Deep Learning Regression of VLSI Plasma Etch Metrology

An Honors Project Manuscript

Presented by:

Jack Kenney

College of Information and Computer Sciences

University of Massachusetts at Amherst

Completion Date:
April 2019

Approved by:

Dr. Edward Rietman

College of Information and Computer Sciences

Dr. Scott Riggs

Lam Research Corporation, Fremont, CA

Abstract

Title: Deep Learning Regression of VLSI Plasma Etch Metrology

Author: Jack Kenney, CICS

In computer chip manufacturing, the study of etch patterns on silicon wafers, or metrology, occurs on the nano-scale and is therefore subject to large variation from small, yet significant, perturbations in the manufacture environment. There is an enormous amount of information that can be gathered from a single process, a sequence of actions taken to produce an etched wafer from a blank piece of silicon. Each final wafer, however, is costly to take final measurements from, which is a limiting factor on the number of examples available to train a predictive model. Part of the significance of this work is the success we saw from the models despite the limited number of examples. In order to accommodate the high dimensional process signatures, we isolated important sensor variables and applied domain-specific summarization on the data using multiple feature engineering techniques. We used a neural network architecture consisting of the summarized inputs, a single hidden layer of 4032 units, and an output layer of one unit. Two different models were learned, corresponding to the two metrology measurements in the dataset, Recess and Remaining Mask, as they are related in an abstract sense and do not belong together in a two dimensional feature space. Our results approach the error tolerance of the microscopic imaging system. The model can make predictions for a general class of recipes that include the correct number of etch steps and plasma reactors with the appropriate sensors, which are chambers containing an ionized gas that determine the manufacture environment. Notably, this method is not restricted to some maximum process length due to the summarization techniques used. This allows the method to be adapted to many other processes, given the data. In order to automate semiconductor manufacturing, models like these will be needed throughout the process to evaluate production quality.

1 Introduction

The manufacture of computer chips uses a plasma reactor that contains an ionized gas to conduct a series of chemical operations on the surface of a silicon wafer. Prior to being placed in the reactor, a mask is applied to the wafer in the shape of a desired circuitry pattern. Inside the reactor, various substances are deposited on the surface of the wafer and activated away from the areas of the wafer that do not have the mask applied. The sequence of depositions and activations is defined in a recipe and referred to henceforth as a process. The outcome of a process is determined by measuring the etch patterns on the silicon wafers, which is called metrology. To ensure that processes have been successful without performing costly tests, such as taking a cross-sectional image of a finished chip, we would like to be able to predict the metrology measurements using automatic tools.

The dataset was comprised of inputs of environmental sensor information from a plasma reactor with a wafer inside during a Very Large Scale Integration (VLSI) process and outputs from the corresponding measurements derived via metrology on images of the etched wafer. The goal of this project was to apply machine learning (ML) regression models to map the sensor data as inputs to the metrology as outputs.

To address this regression problem, we first modified the data using a feature engineering pipeline. Data expansion techniques appropriate given our feature engineering for simulating a larger dataset were considered, but were not found to provide significant improvement. We then trained a neural network with the technique of grouped leave-one-out cross-validation (LOOCV) on the engineered dataset to determine the hyperparameters of the models that are best suited to the data and tested the final hyperparameters on a held-out test set to estimate the generalization error.

Overall, our goal was to make accurate predictions of process outcomes with the provided data, using various statistical and ML methods. A critical feature of this work was that, given the small dataset size and limitations, the results and methodology were as statistically

sound as possible.

The grant contract stipulates that a publication must be sought for the project, meaning that in addition to this manuscript and the software deliverables for the project, a paper will be produced and submitted to the *IEEE Transactions on Semiconductor Manufacturing*, the primary journal on the subject.

2 Significance

Plasma etching is ubiquitous in semiconductor manufacturing because the plasma, an ionized etching gas, can be directed in non-ballistic trajectories directly into the material substrate to be etched by the chemically active ions. Similarly, plasma can used for deposition. The two processes can be combined for complex submicron and nanoscale structure fabrication, often with scores of processing steps. Every etch process has specific chemistry and instrument setpoints including radio frequency (RF) power(s) for creating and maintaining the plasma, direct current bias on the wafer surface, mass flow control of various gasses, pressures, and many more factors. A statistical design of experiments on a set of wafers is used to identify the required conditions for an etch process, followed by fine-tuning the control parameters using another set of test wafers before the process can be used in manufacturing. Plasma etching is a very complicated and costly process, that can be optimized using various machine learning methods, one of which is the goal of this project.

A skilled engineer will be able to guess the approximate end results, that is, the etched structures and features on the wafer. A first step in applying machine learning to streamline the wafer research and design for manufacturing process is to predict the final etch results as would be observed on the wafer surface by metrology tools. To this end, there have been many papers in the literature describing both the end result of an etch and virtual metrology modeling for plasma etch operations. Early examples include [4, 6–8, 11, 12]. More recently are [5, 10, 17].

Given the vast array of chemical processes for both etch stack and gases used in deposition and etching, it is not reasonable to expect any given single neural network, or other instance of a machine learning algorithm to generalize beyond the chemistry it was trained on. Nonetheless, as cited in the literature above, these specific instances of plasma etch modeling and/or metrology prediction can be used for real-time control of manufacturing processes and for failure prediction, as well as automated process analysis.

Predicting end-of-process results from time-series data is a difficult task in modern machine learning, especially when data is collected at various fidelity. The problem is particularly difficult when processes have varying lengths, such as is the case with the raw data with which this project began. Making appropriate feature engineering choices to accurately summarize and consolidate sensor data takes domain knowledge, planning, and careful execution. In the commercial field of VLSI silicon processing, the cost of performing experiments is high and interferes with manufacturing efficiency. Thus, being able to predict results from *in situ* process sensors is highly valuable.

Due to these costs, the number of examples for the model to learn from is low, which makes the problem even more difficult, pushing the boundaries of modern machine learning and engineering. Being able to predict the critical dimension (CD) measurements of the etched silicon with a low enough error, using only process sensors, we take a step forward in automating the silicon etch process, saving time, money, and resources in the industry that ubiquitously underlies modern computing technologies.

The research question we are addressing pushes the limits of machine learning under extreme conditions. In this case, we have very few, 14 in total, examples from the distribution of etched wafers in the recipe class we are addressing. In the raw data, there are millions of features associated with each wafer, and up to eight metrics to regress. With so little data, the process by which we feature engineer and examine model efficacy is crucial to the project's success. Notably, the proper execution of this task will result not in an ideal system model but instead serve as a proof-of-concept in plasma etching complex micro-structures

with multilevel-materials.

3 Background

This project requires a great deal of domain knowledge about the VLSI silicon manufacturing industry. The main work referenced for the background knowledge of this process is [16]. The type of etch process that the provided data corresponds to is a "dry etch," which is covered in Chapter 16 of the textbook. This chapter discusses the basic chemistry that is still used in modern production, the monitoring process, and some of the sensors involved in tracking an etch process. Chapter 12 focuses on lithography, the chemistry behind photoresists, and applying a desired chemical mask to a silicon wafer. These two chapters provided a sufficient introductory knowledge to understand the interactions underlying the data, and the sensor labels present in the raw data. Next, we will discuss some specifics from the text that are important for understanding plasma etching.

Silicon etching is a process by which an etch-resistant mask is applied in a certain pattern, generally corresponding to a circuit diagram, to the surface of a silicon wafer. "The overall goal of an etch process for VLSI fabrication is to be able to reproduce the features on the mask with fidelity." A mask is used in photolithography to apply a photoresist to the silicon wafer according to the desired pattern the etch is intended to produce. Initially, a resist is applied, in our case a gas that reacts with the surface of the wafer in a deposition form, followed by light radiation applied through the mask that blocks the radiation. In a positive resist, the radiation renders the resist soluble, meaning the light-affected parts are now able to be etched or washed away, and conversely, a negative resist hardens the light-affected parts and anything not touched by the radiation can be etched away [16].

Once the mask has been applied, a gaseous plasma is struck above the wafer inside a reactor, and through a variety of deposition and activation steps, the non-resistant portions of the silicon wafer are etched away. Typically, a box of 20-30 wafers is run in sequence,

so that the chamber conditions are as consistent as possible. Ideally, there is consistency within a certain reactor not only across the wafers being run, but from batch to batch [16]. These are some of the factors that make prediction difficult, and it is worth noting that the dataset spans two batches.

My understanding of the plasma etch process was refined by a visit in June '18 to Lam Research Corporation, where I was given a tour of the plasma etch machines, given an overview of the sensors available in the data, the fidelity of each and the way each tool functioned. On the experimental machine from which the data were collected, there are two separate computers, a gas manifold for regulating flow and chemistry, a radio frequency generator that controls the plasma, a "chuck" that holds the wafer during the etching, and a variety of environmental sensors inside the chamber. These sensors include temperature, pressure, optical emissions spectra (OES), and many more. There is a time delay between step boundaries and when the new gasses enter the chamber, so the steps can actually begin several milliseconds after the step-boundary as indicated by the recipe, which motivated a feature engineering technique that minimizes the impact of this delay. With a baseline knowledge of the physics and chemistry involved in the etch process, we move to reviewing relevant literature involving plasma etch modeling using neural networks and other methods below.

The first reviewed paper [12] came from the *IEEE Transactions on Semiconductor Manufacturing*. In section III, this paper discusses the different kinds of prediction problems that can be framed around the plasma etch process. In particular, predictions can be made using reactor properties about the physical and chemical effects on the wafer, etch rate for example, and the work itself discusses a method of taking reactor input parameters and sensor readings and predicting wafer attributes such as line width and mask oxide remaining. Additionally, the neural network architecture is discussed, and uses tanh activation, has two hidden layers, and twenty-three input nodes. The process that generated the data for our project has many more steps in length than what was state-of-the-art in 1996, and we expect

to have many more input nodes mapping to a similar one to four output nodes.

A work referenced by the previous work is, [11]. Here, the authors address the same type of problem as in the first paper. One important discussion from this paper is in section III, where two works, Mocella and Himmel, referenced used particularly small datasets, 34 and 32 examples, respectively, and still produced comparatively strong results. The comparison made the case for neural networks over other statistical methods, however, this is still important for our work because we have a similarly small dataset.

Another paper relevant to the project is [13]. This work used summary techniques to reduce the number of input features to the model in a meaningful way. There is a lengthy discussion regarding the difficulty that neural networks have in working with time-series data, motivating a method of summarization of the time-series data to a digestible size for the neural network to process as a single vector. The problem they work with is not as relevant to our dataset, as it is about failure detection, not measurement regression, but the methods discussed for preprocessing are very relevant to the project.

The work [14] proposes a method of dealing with a variable length input to a neural network using a gating method that would allow for a maximum number of inputs and deal with variability up to that threshold. We considered applying this technique to our model because we also had a variable length input for each example, however, despite there being a known max-length for this dataset, there is no guarantee that future inputs the model is expected to generalize to, will conform to this standard. In fact, as time has progressed and silicon manufacturing has advanced, it appears the trend is to have longer and more involved processes, which makes this approach not ideal for our problem. The paper also provides a detailed statistical analysis and account of the data cleaning and feature engineering that was used. There was a large amount of statistical analysis done to determine which features should be selected to include in the model. Overall, this paper provided insight and was a significant resource for data cleaning and feature engineering in our project.

With a small dataset, it is important to consider different methods for data expansion,

to take the existing data as a sample from the true distribution of like-data and create more data samples from those that are available. One paper, [15], provided a theoretical framework for the benefit of partially destroying the input in a variety of ways, to create a more robust model that cannot rely solely on the value of a single feature, and must distribute its learning across the full feature space. More data expansion techniques are discussed in the works below.

An important work for this project is [1], because it provides a foundation for dealing with a dataset. Not only does it cover handling data using pandas, a Python library, but it covers feature analysis using statistical methods, different data visualizations, preprocessing data, creating formal data pipelines, fine-tuning models, and presenting results. This work covers these methods and more, including regularization techniques, using TensorFlow – another Python library for creating neural networks. The book also covers other models that will be useful in making comparisons against the neural network for efficacy, such as support vector machines.

Another important work for the coding involved in this project is [9]. This work discusses in detail the pitfalls of machine learning that occur in industry, such as pipeline jungles, cyclical data dependencies, and non-versioned data streams. The author provides a framework for creating machine learning software that is written to conform to tests and uses tests to ensure data dependencies are maintained. Writing object oriented machine learning code provides robustness and clarity, and supports incorporating models into existing code-bases safely [9].

Lastly, [2], provides a strong theoretical framework for machine learning using neural networks that influenced the choices of architecture, regularization, hyperparameter optimization, and other points surrounding the theory and methodology of this project. It has introduced new methods and clarified many of the methods discussed in the works described above. Specifically, it describes methods for training networks that were used in the project, such as early stopping, data normalization, stratified and grouped cross validation, and simu-

lated annealing. It also details other methods for effective data augmentation, such as adding random noise to data and statistical sampling from an extrapolated data distribution, both of which make a model more capable of generalization from a small dataset.

4 Methodology

The steps are initially described by time-series data for various sensor types. To compress the data from its raw form, we decomposed it into fitted parameters by cycle and fitted those parameters by type of cycle, or step, such that they were of a reasonable dimension to be used as input for machine learning models. Finally, dataset of inputs-to-outputs was constructed, aligning the input sensor data appropriately with the measurements taken from the corresponding wafer images as output. The images were taken using a Scanning Electron Microscope and then measured by an engineer. The dataset was then used in a cross validation procedure to identify the best model for the problem and estimate generalization error.

4.1 Data Background

4.1.1 Process Overview

The dataset was derived from a plasma etch process outlined by a specific wafer structure, procedure, and sequence of gaseous chemistry. The wafer material stack is displayed in Figure 1. We used a surrogate structure as a test vehicle for a non-self limiting atomic layer epitaxy (ALE) application. The ALE was accomplished by depositing a passivation layer onto the wafer surface using chemistry A. The purpose of A was to generate molecules that will attach to the film(s) of interest. Chemistry A with energy E_A was followed by activating the now-passivated surface with chemistry B at energy E_B, where chemistry B's primary purpose was to create ions to facilitate reactive ion etch (RIE). The surrogate structure we

used was a line and space grating mask. The film of interest we were etching was an oxide, and film we did not want to etch will be referred to as non-etch-material (NEM). The A was a fluorine compound, and the chemistry of B contained Ar. The chemical depositions and activations were divided into a complex sequence of several repeated steps which allowed for deep nanostructure fabrication. Most of the steps were deposition of materials and activation etching steps, but there were others for chamber environmental stabilization and initial striking of the plasma, which were agnostic to process on-wafer.

After the plasma etch process was completed, a die, a small area similar to an integrated circuit, was cut from the same place on each wafer, notably in the center to reduce edge effects. The die was then cut cross-sectionally in the same place for image analysis, and important critical features were measured and recorded by a domain expert process engineer. These measurements served as output for the regression machine.

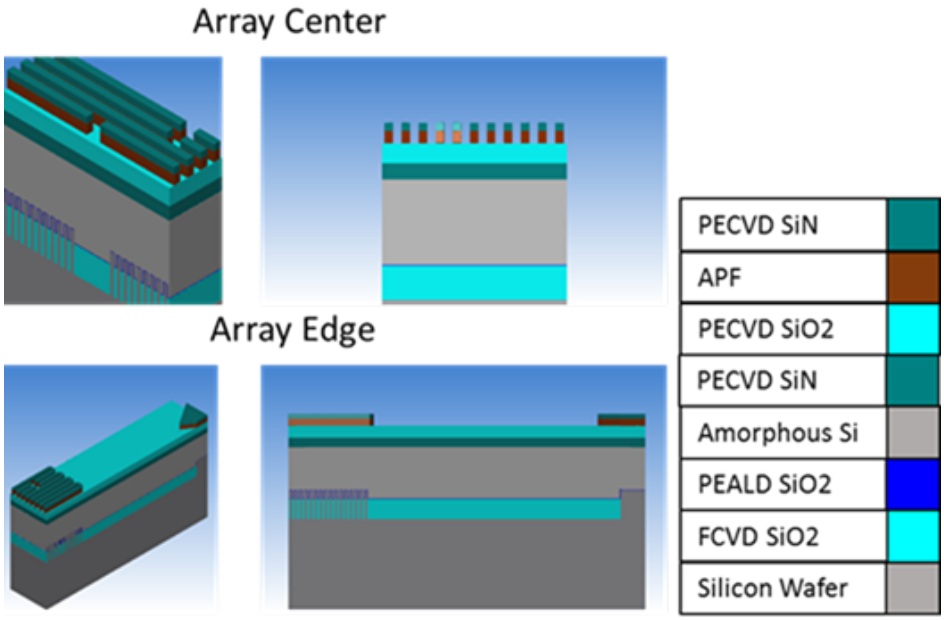

Figure 1: The Etch Stack: An example of the wafer construction used in the etch process, including the chemistry associated with each layer of interest.

4.1.2 Dataset

The dataset consists of fourteen whole wafer runs. Each wafer has a collection of sensor measurements and a set of corresponding metrology measurements. The layered chemical composition of the wafers corresponds to Figure 1. Wafers R17, R19, R20, R21, R22, R23, R24, and R25 nominally came from the same chamber condition, meaning they were processed consecutively with minimal RF hours between wafers. Because they are similar, we will name them collectively Group 1. Running the wafers in close sequence is important because the chamber undergoes many changes during the etch process and the conditions change as etching occurs. These wafers being run in sequence means that the difference in their signals is more likely due to the variation that is occurring in the etch itself rather than the changes in the chamber that have occurred during unrelated etches. These represent the bulk of the data and are considered relatively normal.

R28, R29, R30, R31, on the other hand, were run after the chamber had a full wet clean where various chemicals were run through the interior of the reactor to remove buildup from previous etching processes. Consumable parts were also replaced. We will refer to that set of wafer collectively as Group 2. R32, R33, and R34, were run after those changes and were from a new lot of wafers from the supplier. The new box of wafers from the supplier had slightly smaller stack structure. As the initial wafer geometry changed, the final output geometry is expected to be different as well, so we will refer to those as Group 3. Mapping the raw recipe setpoints to the output measurements alone, without counting for pre-etch geometry, could possibly introduce error into the model, but due to the limited dataset and the variety that the final model would be expected to generalize to, all wafers were used. These were also run after Group 2, meaning the chamber condition was not the same a for Groups 1.

By taking sequential samples before and after a wet clean, we aimed to build a predictive model more robust to changes in chamber condition over time as well as with varied parts. As such, we conclude that though the data was not truly independently and identically

sampled, the variation of the samples was considered to span much of the distribution that future wafers, those to which the model will be expected to generalize, are going to be drawn from. To test this hypothesis, we selected wafers R22 from Group 1 and R34 from Group 2 for testing, as they represent very different sections of the dataset: the former was early in the process batch, and the latter came after the wet clean and from a different lot of wafers. The results from testing on these wafers is discussed in Section 5. In this way, the experiment is reasonably directed toward application with models that carry implicit assumptions of independence among training and testing samples.

The recipe produces data that occurs in multiple steps, the most important of which are the etch steps. We focus on those steps for the problem, because the deposition and activation steps are the most relevant to the etching process and the other steps were not considered important. The sensor data is compressed by selecting a subset of the sensor variables, calculating linear fit coefficients across each cycle, and then fitting a third-order polynomial to those calculated coefficients across the *cycles* of the each distinct step type. We selected only deposition steps and activation steps, due to variety in the recipes in the data set around one of the activation steps, and ignored non-etch steps.

4.2 Data Preprocessing

To take the raw data files from the reactor and produce meaningful samples for a neural network to consume, the order of the data needs to be dramatically reduced. There are a set of wafers $W = \{w_1 \cdots w_r\}$, where $r = 14$. Originally, the data files are on the order of 1,000 columns and 50,000 rows. The columns were represented by a set $V = \{v_1 \cdots v_c\}$, where c was the number of columns, we selected some c' variables to use in creating an example. To further reduce the quantity of features per example, we defined a set $S = \{s_1 \cdots s_p\}$, where p was the number of critical steps in the recipe. For this etching process, the engineers determined that focusing only the activation and deposition steps is sufficient to learn meaningful relationships. Each step was repeated q times, which defines a set of

cycles $C = \{c_{11} \cdots c_{pq}\}$that we used to compress the rows of the example. To do this, we considered each cycle c_{ij} separately. $\forall i : \forall j$: we trim the starting values to a critical window and collect $A = \{m, b, f\}$, which are the slope and intercept of the linear fit and the approximated asymptote of the cycle, respectively, these will be referred to collectively as the "intracycle augmented linear fit coefficients." Henceforth, we will indicate the A_kth fit coefficient for cycle c_{ij} using a superscript, e.g. c_{ij}^k as seen in Equation 1, and in in the presence of only a step s_i as some coefficient x_{ik} as seen in Equation 2.

$$\forall i \in [0, p) : \forall j \in [0, q) : \quad L(c_{ij}) = c_{ij}^b + c_{ij}^m(t + o), \quad c_{ij}^f = \frac{1}{N} \sum_{i=(1-l)N}^{N} c_{ij}(t + o) \tag{1}$$

where t is time and o is the start of the cycle including trimming from the beginning of the time series, N is the length of the cycle, and l is the percent of the cycle to consider representative of the final magnitude of the cycle, which was in most cases 10%.

Considering one distinct augmented fit coefficient A_k at a time, we fit the cycles for a distinct step, gathering coefficients referred to in Figure 2 as the "stepwise intercycle polynomial fit," as follows:

$$\forall s_i \in S : \forall j \in [0, q) : P(c_{ij}^k) = a_{im} + b_{ik}j + c_{ik}j^2 + d_{ik}j^3 \tag{2}$$

Finally, with these $c' \times |S| \times |A| \times |P|$ coefficients and using our domain knowledge of the etch process our data is sufficiently summarized and is ready for input into the neural network.

Unmentioned above in Equation 2, we additionally added a weighting scheme to the first few cycles $c_i j, \forall j \in [0, 2]$ of the data in a 10%, 20%, and 70% ratio because those cycles were determined in preliminary analysis to be unrepresentative of, and less influential than, rest of the cycles later during the process. It combines the three data into a single data point for each function of the cycles coefficients. This was due to the reactor being still in its initial

phases of startup during those cycles.

For output metrics, cross-sectional images were taken of each wafer and metrology measurements calculated by Lam engineers on each of the five to six images of the die taken. Each image was measured on four metrics. Critical Dimension Trench is the width of the part of the surface that is being etched. Critical Dimension Trench Mask is the width of the part of the surface that has a mask. Recess the depth of the trench that was etched. And lastly, Remaining Mask is the amount of residual mask that is still present on the surface of the wafers. The two measurements we focused on in the prediction problem were Recess and Remaining Mask, as the other two are less deterministically related to the sensor data collected and more related to the geometry of the feature mask applied to the surface of each wafer.

4.3 Neural Network

To determine the appropriate neural network architecture for the features we engineered, we performed cross-validated search over various hyperparameters. Specifically, we used LOOCV on the wafers, meaning that it was twelve-fold cross-validation over the training set. The loss function used for evaluating the efficacy of a model was negative mean squared error (NMSE), which we use because we are regressing real valued numbers and is provided standard in the libraries used for the project. In the next section, we discuss the cross-validation procedure, followed by the hyperparameter search for the neural network. In Figure 2, the "effective model" is described, which is the full data pipeline represented as a deep neural network where the first three layers are considered frozen, static transformations of the input data, and the last three layers correspond to learned weights, as determined by the details in the next two sections.

4.3.1 Cross-Validation Procedure

To ensure that the experiment is statistically sound, the cross-validation procedure was carefully constructed to preserve the validation sets as strictly excluded from their corresponding training sets. When the dies were cross-sectioned, five to six images were taken and output measurements were taken for each of the images. This means that for a given input vector, there are five to six corresponding output examples, which are related to one another. To avoid a situation of testing on the training set, each wafer was given a label, and all of the five to six examples corresponding to that wafer were used as either training or validation data, but not both. Only one wafer, including all of its sub-examples, was used for validation at a time, thus LOOCV was used, despite having multiple samples per validation set. This strict partition of data ensured the statistical viability of the cross-validation procedure for this experiment.

When batch normalization [2] was used to normalize the training and validation sets, specifically using the *training set* as the baseline, not the validation set.

$$\forall v_i \in V : (v_i - \mu_T)/\sigma_T \tag{3}$$

In Equation 3 above, T is the set of examples in the training set and V is the set of examples in the validation set. This is statistically sound, because there is no implicit information about the mean and variance of the validation data provided in the normalization, and as mentioned before, the validation wafers are considered to have been drawn from the same distribution as the training set. Thus, using the mean and variance of the training set as an estimate of the mean and variance of the distribution is sound, within some level of confidence.

When data expansion techniques [1, 2] were used, the training data consisted of the expanded data and the validation data consisted of the non-expanded original data, because testing on expanded data is not what the model will be expected to do in the future. The

expanded data were generated by sampling from a Gaussian distribution that used the covariance matrices of the fit parameters in Equations 1 and 2, where the mean and variance were each calculated using a hyperparameter that specified the sampling percentage of the standard deviation.

Unfortunately, neither the batch normalization nor the data expansion techniques proved fruitful in improving model generalization error. The full table of cross-validation training errors, validation errors, and their corresponding measurement errors from the supervised learning process is available in Appendix B.

4.3.2 Hyperparameter Search

In the search for the model that would generalize best on future examples, we considered the number of layers involved, number of hidden units in those layers, activation function, amount of L2 regularization to apply to the weights [1], learning rate, decay of the learning rate, loss function, number of epochs, and early stopping [2] sensitivity in terms of minimum threshold and patience. The full number of hyperparameter combinations attempted was not exhaustive, as the process would be too resource intense. Due to the limited size of the dataset, however, we were able to be more exhaustive in our search of the hyperparameter space than is typical for deep learning projects. Usually deep learning projects utilize massive amounts of data to learn the weights of many layers and a larger feature input space, but due to the frozen transformations of the input data we are only learning a relatively small single hidden layer automatically and thus many hyperparameter settings were able to be tested. The full list of parameters tested can be found in Appendix A.

The architecture that best approximates the solution to this regression problem had an input layer of size 252, one hidden layer of size 4032, and a single number regressed as output. The hyperbolic tangent activation function was used to add non-linear capabilities to the model. The L2 regularization constant was 100, which helped prevent the model from overfitting to the training data. The learning algorithm used was stochastic gradient descent

with a learning rate of 1e-5 and a simulated annealing decay value of 1e-8, which finds a set of weights that minimizes the prediction error. It used a batch size of 32 to train the model over 100 epochs. On certain examples, early stopping, which helps prevent overfitting by saving and using a model trained up to an epoch before the 100th, if it is better than the 100th. The minimum validation error delta for early stopping was set to 0 with a patience of 10 epochs. This allows for noise not to cause early stopping to terminate training too early. To see the learning curves for each fold in the final two versions of the neural network, see Appendix C.

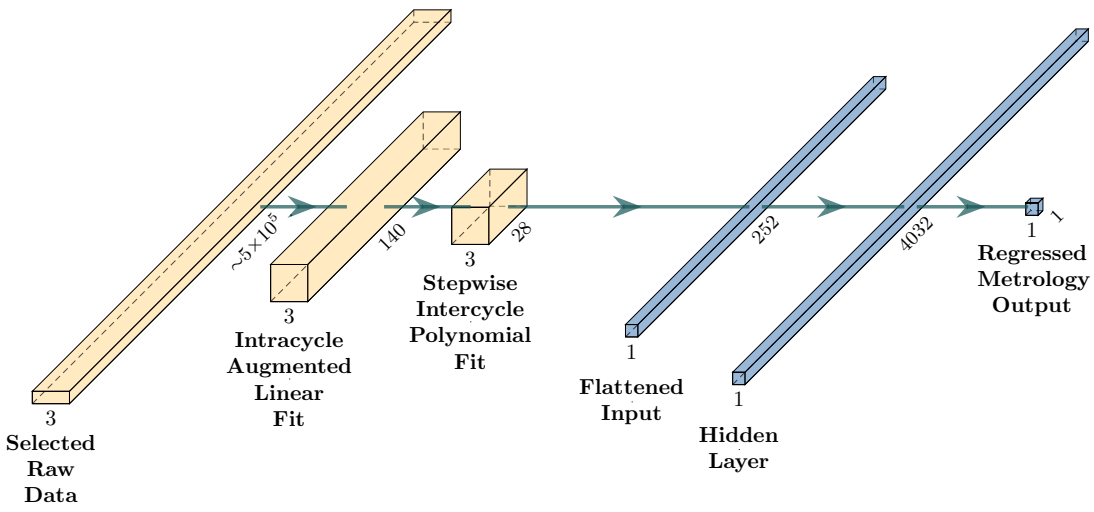

Figure 2: The Effective Model: In this diagram, we see the effective regression model. The transitions between the first three layers are static transformations from the raw reactor data. First, the input data is converted into two linear fit coefficients over each process cycle. We also augment these coefficients by including the final magnitude of each, which carries information about the aysymptote of the cycle. See Equation 1. This generalizes the model to incorporate time-series data of any length, when it is divided into cycles, a common procedure in etch processes. The second set of transformations apply a third-degree polynomial to the augmented linear coefficients over each distinct type of cycle, or step, see Equation 2. The fourth layer is the unraveled vector version of the third layer, and is the input to the neural network model. Each blue layer is part of the model with learned weights. There are 4032 hidden units in the fifth layer, which is a learned linear combination of the polynomial coefficients. After the activation is applied, the hidden units are again linearly combined into the final numeric output, using learned weights. This diagram was created using a modified version of [3].

5 Results

When considering the relative success of this project, it is important to consider the ability of the model to generalize to *new* data and thus demonstrate its ability to learn. Our constrained version of LOOCV allowed us to perform hyperparameter search to select hyperparameters that will generalize well to unseen instances of wafers from this production process. By performing LOOCV, we limit the bias that is introduced when hyperparameters are selected from validating on only a small subset of the data. Instead, the method uses every part of the training dataset for validation. This limits the amount of influence that a single validation wafer would have on the chosen hyperparameters. To estimate the error rate of the model on new data, we held out two testing wafers thought to be from very different sections of the distribution. For each of the output variables we focused on, we computed testing error and the corresponding measurement error on each wafer.

Mean Squared Error obscures the units of the error because it is squaring the error and taking the mean over many samples. Technically the units are nm^2, but this does not translate to a tangible quantity. The measurement error metric is more meaningful because it is the average error *in nanometers* among predictions of output variables on unseen data. Using the appropriate hyperparameters in Table 1, we are able to accurately regress the Remaining Mask and Recess output measurements *approaching the sensitivity of the imaging machine.*

Variable	**Wafer**	**Testing Error (MSE)**	**Measurement Error (nm)**
Remaining mask	R22	9.11268	2.73841
Remaining mask	R34	0.31283	0.44993
Recess	R22	191.34170	13.73289
Recess	R34	194.64074	13.87892

Table 1: Estimated generalization errors for two separate models, each regressing a different output variables on the two wafers kept in the held-out test set.

As we can see in the above table, there is a much larger error in predicting the value of Recess. This is thought to be predominantly caused by two factors: first, the scale of the Recess variable is much larger than the scale of the Remaining mask variable, and second, as seen in Appendix C, the hyperparameters resulted in training curves that are much more ideal for the Remaining mask model than for the Recess model. Future work includes determining a separate set of approximated optimal hyperparameters for the Recess variable, as well as the other two variables, CD Trench and CD Trench Mask, which were not analyzed to the degree of the two discussed here.

6 Conclusion

The prediction of process results in general is a difficult task, particularly in the context of plasma etch integrated circuit fabrication, in which the 3D plasma behavior is too complex for current physics modeling techniques. Additionally, running experiments to empirically understand the plasma conditions is too costly and time-consuming. Furthermore, with such a small dataset as was provided for this project, the limits of the efficacy of ML were being tested. That said, with the constraints on the project, we feel the results are promising and, especially with a larger dataset, a full ML prediction pipeline, that ingests sensor data from the reactors, makes predictions, and prescribes recipe adjustments, could be implemented in both the production and research settings to aid a skilled engineer in refining recipes for specific wafer structure, mask design, and reactor settings.

In general, the model is robust to variable-length time series data without requiring complicated neural network architectures. From a data-type perspective, the only requirements for this model to be trained or predict on a new instance is that the data generating process contains the three sensor variables we isolated and contains multiple cycles of the seven steps identified as influential in predicting the outputs. The robustness of this model is particularly convincing due to the sparseness and variety of the sample from which the model was

trained, as described in Section 3, paired with its relative generalization error in predicting the measurements of the post-production wafer seen in Table 1.

Personal growth from working on this project came from using a real-world dataset, learning proper statistical methods for processing and using data, applying various mathematical techniques to alter data, and exploring ML models with varying degrees of efficacy in accomplishing this task. Furthermore, the experience of working with engineers in a manufacturing environment and working with limited data to prove feasibility of the desired mapping relations was invaluable.

These factors considered, the project can be concluded to have succeeded in its goal of utilizing a small dataset to regress a difficult output to a realistic error for proof of concept for a larger project, incorporating a more significant dataset size and variety of both recipe and machine. Ideally, data science processes such as this could be accurate and robust across a wide variety of plasma etch silicon manufacturing processes and conditions to the degree that much of the process of recipe adjustment can be automated using ML pipelines. This project has accomplished its goals by resulting in a viable solution to the presented problem that tests the limits of practical machine learning.

7 Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof. Edward Rietman for the continuous support of this effort, for his guidance, motivation, knowledge, and for orchestrating and generously sharing the grant that provided both the data and funding for this project.

I would also like to thank my second committee member, Dr. Scott Riggs, for his attentiveness to the project during its critical periods, willingness to help me understand the plasma etch process, and encouragement throughout the research effort.

My sincere thanks also go to John Valcore, the lead engineer on the team, whose own

research efforts produced the data used for this project and without whose guidance and domain knowledge, this project would not have been successful.

I would also like to thank Lucas Frey, who was the engineer tasked with measuring the images that determined the output values for the model, which was crucial to the implementation of the model we settled on. Many others at Lam Research Corporation were instrumental to the success of this project and we extend our greatest appreciation to all those involved.

We would like to thank Lam Research Corporation for providing both the data and the grant funding for this project and the BINDS Laboratory in the College of Information and Computer Sciences for facilitating the project.

References

[1] Géron, A. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Inc., 2017.

[2] Goodfellow, I., Bengio, Y., and Courville, A. *Deep learning*. MIT Press, 2016.

[3] Iqbal, H. Harisiqbal88/plotneuralnet v1.0.0, Dec. 2018.

[4] Kim, B., Kim, D. H., Park, J., and Han, S. S. Prediction of surface microtrenching by using neural network. *Current Applied Physics 7*, 4 (2007), 434–439.

[5] Kim, B., Kwon, M., and Kwon, S. H. Modeling of plasma process data using a multi-parameterized generalized regression neural network. *Microelectronic Engineering 86*, 1 (2009), 63–67.

[6] Kim, B., and Lee, B. T. Prediction of sic etching in a nf 3/ ch 4 plasma using neural network. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films 22*, 6 (2004), 2517–2522.

[7] Kim, B., and May, G. S. An optimal neural network process model for plasma etching. *IEEE Transactions on Semiconductor Manufacturing 7*, 1 (1994), 12–21.

[8] Kim, B., and May, G. S. Real-time diagnosis of semiconductor manufacturing equipment using a hybrid neural network expert system. *IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part C 20*, 1 (1997), 39–47.

[9] Kirk, M. *Thoughtful Machine Learning with Python: A Test-driven Approach*. O'Reilly Media, Inc., 2017.

[10] Lynn, S. A., Ringwood, J., and MacGearailt, N. Global and local virtual metrology models for a plasma etch process. *IEEE Transactions on Semiconductor Manufacturing 25*, 1 (2012), 94–103.

[11] Rietman, E., and Lory, E. R. Use of neural networks in modeling semiconductor manufacturing processes: An example for plasma etch modeling. *IEEE transactions on semiconductor manufacturing 6*, 4 (1993), 343–347.

[12] Rietman, E. A. A neural network model of a contact plasma etch process for vlsi production. *IEEE transactions on semiconductor manufacturing 9*, 1 (1996), 95–100.

[13] Rietman, E. A., and Beachy, M. A study on failure prediction in a plasma reactor. *IEEE Transactions on Semiconductor Manufacturing 11*, 4 (1998), 670–680.

[14] Rietman, E. A., Whitlock, S. A., Beachy, M., Roy, A., and Willingham, T. L. A system model for feedback control and analysis of yield: A multistep process model of effective gate length, poly line width, and iv parameters. *IEEE transactions on semiconductor manufacturing 14*, 1 (2001), 32–47.

[15] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning* (2008), ACM, pp. 1096–1103.

[16] Wolf, S., and Tauber, R. N. Silicon processing for the vlsi era, vol. 1: Process technology. *and 526* (1986), 388.

[17] Zeng, D., and Spanos, C. J. Virtual metrology modeling for plasma etch operations. *IEEE Transactions on Semiconductor Manufacturing 22*, 4 (2009), 419–431.

Appendices

A Hyperparameters Appendix

Attribute	**Value**
Layers	0, 1, 2, 3
Hidden units	32, 125, 252, 350, 1e3, 2e3, 3.5e3, 4032, 5e3
Activation function	Hyperbolic Tangent, ReLU
L2 Regularization Constant	0, 1, 5, 50, 100
Learning Rate	1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7
Learning Decay	0, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5
Learning algorithm	Stochastic Gradient Descent, Adam
Epochs	1, 10, 25, 50, 75, 100, 150, 500, 1e3, 1e4
ES Minimum Delta	0, 1e-100, 1e-20, 1e-10
ES Patience	0, 1, 5, 10, 25, 50

Table A.1: Enumeration of all hyperparameters compared to determine the settings for the model that generalized best as defined by the cross-validation procedure.

B Cross-Validation Errors Appendix

B.1 Remaining Mask

Fold	Training Errors	Validation Errors	Measurement Errors (nm)
0	-1.61190	-10.71951	3.27001
1	-2.17978	-3.46008	1.81944
2	-2.44279	-0.29940	0.47359
3	-2.20296	-3.69779	1.91397
4	-2.30286	-2.12411	1.37505
5	-2.20845	-3.37437	1.82897
6	-2.31366	-1.84986	1.30542
7	-2.26967	-2.35925	1.49089
8	-2.44939	-0.15040	0.33892
9	-2.41733	-0.58120	0.46082
10	-2.39607	-1.06960	0.89664
11	-2.35517	-1.33912	1.13999
Mean	-2.26250	-2.58539	1.35948

Table B.1: The errors catalogued by the cross-validation procedure for the output variable of Remaining Mask. We notice a fair amount of variety in the measurement error, indicating that certain wafers are more extreme examples with respect to the dataset than others.

B.2 Recess

Fold	Training Errors	Validation Errors	Measurement Errors (nm)
0	-239.70212	-62.31758	7.57505
1	-164.60648	-26.26657	4.76816
2	-216.93628	-13.48704	2.97182
3	-200.22007	-28.50959	5.23397
4	-218.38041	-11.94878	2.78657
5	-218.47845	-5.04322	1.89414
6	-196.88474	-19.16364	4.27496
7	-198.58153	-9.00566	2.70473
8	-163.59976	-6.82301	2.26224
9	-162.42805	-17.24353	3.91904
10	-162.77509	-12.34732	3.39943
11	-258.79387	-0.89413	0.90290
Mean	-200.11557	-17.75417	3.55775

Table B.2: The errors catalogued by the cross-validation procedure for the output variable of Remaining Mask. We notice a fair amount of variety in the measurement error, indicating that certain wafers are more extreme examples with respect to the dataset than others.

C Learning Curves Appendix

C.1 Remaining Mask

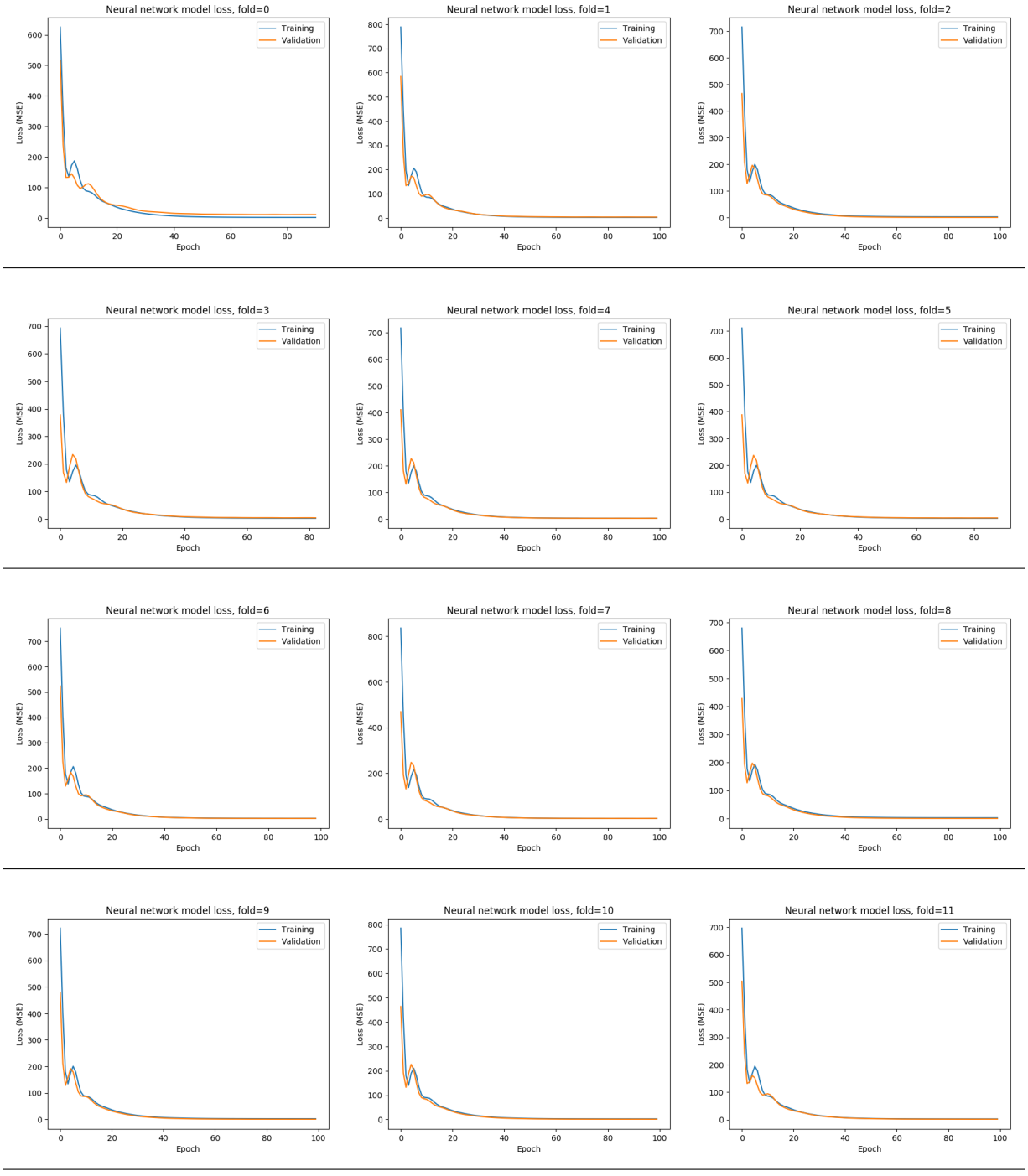

Table C.1: Loss curves for each fold of training and validation mean squared error during the cross-validation regressing the Remaining Mask output variable. The orange represents validation error and the blue represents training error. As we can see above, the validation curve and the training curve both asymptote toward the Bayes error rate, and early stopping allows us to halt model training before the validation error rises again. In this way, we avoid

C.2 Recess

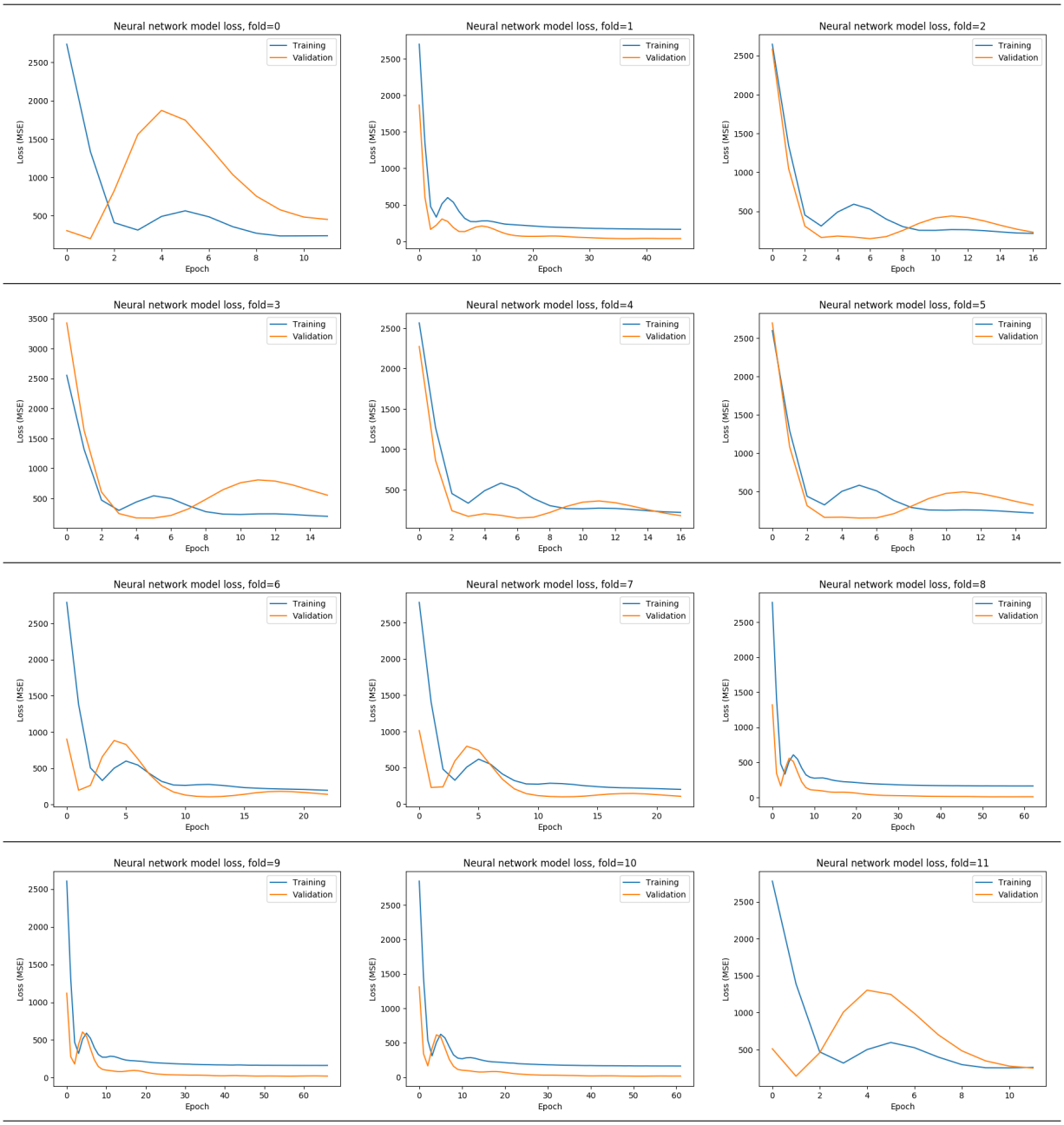

Table C.2: Loss curves for each fold of training and validation mean squared error during the cross-validation regressing the Recess output variable. Notably, the hyperparameters selected are more well-suited for learning the Remaining Mask output variable, likely because of the limited scale of that variable. Future work includes, determining separate hyperparameters for each desired output variable, including the other two available not detailed in this report.