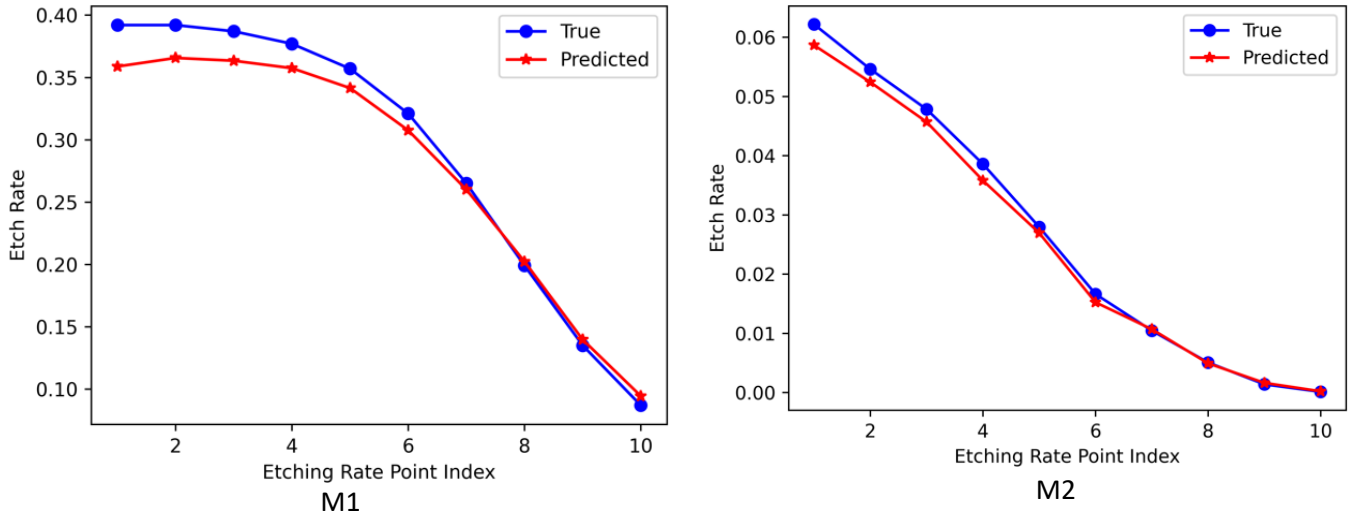

Figure 14: Cases with the worst residual for both models, from the test set.

Transfer Learning

To perform transfer learning, a series of experiments were performed, including fine-tuning the NN from the first dataset to the second and vice versa, explore catastrophic forgetting, freezing layers and finally investigating training with a reduced amount of data. Note that, in all experiments the previous mentioned datasets were used. M1 refers to the O_2 model, trained in the O_2 dataset, M2 to the Ar model trained in the Ar dataset, M12 refers to the pre-trained O_2 model fine-tuned to the Ar dataset and M21 the opposite of this.

First things first, a simple test was conducted on the pre-trained model using the opposite test set, i.e. feeding Ar test data to M1 and vice versa. As expected the results were poor in both scenarios, since each NN knows the dataset that is trained on.

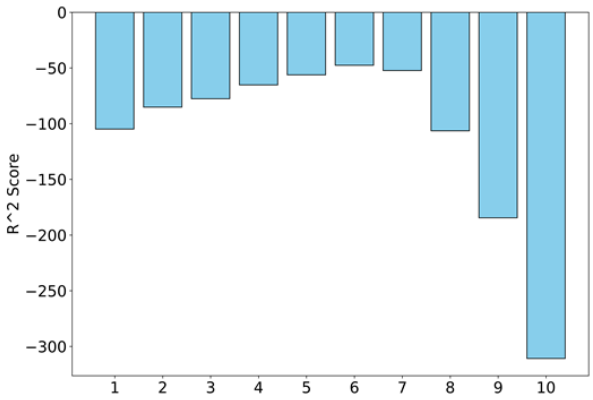

Figure 9: M1 tested on Ar Dataset (D2)

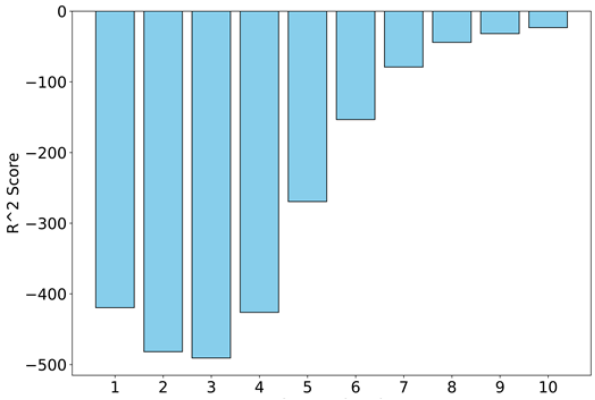

Figure 10: M2 tested on O_2 Dataset (D1)

At this stage, the pre-trained NNs are fine-tuned by training on the opposite dataset. This is done by loading the NN (not initializing randomly this time) and train with a learning rate 20 times smaller. M12, M21 NNs are derived.

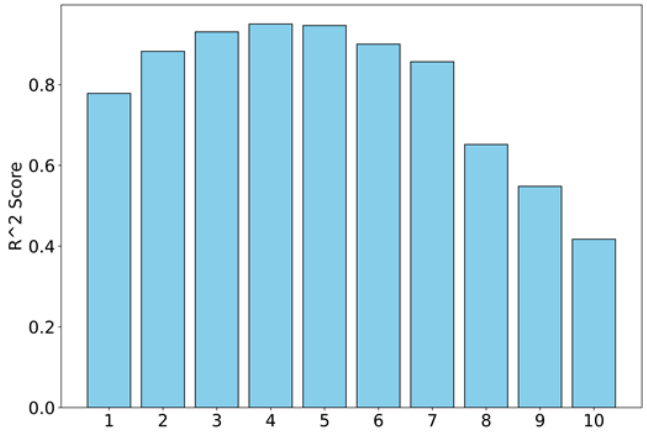

Figure 11: M12 neural network performance on the test set (M1 model on D2)

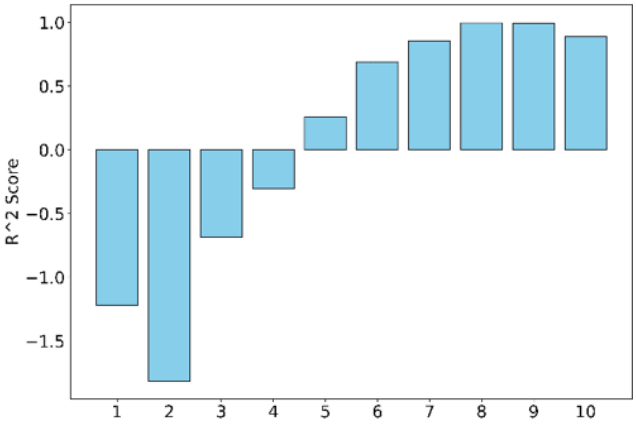

Figure 12: M21 neural network performance on the test set (M2 model on D1)

Table *3*: Evaluation of the fine tuned models on the opposite test sets using Mean Absolute Error (MAE) and the coefficient of determination (R^2). M12 denotes the O_2 plasma neural network trained on the Ar data, while M21 denotes the Ar plasma neural network trained on the O_2 data.

Metric	M12 (O_2 model on the Ar dataset)	M21 (Ar model on the O_2 dataset)
Average MAE	6.63×10^{-3}	3.92×10^{-2}
Average R^2	0.8623	-0.1523

At this point, comparing tables 2 & 3 we observe that fine-tuning performs fairly well for the Ar plasma, as the drop of R^2 is negligible. However, for the O_2 plasma, the results are suboptimal. As illustrated in Figure 12, the first half of the outputs is under-predicted. If we test M12 and M21 on their initial dataset (M12 on D1) and (M21 on D2) we will see that both models forget their initial task (Table 4). However, note that for the domain of interest, this is a desired outcome, as opposed to what the general wisdom in machine learning states. We want the new models to adjust to the new chemistry.

Table *4*: Evaluation of the fine tuned models on the initial test sets using the coefficient of determination (R^2).

Metric	M12 (O_2 model on the O_2 dataset)	M21 (Ar model on the Ar dataset)
Average R^2	-200	-62.9

How to transfer knowledge more efficiently? Re-thinking the present task, an assumption can be made and further validated/canceled; the initial layers of the NN adapt to the operating conditions of each case and its governing chemical reaction mechanism, while the deeper layers account for the solution of the PDEs and the surface mechanism, which are simulated similarly in both PBMs. Series of experiments with freezing of layers are made which

include freezing the first layer, the last or the middle one. Also an experiment including gradually unfreezing a layer after 100 epochs was conducted. For the sake of time, experiments of unfreezing after a different number of epochs were not conducted. Also, freezing is done by setting the parameter requires.grad to False in the code. With this, gradients are not computed and the weights are not updated – so the neurons do not learn/adjust. Table 5 has the freezing results in a nutshell.

	Freeze First Layer		**Freeze Last Layer**		**Freeze Middle & Last layer**	**Gradually Unfreeze Last Layer**	
Metric	M12	M21	M12	M21	M21	M12	M21
Average MAE	5.89×10^{-3}	4.65×10^{-2}	**5.04×10^{-3}**	**1.66×10^{-2}**	5.00×10^{-2}	5.04×10^{-3}	3.21×10^{-2}
Average R^2	0.8923	-0.9078	**0.9226**	**0.6463**	-0.0028	0.9185	0.5745

These experiments illustrate that inverse transfer learning by freezing the last layer produced the best results out of all experiments and outperformed fine tuning. Regarding the Ar plasma data, this transfer learning technique yielded results comparable to the initial M2 model trained on D2 (Ar data), while for the O_2 plasma data, this method outperformed the initial model. As seen in Figures 14, 12 the frozen model made errors similar to the initial one but to a much lesser degree. This is a promising result, as it demonstrates that tasks in our domain of application, which can be successfully modeled using NNs, can serve as a starting point for training models on different chemistries where the initial training is not that satisfactory. Also, note that gradually unfreezing the last layer did not yield better results. This further validates the initial assumption that the last layer holds universal knowledge shared by the two chemistries.

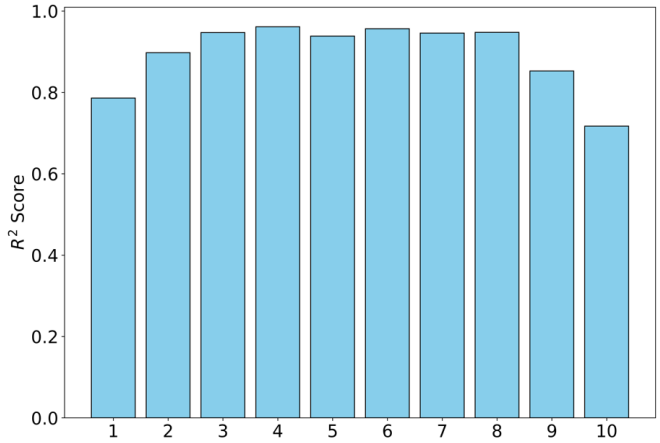

Figure 13: Per output R^2 for the M12 (frozen the last layer)

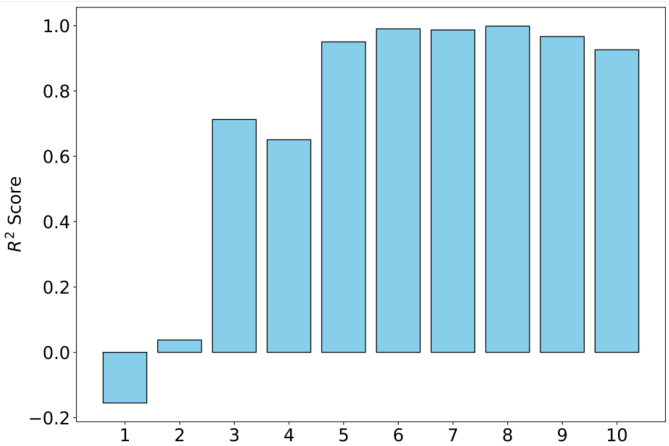

Figure 14: Per output R^2 for the M21 (frozen last layer)

The last set of experiments include cutting down the amount of data. Since I have one trained neural network on one chemistry (O_2 plasma) is it possible to fine tune it to the Ar plasma using less data and taking into account the previous freezing technique? This would be very helpful because it will reduce the amount of data from the next simulation with different chemistry. The experiments included using 80, 50, 20% of the second chemistry data.

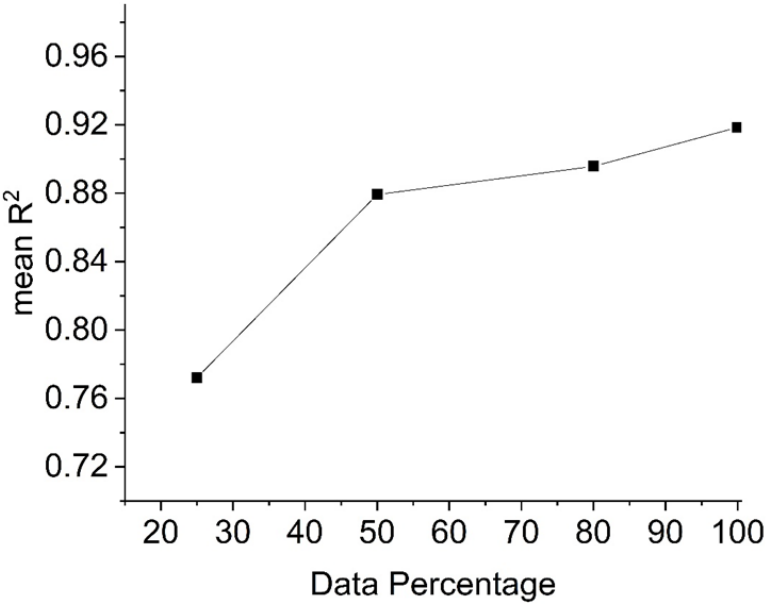

Figure 15: Mean R^2 of the Neural Network when fine tuned to the second chemistry.

From Figure 13 it can be seen that the data required to fine-tune the pre-trained neural network on the O_2 dataset to the Ar dataset can be reduced by 50% without significantly compromising accuracy and even with less trainable parameters (since we freeze a layer).

Challenges

The scope of this work was to build a pipeline that includes the development of the PBM, the training procedure of the neural network model and the knowledge transfer from the pre-trained model on one chemistry to another. The neural network is going to be used in optimization algorithms to accelerate the computations. Some key challenges are noted below:

- Simulation development; in order to have an accurate model, exhaustive literature search was required to gather all the chemical reactions in the gas phase of the reactor.
- Data generation; the dataset may seem rather small, but taking into account the limited computational resources available and the computational cost of the simulation, having a larger dataset required more time.
- Neural Network architecture and hyperparameters optimization; as mentioned, due to limited time and computational resources, the grid search for finding the hyperparameters was limited. Having a larger neural network may lead to better results.
- Exhaustive transfer learning experiments; in the beginning, transfer learning gave poor results. So, I had to conduct multiple experiments to find a technique that would work. The procedure included freezing layers, trying to unfreeze after a number of epochs, freezing a combination of layers and in the end some experiments of cutting down data. Reducing the data required from the simulation by using pre-trained models is the ultimate goal. The transfer learning was conducted using the MLP, because KANs required almost 6-7 times more training time. However, KAN had better results than the MLP where, especially in M1 model, it showed a low overall R^2.
- I could use PINNs, but in the simulation there are solved multiple PDEs and implementing them in the loss function is not straightforward. PINNs may be useful in the future when we will use AI to predict the spatial distributions with coordinates in the reactor.

References

[1] D. B. Graves, C. B. Labelle, M. J. Kushner, E. S. Aydil, V. M. Donnelly, J. P. Chang, P. Mayer, L. Overzet, S. Shannon, S. Rauf, and D. N. Ruzic. Science challenges and research opportunities for plasma applications in microelectronics. Journal of Vacuum Science & Technology B, 42: 042202, 2024. doi: 10.1116/6.0003531.

[2] S. Mouchtouris and G. Kokkoris. A hybrid model for low pressure inductively coupled plasmas combining a fluid model for electrons with a plasma-potential-dependent energy distribution and a fluid-monte carlo model for ions. Plasma Sources Science and Technology, 25:025007,2016. doi: 10.1088/0963-0252/25/2/025007.

[3] D. A. Toneli, R. S. Pessoa, M. Roberto, and J. T. Gudmundsson. On the formation and annihilation of the singlet molecular metastables in an oxygen discharge. Journal of Physics D: Applied Physics, 48(32):325202, 2015. doi: 10.1088/0022-3727/48/32/325202.

[4] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Y. Hou, Max Tegmark KAN: Kolmogorov–Arnold Networks [https://openreview.net/forum?id=Ozo7qJ5vZi]

[5] Kondyurin, A.; Bilek, M. Etching and structure changes in PMMA coating under argon plasma immersion ion implantation. Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms 2011, 269 (12), 1361-1369. DOI: 10.1016/j.nimb.2011.04.001.