Proceedings of the ASME 2016 International Design Engineering Technical Conference & Computers and Information in Engineering Design
IDETC/CIE 2016
August 21-24, 2065, Charlotte, North Carolina, USA

DRAFT 2016-59664

EVALUATING THE USE OF ARTIFICIAL NEURAL NETWORKS, GRAPH THEORY, AND COMPLEXITY THEORY TO PREDICT AUTOMOTIVE ASSEMBLY DEFECTS

Apurva Patel
Research Assistant
Mechanical Engineering
Clemson University
Clemson, SC 29634-0921
apurvap@g.clemson.edu

Patrick Andrews
Research Assistant
Mechanical Engineering
Clemson University
Clemson, SC 29634-0921
pcandre@g.clemson.edu

Joshua D. Summers
Professor
Mechanical Engineering
Clemson University
Clemson, SC 29634-0921
jsummer@clemson.edu

ABSTRACT

Artificial Neural Networks (ANNs) have been used to predict assembly time and market value from assembly models. This was done by converting the assembly models into bipartite graphs and extracting 29 graph complexity metrics which were used to train the ANN prediction models. This paper presents the use of sub-assembly models instead of the entire assembly model to predict assembly quality defects at an automotive OEM. The size of the training set, order of the bipartite graph, selection of training set, and defect type were experimentally studied. With a training size of 28 parts, an interpolation focused training set selection, and second order graph seeding, over 70% of the predictions were within 100% of the target value. The study shows that with an increase in training size and careful selection of training sets, assembly defects can be predicted reliably from sub-assemblies complexity data.

Keywords: Complexity, Assembly, Defect Prediction, Artificial Neural Networks

1. MOTIVATION: ENHANCING DECISION MAKING THROUGH PREDICTION MODELS

Engineering design has often been perceived as a compound decision making task [1–5]. Designers are challenged to make decisions in engineering activities such as problem definition, concept selection, geometric sizing, configuration, manufacturing, distribution, or marketing [1,2,6,7]. As these decisions directly impact the product development process, designers continually seek ways to make design decisions earlier in the design process to reduce the cost of changes that occur late in the design stage [8–11]. For instance, during the assembly of a new automotive design, an alignment issue is identified for the door handle, which causes the associate on the assembly line to occasionally scratch the exterior surface of the panel. A fix for this change may require a modification in either the door handle or the panel design, resulting in a modification of the tool used to manufacture the particular part. When in production, a change will be difficult and costly to implement [12–14]. However, if the issue could have been predicted earlier, perhaps before the fixtures and tools are developed, the solution would be easier and less costly to implement. Using historical data to develop models that can predict assembly defects in the early stages will help reduce the cost of design changes later in the development process. With such a method, engineers will be able use solid models along with automated processing to predict the number of assembly defects for specific parts. This information can then be used to make decisions about different configurations and architectures of the assemblies and part interactions to minimize these defect issues. Research demonstrates that Artificial Neural Networks (ANNs) can be used to predict assembly time using product complexity information [15–22]. In this paper, the use of such a method is evaluated for the purpose of predicting assembly defects in automotive manufacturing.

ANNs have been shown to have powerful prediction capabilities and do not require model assumptions by using Socratic learning to generate prediction models [23]. However, ANNs are black box models and do not provide any direct understanding about the relationships between the inputs and outputs. These relationships must be deduced externally. ANN modeling is a promising option for predicting quality in a complex assembly, as the relationships between known information and the desired information in cases such as these are highly nonlinear and context dependent. This makes derivation of the relationships difficult, motivating to the use of alternative methods that can be black box approaches.

In order to use an ANN prediction tool, it is necessary to develop a representation for parts that could be used to set up a relationship between the parts and the desired output. In order to

address this need, connectivity information is collected for the entire assembly. This connectivity data is then used to generate bipartite graphs of varying orders (depths) for individual parts. A bipartite graph separates the nodes in the graph into two groups that are not directly interconnected. Here, the first group of nodes is the "part" group and the second group of nodes is the "contact relation" group. This is a generalized approach developed to support other types of engineering design graph and network representations [24]. For the assemblies, a connectivity graph generated by detailing connections of a system by grouping the components into pairs. An example of a bipartite graph is shown in Figure 1.

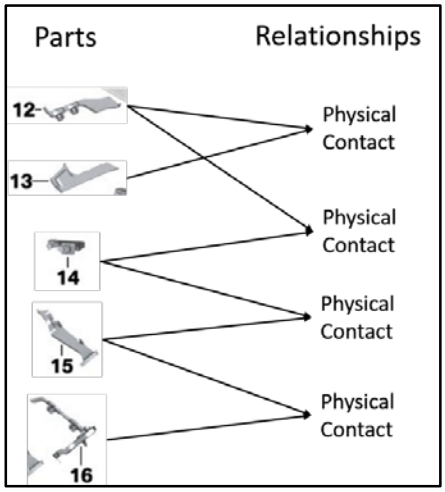

Figure 1: Bi-Partite Graph of Assembly Model

As shown in figure above, the bipartite graph has two main components: the parts and the relationships between two parts. Part 12 is in physical contact with Part 13. This is captured through a physical contact relation. Further, Part 12 is in physical contact with Part 14. In this case, the physical contact is determined through a clash detection algorithm in a CAD software [20]. Results from the clash detection analysis are then used to compile a list of connections between two parts, with one identified as a source, and the other as sink. The complete list of connections is subsequently used to develop bipartite graphs for individual parts. From these graphs, complexity metrics are mined [25], and the metrics are then used as the input for training the ANNs [21,26,27].

2. RESEARCH QUESTIONS

The primary objective of this research was to evaluate the capabilities of Artificial Neural Networks in predicting assembly defects using connectivity information for the constituent parts. Specifically, graph complexity metrics generated from a bipartite graph were to be used as inputs for the prediction tool. Therefore the following research questions are defined:

- Can ANNs trained with part complexity and defects be used to predict assembly defects for an automotive manufacturing facility?
- Can ANNs be used with sub assembly models, instead of complete assembly models, to predict part information for a large automotive manufacturing facility?

In order to address these questions, a series of experiments were conducted to evaluate the prediction capabilities of an artificial neural network structure given a relatively small data set and a large variance in the prediction targets. An existing ANN structure, described in [17,19] is used for the actual prediction process. In previous research, the entire assembly model is used to predict the complete assembly time. In this research, the defects for individual parts are of interest; training with the entire assembly model is not possible. Therefore, it is critical to determine the size of the graph seeded around the part of interest that will generate the best prediction models. It should be noted that this research is focused only on the topological adjacency graphs of the assemblies and other part information (material, size, mass) and joining information (overlap area, joint type) are explicitly not considered. Incorporating this other information is reserved for future work to extend the tool.

3. EXPERIMENTAL PROCEDURE

The artificial neural networks used for the experiments presented in this paper are trained using back propagation with a cascade-forward network [28–30]. A backpropagation architecture was chosen because it was suitable to the type of inputs being used and the type of outputs that were desired. Additionally, a cascade-forward network was selected because the mechanics allow for better accommodation of small data sets as each layer receives the original input as well as the outputs from all previous layers [15].

3.1.Basic Procedure

A general procedure used for these experiments is outlined in [31]. A graphical representation of this process is shown in Figure 2.

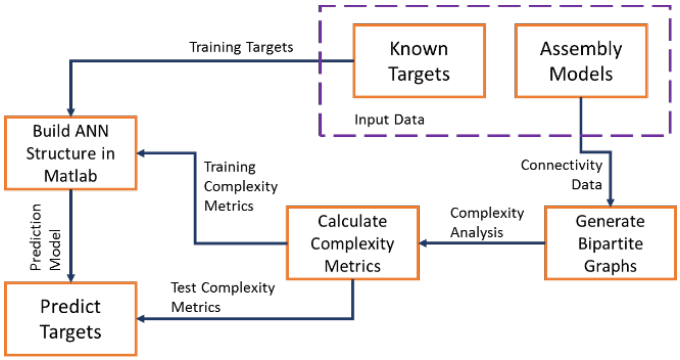

Figure 2: Basic Procedure for ANN Prediction

As shown in Figure 2, the basic procedure for ANN prediction has five main steps. First, the information used for prediction and the performance metric being predicted need to be collected. In this case, an assembly model is being used for prediction. Next, the assembly model is transformed into a bipartite graph. Graph complexity metrics are generated from this bipartite graph using a complexity analysis tool. The complexity metrics and the known targets with associated parts are then used to train an ANN structure which estimates a

relationship between the input complexity metrics and the desired output target. This trained and populated ANN structure is next used to estimate the targets for new complexity metrics.

Figure 3 shows the standard experimental procedure for predicting the assembly quality data of a product using the entire assembly model.

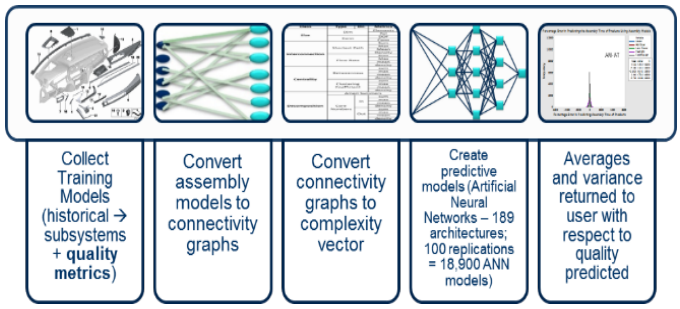

Figure 3: Procedure for ANN Prediction with Assembly Models

The procedure is similar to that described in Figure 2, and follows converting the assembly models into bipartite graphs and extracting the complexity metrics from the bipartite graphs. This complexity vector is then used as the input for the training procedure. The assembly quality data gathered are used as the training targets.

The goal of the research presented in this paper is to predict assembly defects for individual parts in an assembly. For this purpose, the complete assembly model cannot be used because that does not provide any differentiation between parts in the assembly. Therefore, a different description of the parts in an assembly is needed to use for ANN training. The first phase of the basic procedure shown in Figure 3 was modified in order to allow for a different representation of part information. Previously, the entire assembly model was being used for training models. The experiments presented in this paper use an assembly model that is pruned based on the connections of one part. Figure 4 below shows the growth of the connectivity graph for an individual part from that of the entire assembly.

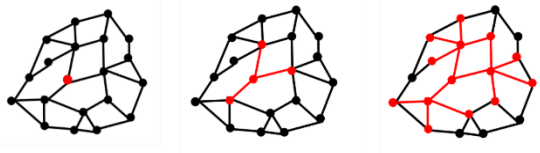

Figure 4: Graph order growth of assembly model

In Figure 4, the first image shows the connectivity graph of a complete assembly model, with a part of interest identified. The second image shows a first order connectivity graph for that part. Finally, the third image shows a second order connectivity graph for the identified part. As shown in Figure 4, first order connectivity graph is every part in direct contact with the part of interest, which can be referred to as the first order connections. A second order connectivity graph is a model consisting of all the first order connections, as well as all parts in contact with the first order connections, which are the second order connections. A third order connectivity graph adds third order connections to the model in a similar manner.

The use of partial connectivity graphs from the assembly models represents a significant change from previous experiments with ANNs and assembly models. Instead of the information available from the complete assembly model, only the information available from the pruned assembly model is being used to predict the assembly defects. As such, information that is more than a specified degree of connectivity away from the part of interest is not available to develop the prediction models. For automotive manufacturing, this is particularly important because many of the parts are in physical contact with the vehicle body. Once the vehicle body appears in the connectivity graph for a part, the graph will likely be saturated. Saturation will cause the higher order bipartite graphs to approach the entire assembly. This results in a situation similar to previous work with ANNs using the entire assembly as instead of a sub-assembly. The effects of training with saturated graphs are not known, however, there has been research that shows it can help identify categories of parts [32].

3.2.Detail Experimental Procedure

In order to evaluate the use of ANNs as a prediction tool for assembly defects, the assembly model for an automobile was collected. A clash detection analysis on the assembly model was used to generate a bipartite graph showing all the physical relationships between components in a source-sink form.

3.2.1.Generating Individual Bipartite Graphs

This complete bipartite of the assembly model was then pruned based on the specific parts for which assembly defect data was available. Figure 5 shows the process used to extract bipartite graphs from the complete assembly bipartite.

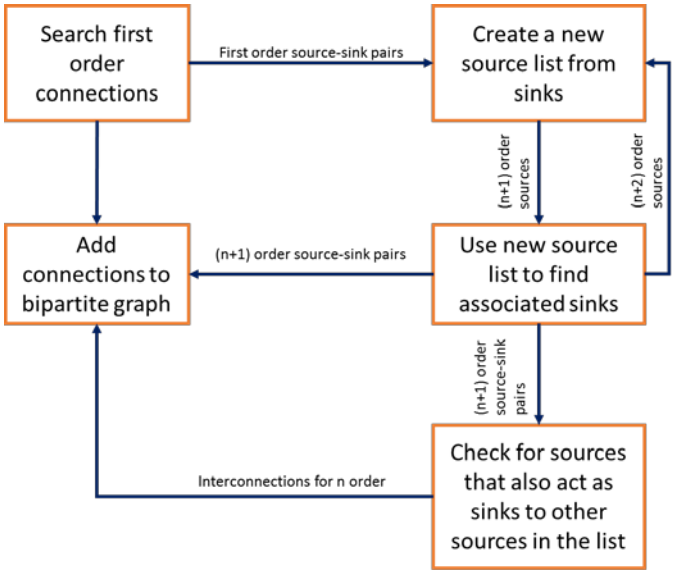

Figure 5: Algorithm used for generating bipartite graphs for individual parts

The algorithm shown in Figure 5 can be used to generate a bipartite graph for any order. However, for the purpose of this

experiment, only first, second and third order bipartite graphs were generated.

The bipartite graph consists of two main parts: the first to n^{th} order connections, and the n^{th} order interconnections. The input to this script is an excel file containing the complete assembly bipartite and the part numbers for which the bipartite graphs need to generated. As shown in the figure, the first step to extracting the bipartite graph of one part from the entire assembly is to search for first order connections. This means that the source column of the assembly bipartite is filtered for the part number of interest. The associated part numbers in the sink column are first order sinks. Next, the first order sinks are used as sources to find next order sinks. This process is repeated until the desired graph order is achieved. The final step is to search the connections one order above the desired graph order for parts that act as sources as well as sinks.

3.2.2. Calculating Complexity Metrics

After generating the bipartite graphs, the next step is to calculate 29 complexity metrics for each part. The method used to generate the complexity vector is presented in [12]. Using this method, a complexity vector containing 29 complexity metrics was generated for each part being used in the experiment. An example complexity vector is shown in Table 1.

Table 1: Example Graph Complexity Vectors

Complexity Metric			Part 1	Part 2	Part 3
Size	Dim	Elem.	6	4	17
		Rel.	5	3	23
	Con	DOF	5	3	23
		Conn	10	6	46
Interconnection	Path	Sum	11	5	59
		Max	2	2	2
		Mean	0.367	0.417	0.217
		Density	0.073	0.139	0.009
	Flow	Sum	13	7	72
		Max	2	2	15
		Mean	0.361	0.438	0.249
		Density	0.072	0.146	0.011
Centrality	Between	Sum	3	1	18
		Max	2	1	15
		Mean	0.50	0.25	1.06
		Density	0.10	0.08	0.05
	Cluster	Sum	0	0	2.946
		Max	0	0	0.5
		Mean	0	0	0.173
		Density	0	0	0.008
Decomp	Ameri-Summers		7	4	47
	Core In	Sum	5	3	23
		Max	1	1	4
		Mean	0.833	0.750	1.353
		Density	0.167	0.250	0.059
	Core Out	Sum	5	3	23
		Max	2	2	15
		Mean	0.833	0.750	1.353
		Density	0.167	0.250	0.059

The set of 29 complexity metrics describe the connectivity of an individual part within the complete assembly with respect to a specific graph order. The complexity metrics are divided into four main categories: size, interconnection, centrality, and decomposition. These are directly derived from the bipartite graph, and therefore will vary between graph orders of the same part. Moreover, if two different parts have the same bipartite graph, the complexity metrics will also be identical.

After calculating the complexity metrics, the next step is to acquire the performance metrics, or the prediction targets. In this case, the assembly defects for the relevant parts were collected. Defects data was only collected for the vehicle model that was used to generate the initial bipartite graphs. The defects were categorized into gap, loose, wrong, missing, miscellaneous, and total. The different categories were used as different training targets. One set of the defects data is shown in Table 2.

Table 2: Initial Defects Data Set

Part #	Total	Gap	Missing	Wrong	Misc.	Loose
Part 1	53	16	0	35	0	2
Part 2	131	111	0	7	0	13
Part 3	35	0	0	29	0	6
Part 4	3932	3750	29	128	4	21
Part 5	34	31	0	1	0	2
Part 6	22	11	0	9	0	2
Part 7	1248	640	0	513	0	95
Part 8	155	88	0	64	1	2
Part 9	5	4	0	1	0	0
Part 10	55	51	0	4	0	0
Part 11	57	50	0	3	0	4
Part 12	1941	1506	0	431	0	4
Part 13	53	16	0	35	0	2
Part 14	265	164	1	84	1	15
Part 15	267	55	0	3	0	209
Part 16	167	19	39	86	23	0
Part 17	203	108	0	94	0	1
Part 18	1941	1506	0	431	0	4
Part 19	125	71	0	39	0	15
Part 20	246	194	0	33	0	19
Part 21	631	533	0	65	33	0
Part 22	85	19	14	14	37	1
Part 23	1121	378	35	119	551	38

3.2.3. Selection of Experimental Sets

During the experiment, it was necessary to select a group of parts to be used as the training set and the other parts to be used as the testing set. The parts were divided so that roughly 80% of the parts would be used for training and the remaining parts would be used for testing. Five different methods were used for selecting the training and test sets. The first group of sets were chosen at random. The following sets were chosen using vector distance calculations from the mean vector of all of the parts. The first method was to use the absolute distance from the mean vector and order the parts from nearest to the mean to farthest for all 3 orders. The testing set was chosen as the parts nearest to the mean across all 3 orders of graph. The second group of sets was chosen using the cosine distance from the mean instead of the absolute distance. A similar selection process across all 3 orders was used. This process was then repeated using only the 11 metrics identified as being significant in previous work [19].

This was done in an attempt to increase the accuracy of the prediction as ANNs are typically trained on large data sets. Since extensive defect data was not available, alternative methods to increase the accuracy of the prediction were investigated. This increases the likelihood of interpolating amongst the data and not extrapolating. An example of extrapolating test set can be seen in Figure 7, while an interpolating test set can been seen in Figure 8. The parts reserved for testing are orange and the training set is blue. In both figures, the rectangle shows the area encapsulated by the training set. Any parts that fall outside this area will require the estimations to be extrapolated. Ideally, graphs shown in Figure 7 and Figure 8 would need to be generated for all combinations of pairs within the 29 complexity metrics. However, in order to simplify the pairwise comparisons, clustering approaches such as finding the minimum cosine distances were used to identify interpolating test sets.

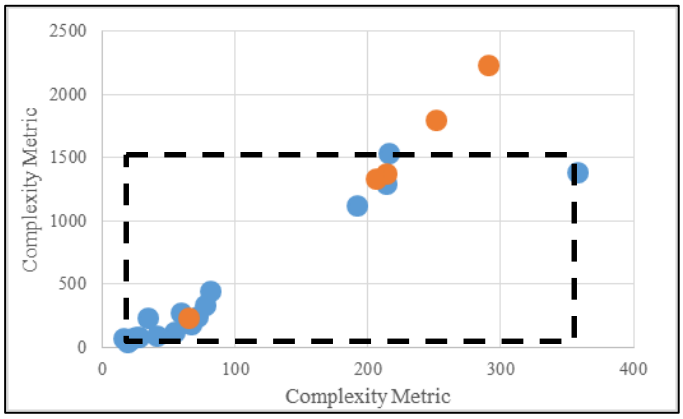

Figure 6: Extrapolating test set

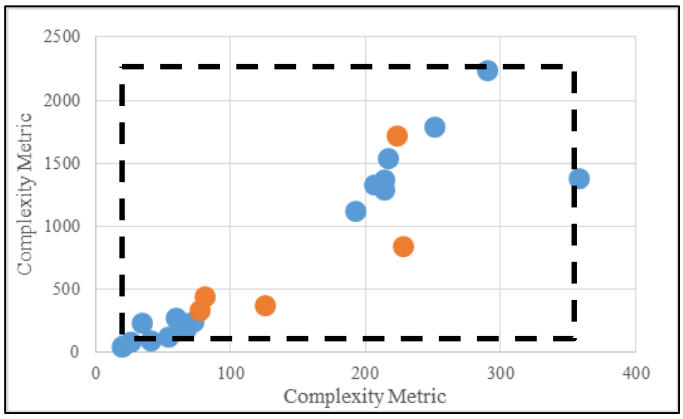

Figure 7: Interpolating test set

3.2.4. Training and Testing of ANN Structure

The ANNs structure was then trained using the assembly complexity information from the training sets with the corresponding assembly defect information. To reduce the variability of the ANN training process with small data sets, 189 different ANN architectures were used. The training process for each architecture was replicated 100 times resulting in 18,900 unique ANNs. Figure 9 shows a graphical representation of the process used to build and test the ANN structure.

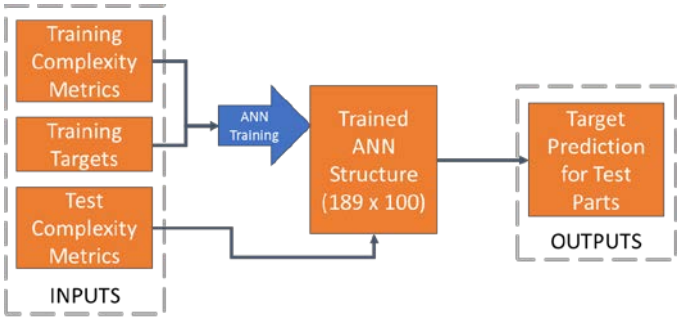

Figure 8: Building and testing the ANN structure

The trained ANNs were then used to predict the assembly defects for the test sets using the test set assembly complexity information. The 18,900 outputs were then averaged as the prediction for the assembly defects. Using the predictions and the existing assembly defect data for the test set, several different error values were generated.

3.2.5. Error Calculation

The predictions generated from the ANN needed to be compared to the known values to understand the results of the experiment. As the prediction targets ranged widely, three different error calculation methods were used. For prediction targets between 0 and 15, a residual error was used. Equation 1 shows how the residual error was calculated.

$$E_{res} = Target - Predicted \quad (1)$$

In the Equation 1, *Target* refers to the known target value, while predicted refers to the value prediction by the ANN. For this case, the error was considered to be larger than 100% when the residual was greater than 15. Similarly, if the residual was between 15 and 30, the error was considered to be between 100% and 200%. Finally, any residuals larger than 30 were considered to have an error higher than 200%.

Next, for prediction targets between 16 and 200, a standard percent error was calculated. Equation 2 was used to calculate this error.

$$E_{std} = \frac{|Target - Predicted|}{Target} \quad (2)$$

Finally, for prediction targets of 201 and higher, a normalized error was calculated using Equation 3.

$$E_{norm} = \frac{(Target - Predicted)^2}{|Target \times Predicted|} \quad (3)$$

The primary objective of using different error calculation methods was to better represent the impact of the errors in prediction.

4. EXPERIMENTS ON TRAINING CHOICE

In order to better understand the predicting capabilities of the neural network structure being used, certain variables were tested to understand their relationship with the prediction accuracy. The variables tested were as follows.

- Training size (Experiment 1)

- Test group selection (Experiment 2)
- Graph order (Experiment 3)
- Defect type used as target for prediction (Experiment 4)
- Multiple defect data sets (Experiment 5)

4.1. Experiment 1: Training Size

It is common for ANN prediction models to be developed using a large training set. However, for this project, the training set available was relatively small. Therefore, a compound neural network made from 189 different architectures repeated 100 times each was used. However, it was still important to understand how the prediction models would behave if larger training sets were available.

The initial training data used consisted of 10 training parts and 5 test parts. An example of the prediction results for this test are shown in Table 3.

Table 3: Defect Prediction with 10 Training Parts

Part	Known Defects	Predicted Defects	Error
Part 1	53	2297	4136%
Part 2	34	277	627%
Part 3	265	1216	281%
Part 4	155	-423	510%
Part 5	97	79	4%

Subsequent tests were conducted with 18 parts and 28 parts for training. The overall results from these tests are shown in the Figure 10.

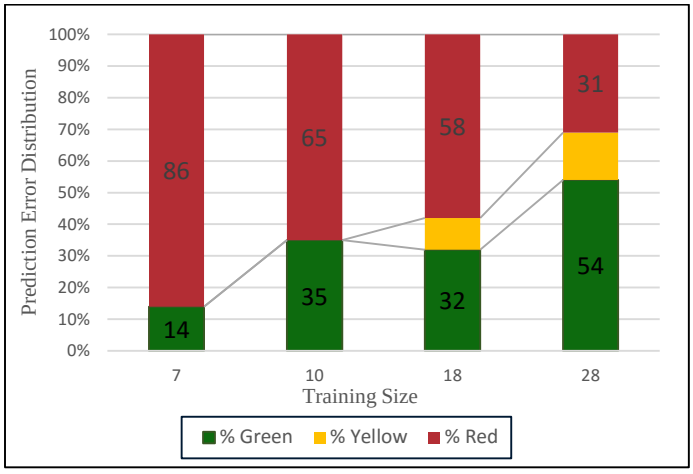

Figure 9 : Error distribution with respect to training size

In Figure 10, the red section of the bar shows prediction errors higher than 200% of the target value. The yellow sections of the graph show errors between 100% and 200% of the target value, whereas the green section shows the errors less than a 100% of the target value. As shown in Figure 10, with an increasing training size, the percentage of predictions with error less than 100% has increased from 14% at 7 parts in training to over 50%, at 28 parts. The larger data sets introduced the intermediate predictions where the error was significant but within a reasonable order. This is a growth from the smaller training sets where the bad predictions were typically several orders of magnitude off. This is a challenge that modelers using ANNs face frequently when dealing with small data sets. The continued growth of the training sets is a goal of this project to increase the prediction accuracy.

4.2. Experiment 2: Graph Order

In addition to the size of the training set, the order of the bipartite graph generated for the part was also experimented with to determine an appropriate level of detail needed for predictions. In the early stages of the project, only first and second order graphs were available to experiment with. Table 4 contains the results of 5 test parts being tested across 2 orders of assembly models (1st order, 2nd order) and 4 assembly defects (Gap, Loose, Wrong, and Total).

Table 4: First and Second Order Predictions for Test Parts

		First Order				
		Part 1	**Part 2**	**Part 3**	**Part 4**	**Part 5**
Gap	Targ.	16	111	0.001	3750	31
	Pred.	519.36	519.36	357.19	187.04	395.93
	Error	3	3	3	3	3
Loose	Targ.	2	13	6	21	2
	Pred.	47.05	47.05	23.25	76.96	75.11
	Error	3	3	2	3	3
Total	Targ.	53	131	35	3932	34
	Pred.	687.82	687.82	529.13	426.72	564.12
	Error	3	3	3	3	3
Wrong	Targ.	35	7	29	128	1
	Pred.	254.09	254.09	226.20	50.17	105.37
	Error	3	3	3	1	3
		Second Order				
		Part 1	**Part 2**	**Part 3**	**Part 4**	**Part 5**
Gap	Targ.	16	111	0.001	3750	31
	Pred.	452.05	264.61	292.05	528.05	77.67
	Error	3	3	3	3	3
Loose	Targ.	2	13	6	21	2
	Pred.	103.31	85.17	13.48	27.11	13.26
	Error	3	3	1	1	1
Total	Targ.	53	131	35	3932	34
	Pred.	671.29	436.90	386.85	669.70	164.93
	Error	3	3	3	3	3
Wrong	Targ.	35	7	29	128	1
	Pred.	289.24	222.43	72.63	120.04	37.10
	Error	3	3	3	1	3

These test parts were randomly selected from a pool of 23 parts where the remaining 18 parts were used for training. The error was calculated using the normalized percentage error. A green, or group 1, prediction was considered to be within 100%. A yellow, or group 2, prediction was considered to be from 100% to 200% error. A red, or group 3, prediction included errors greater than 200%. In the first order results, only one of the predictions was within 100%. The second order results improved to 4 predictions within 100%.

As the available data increased, third order assembly models were also compared to the results of the first and second order graphs. The experiments including third order graphs were done

simultaneously with other experiments and the specific results can be seen in the following sections. Figure 11 shows the results of the predictions based on the graph order used for training the ANN structure for all tests to date.

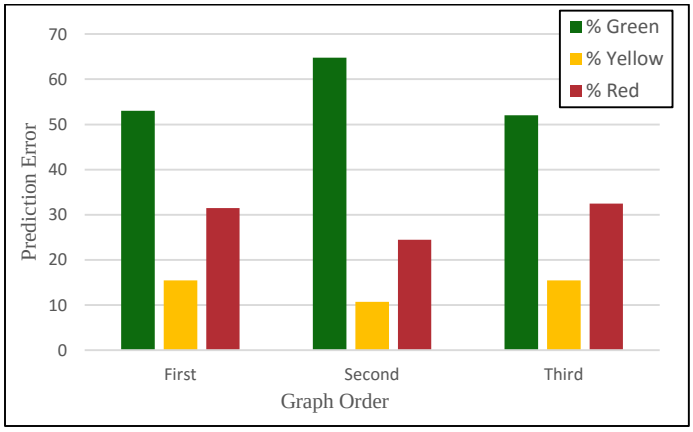

Figure 10: Error distribution based on graph order

As shown in Figure 11, second order graphs, where over 65% of the predictions were within 100% of the target, have the best predictions, while first and third order graphs behave roughly the same.

4.3. Experiment 3: Set Selection

Besides the size of the training set and the graph order, the selection of the training set was also experimented. As discussed previously, it is important to select a training set that encompasses the test set for which the ANN structure is being used to predict the targets. In order to verify this claim, as well as further understand the impact of changes in the training set, the training set selection was varied in five different alternatives: random grouping, cosine significant, cosine total, means significant, and means total. Table 5 shows the distribution of prediction error with five randomly selected groups that were used for testing the ANN structure.

Table 5: Error Distribution for Random Groups

	Groups	G1	G2	G3	G4	G5
First Order	%Green	5%	40%	35%	65%	40%
	%Yellow	5%	0%	5%	0%	5%
	%Red	90%	60%	60%	35%	55%
Second Order	%Green	20%	20%	25%	30%	35%
	%Yellow	0%	20%	20%	20%	0%
	%Red	80%	60%	55%	50%	65%

As shown in Table 5, the majority of the predictions in this experiment resulted in an error of 200% or more. Predictions generated from second order graphs were less accurate that those generated from first order graphs. First order group 4 resulted in the best prediction accuracy with 65% of the predictions being less than 100% error, whereas, first order group 1 was the least accurate prediction with 90% of the predictions being 200% or higher in error. The results of the four different set selection methods across all three orders of assembly model can be seen in Table 6.

Table 6: Selected Sets Error Distribution

	First Order	Second Order	Third Order	
Cosine Significant	60	62	48	Green (%)
	7	7	16	Yellow (%)
	33	31	36	Red (%)
Cosine Total	50	76	55	Green (%)
	19	12	16	Yellow (%)
	31	12	29	Red (%)
Means Significant	40	50	48	Green (%)
	20	14	16	Yellow (%)
	40	36	36	Red (%)
Means Total	62	71	57	Green (%)
	16	10	14	Yellow (%)
	22	19	29	Red (%)

The best performing selected test set was the means total set. The percentages of good predictions were 62%, 71%, and 57% for first, second, and third order, respectively. Second order performed the best across all four selected sets with an average of 64.75% good predictions. The best performing individual test was the cosine total set, second order with 76% good predictions. As a whole, the total method worked better than the significant method for selecting the test sets. This is not expected as the metrics deemed significant previously should be more important to the prediction than the entire complexity vector.

4.4. Experiment 4: Target Defect Type Selection

The next experiment conducted was to test the prediction capabilities for different types of defect targets. As shown in Figure 12, six total defect types were investigated: gap, loose, missing, miscellaneous, total, and wrong.

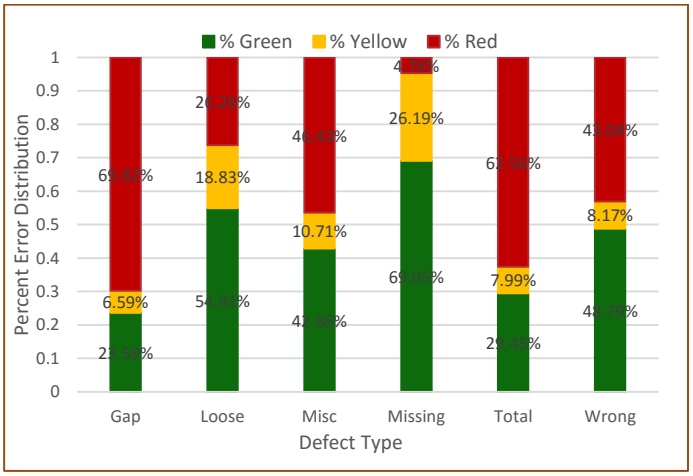

Figure 11: Error distribution based on defect type

From the ANN prediction experiments, the missing defect was found to be the most accurate, followed by loose, wrong, miscellaneous, total and gap, in that order. However, it is important to note that in initial defects data set, missing and miscellaneous defect type were not significant as they were mostly zero. As shown previously in Table 2, only 6 out of 23 parts contained non-zero defects for both missing and miscellaneous defect types. As such, prediction using missing and miscellaneous defect types was found to be unreliable. Table

7 shows the prediction results for missing and miscellaneous defect types from an experiment conducted with eighteen training parts and five tests parts.

Table 7: Defect Prediction for Missing and Miscellaneous

First Order, Random Grouping - Missing Defects					
	Part 1	Part 2	Part 3	Part 4	Part 5
Known Defects	0	0	0	1	0
Predicted Defects	-3	31	48	16	53
Second Order, Random Grouping - Miscellaneous					
	Part 1	Part 2	Part 3	Part 4	Part 5
Known Defects	0	0	0	1	0
Predicted Defects	38	40	41	10	21

According to Table 7, majority of the predictions will yield a residual larger than 30, resulting in an error greater than 200% as defined in the error calculation methods. After examining these results, the missing and miscellaneous experiments were dropped for this defects data set. When an updated data set was available, the missing and miscellaneous experiments were resumed as the number of defects in the new defects data set was reasonable.

In addition to the six defect types mentioned above, a bucketing approach to the total defects was also investigated. The list of parts with total defects was sorted in five buckets. The buckets are shown in the Table 8.

Table 8: Bucketing Parameters

Bucket	Defect Count (targets)
1	0 to 50
2	51 to 100
3	101 to 200
4	201 to 1000
5	> 1000

The bucketing approach was applied to the ANNs trained with four selected sets as mentioned earlier. The prediction results from means total set across three graph orders is shown in Table 9.

Table 9: Bucket Prediction Results

		Part 1	Part 2	Part 3	Part 4	Part 5
Order	Targ.	1	5	1	5	4
	Pred.	3	3	3	3	3
1st	Error	2	2	2	2	1
	Pred.	6	3	3	4	4
2nd	Error	5	2	2	1	0
	Pred.	5	3	3	4	0
3rd	Error	4	2	2	1	4

The targets are the bucket within which each defect count falls, and the predicted value is the output of the model when trained with the buckets instead of the actual number of defects. This was done in an attempt to increase the functionality of the predictions. It is also expected to help in dealing with the wide range of defects across different parts. Only three of the predictions lied within 1 bucket of the target value. Six predictions lied between one and two buckets of the target value.

In another attempt to manipulate the defect counts for prediction accuracy, the log of the defect counts was used. This attempt was looking to accomplish the same thing as the bucketing except becoming less subjective. The determination of the bucketing bounds was done manually. The results of log experiment will be calculated similar to the bucketing experiment with within 1 count of the predicted value will be a good prediction. The experiment was done using the means total selected group across all three orders of graph. Table 10 shows the results from using the base 10 log of the number of defects as a target for the ANN training.

Table 10: Log Defect Prediction Results

		Part 1	Part 2	Part 3	Part 4	Part 5
Order	Targets	1.53	3.10	1.74	3.29	2.42
1st	Pred.	3.08	3.30	2.96	3.36	2.90
	Error	2.08	1.70	1.96	1.64	1.10
2nd	Pred.	6.02	2.50	2.90	4.36	3.51
	Error	5.02	2.50	1.90	0.64	0.49
3rd	Pred.	5.04	2.59	2.60	4.19	0.28
	Error	4.04	2.41	1.60	0.81	3.72

There were twelve total predictions that lied within the acceptable range. This is a much greater improvement than the bucketing experiment. It should be noted that this is also a much larger spread than the ranges of the buckets and may be too wide of a range to be applicable.

4.5. Experiment 5: Target Data Selection

As the project progressed, new defects data was available. This resulted in an opportunity to test the ANN prediction capabilities with a larger data set as the two defects data sets could be combined to generate a new prediction model. The goal of this experiment was to understand whether the ANN prediction model would be more accurate if historical data from different intervals of time was available for training. Table 11 shows the prediction results when targets from both and old and new defects data set were selected for training.

Table 11: Defect Prediction with Multiple Defects Data Sets

	Target (Old)	Target (New)	Target (Average)	Prediction
Part 1	791	155	473	624.585
Part 2	267	1505	886	705.756
Part 3	167	1031	599	484.822
Part 4	265	2159	1212	446.286
Part 5	114	1384	749	557.245
Part 6	267	155	211	705.756
Part 7	155	395	275	1480.668
Part 8	395	128	261.5	781.459
Part 9	131	248	189.5	1095.017

The prediction shown in table above were analyzed in two different ways. One method was to compare the prediction to the average of old and new targets. The predictions for part 1 to part

5 were within a 100% of the average target. Part 6 to 9 however, resulted in predictions higher than 300% of the average target.

The second method was to check if the prediction fell within the bounds generated by using the old and new targets as minimum and maximum. As shown in table above, the prediction for Part 1 to Part 5 all fell within the bounds generated by old and new targets, whereas Parts 6 to Part 9 all fell outside the maximum target.

5. DISCUSSION AND CONCLUSION

The current collection of experiments represents a variety of variables and concepts influencing the ability to accurately predict assembly defects from pruned assembly models. In the first experiment, the impacts of the training size on the results was observed. It is shown that as the training size increases, the test results become increasingly better (from 14% to 54% of predictions within 100% error). This is an expected result as using ANNs is typically done with large data sets. Continually increasing the size of the data set will continue to positively impact the results of future experiments.

The second experiment conducted was on graph order of the pruned assembly models. This has design implications for manufacturing. The designers can learn how many connections away can decisions impact the assembly quality of a specific part. The current experiments included first, second, and third order assembly models. An average of all of the experiments run has second order graphs outperforming first and third order graphs by more than 10%. Reviewing specific experiments, instances can be shown where first order and second order are the best performing. To develop a better understanding of the effects of the graph order on predicting assembly defects, more testing must be completed. First, fourth and fifth order graphs should be generated to observe the effects of the higher orders. Simultaneously, the saturation rates of the assembly graphs should be studied for trends among parts with higher and lower saturation rates and the order of graph that is optimal for predicting the assembly defects. This is relevant because specific parts are highly integrated into the assembly as others are more isolated. The difference between the two parts could potentially provide insight into predicting the assembly defects. Second, graph order should continue to be isolated in experiments to observe the effects independently of other experimental changes.

The third experiment conducted involved specifically selecting test sets to ensure that the model was attempting to predict information that lied within the current set of training data. This was done as part of an effort to increase the accuracy of the predictions regardless of the size of the training set. The results of the experiment show a tremendous increase in the accuracy of the predictions once using selected tests sets as opposed to random. It is also viable to use similar vector mathematics to compare an unknown test part to the mean of the training vectors to see if the product lies within a reasonably predictable range. The total metrics sets outperformed the significant metrics sets in the experiment. This was unexpected as the total metrics set was considered to contain information that was not related to the prediction. This could mean that the entire complexity vector is necessary for an accurate prediction. Another possibility is determining the significant metrics through alternative techniques or performing a more robust validation on those decisions.

The fourth experiment was conducted to understand what types of targets were most suited for prediction. The experiment tested six different types of defects, as well as a bucketing of the total defects, and a log of total defects. Only the total defects were selected for this because total defects included all other defects types, and therefore was seen as a better representation of assembly quality. The experiment results showed that certain defect types consistently yielded better predictions. However, it should be noted that “missing” and miscellaneous” defect types only yielded good predictions when the training size was 28 parts and new defect data was used as targets. Besides using different defect types, a numerical manipulation of the number of defects was also used as targets in the form of “bucketing” and “log targets.” The results from these tests were different from those obtained from simply using number of defects. However, the prediction was not consistently better, and therefore the alternative targets were not considered reliable.

Finally, a combination of old and new defects data was used to generate predictions. This training set was chosen to understand the effects of multiple targets for the same part. The expectation was that training the ANN in this manner will generate a more flexible prediction model resulting in a reduction of the larger errors. It was also expected that the prediction would fall between the two targets provided, effectively feeding the ANN with a range of values as a target instead of a single number. The results from the experiment suggested that if the range is relatively large, it is likely that the prediction will fall in the range, however, if the range is no larger than 200% of the lower bound, the predictions will have a larger error.

As a whole, the experiment has led to a better understanding of the specific combinations of procedures that can be used to improve the prediction accuracy of assembly quality issues. Using as large as possible a training set with the selected sets has shown to be beneficial. As the data set increases, using selected sets can potentially be phased out as the introduction of this methodology was to combat the small data sets. For the graph order, more experimentation is required, however, second order graphs seem to be a reasonable solution at this time. Second order graphs make sense from a physical realization stand point as well, because in a single assembly, higher order bipartite graphs may approach saturation, resulting in a loss of the differentiation between parts.

The next significant challenge of the experiment is move beyond one vehicle of assembly data. Currently, all of the parts and defect data are extracted from one vehicle. The next step would be to repeat combinations of the previously performed experiments on a new vehicle. Also, determine if the defect data from one vehicle can be used to predict assembly defects on a separate vehicle, and the degree of accuracy for the predictions.

If data from one vehicle is able to prediction defects for another vehicle, the aspects of similarity between the vehicles should also be determined.

One of the greatest limitations on continuing the experiment moving forward is the ability to collect more part and defect information for said parts. This hinders the accuracy of the experiment greatly from the standpoint of the training the ANN. This is because the training algorithm begins with randomly generated weights and without a large training set, this "randomness" can be difficult to remove from the prediction. Another significant limiting factor of the experiment is the simultaneous change of two procedural steps from a previously successful model. By moving from the entire assembly model to pruned assembly models, and moving to assembly defects from assembly time or market value, it is possible that either change could be hindering better predictions from the model. Isolation of these two variables moving forward could potentially add insight to the model.

From the experiments performed, it is reasonable to believe that assembly defects can be predicted from pruned assembly graphs. The prediction accuracy has improved as a result of the various experiments conducted to determine the impact of different variables on the predictions. The most suitable combination of training size, graph order, and set selection found during the experiments was able to generate more than 70% of predictions that lied within a 100% of the target value. A general trend was identified from the experiment results which suggest that an increase in training size and the use of selected sets have positive impact on the prediction accuracy.

The success of the experiments so far encourage more work be done to improve the prediction model. This includes the addition of more training data to confirm that larger training sets will continue to improving the predictions. The order of the graph and the corresponding effects of this order on the prediction will also be observed more closely and isolated in experimental cases. The expansion of the experiments beyond third order will hopefully provide insight into some of the behaviors already observed among the first, second and third order experiments. Additionally, the use of a range of values as a target for prediction will also be investigated further.

ACKNOWLEDGMENTS

The authors would like to thank Erin Harrison and Joerg Schulte of BMW and Laine Mears and Amaninder S. Gill of Clemson for supporting the research project.

REFERENCES

[1] Pahl G., Beitz W., Wallace K., and Blessing L., 2007, *Engineering Design: A Systematic Approach*, Springer-Verlag London Limited, London.

[2] Ullman D. G., 2010, *The Mechanical Design Process*, McGraw-Hill, New York, NY.

[3] Hubka V., and Eder W. E., 1988, *Theory of technical systems*.

[4] Hazelrigg G., 1999, "An Axiomatic Framework for Engineering Design," *J. Mech. Des.*, **121**(3), p. 342.

[5] Hauser J. R., and Clausing D., 1988, "The house of quality," *Harv. Bus. Rev.*, pp. 63–73.

[6] Dieter G. E., and Schmidt L. C., 2013, *Engineering Design*, McGraw Hill, New York.

[7] Buede D. M., 2011, *The Engineering Design of Systems*, Wiley, Somerset.

[8] Phelan K., Wilson C., Summers J. D., and Kurz M. E., 2014, "A Case Study of Configuration Management Methods in a Major Automotive OEM," *ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, American Society of Mechanical Engineers, pp. V004T06A024–V004T06A024.

[9] Phelan K. T., Summers J. D., and Guarneri P., 2014, "Engineering change management - verification, validation and testing planning tool development," *Tools Methods Compet. Eng.*, pp. 1–12.

[10] Eckert C., Clarkson P. J., and Zanker W., 2004, "Change and customisation in complex engineering domains," *Res. Eng. Des.*, **15**(1), pp. 1–21.

[11] Clarkson P. J., Simons C., and Eckert C., 2004, "Predicting Change Propagation in Complex Design," *J. Mech. Des.*, **126**(5), p. 788.

[12] Shankar P., Mathieson J., Ramachandran R., Summers J. D., and Mocko G. M., 2012, "Review of the Design Tools and Methods to Predict Change Propagation Pathways," *Tools and Methods for Competitive Engineering (TMCE 2012)*, p. No–39.

[13] Shankar P., Morkos B. W., and Summers J. D., 2012, "Reasons for change propagation: a case study in an automotive OEM," *Res. Eng. Des.*, **23**(4), pp. 291–303.

[14] Jarratt T. A. W., Eckert C. M., Caldwell N. H. M., and Clarkson P. J., 2011, "Engineering change: An overview and perspective on the literature," *Res. Eng. Des.*, **22**(2), pp. 103–124.

[15] Mathieson J. L., 2011, "Connective Complexity Methods for Analysis and Prediction in Engineering Design," Clemson University.

[16] Mathieson J. L., Wallace B. A., and Summers J. D., 2010, "Assembly Time Modeling Through Connective Complexity Metrics," *International Conference on Manufacturing Automation (ICMA)*, pp. 16–23.

[17] Mathieson J. L., Shanthakumar A., Sen C., Arlitt R., Summers J. D., and Stone R., 2011, "Complexity as a Surrogate Mapping between Function Models and Market Value," *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, ASME, Washington, DC, pp. DETC2011–47481.

[18] Vasala S., 2014, “A Comparative Study: Structural Complexity Metrics Applied Against Function and Assembly Product Graphs in Predicting Market Price and Assembly Time,” Clemson University.

[19] Sridhar S., 2015, “Sensitivity and Precision Analysis of the Graph Complexity Connectivity Method.”

[20] Namouz E., and Summers J. D., 2014, “Comparison of Graph Generation Methods for Structural Complexity Based Assembly Time Estimation,” *ASME Trans. J. Comput. Inf. Sci. Eng.*, **14**(2), p. 021003.

[21] Namouz E. Z., and Summers J. D., 2013, “Complexity Connectivity Metrics – Predicting Assembly Times with Low Fidelity Assembly CAD Models Low Fidelity CAD Model,” *Smart Prod. Eng.*, pp. 777–786.

[22] Owensby J. E., Namouz E. Z., Shanthakumar A., and Summers J. D., 2012, “Representation: Extracting Mate Complexity from Assembly Models to Automatically Predict Assembly Times,” *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, ASME, Chicago, IL, pp. DETC2012–70995.

[23] Agatonovic-Kustrin S., and Beresford R., 2000, “Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research,” *J. Pharm. Biomed. Anal.*, **22**, pp. 717–727.

[24] Summers J. D., Vargas-Hernández N., Zhao Z., Shah J. J., and Lacroix Z., 2001, “Comparative study of representation structures for modeling function and behavior of mechanical devices,” *Proc. DETC2000 Comput. Eng.*, pp. 775–787.

[25] Mathieson J. L., and Summers J. D., 2010, “Complexity Metrics for Directional Node-Link System Representations: Theory and Applications,” *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, ASME, Montreal, Canada, pp. DETC2010–28561.

[26] Namouz E. Z., and Summers J. D., 2013, “Complexity Connectivity Metrics-Predicting Assembly Times with Abstract Assembly Models,” *Smart Product Engineering*, M. Abramovici, and R. Stark, eds., Springer Berlin Heidelberg, Bochum, Germany, pp. 77–786.

[27] Owensby E., Shanthakumar A., Rayate V., Namouz E. Z., and Summers J. D., 2011, “Evaluation and Comparison of Two Design for Assembly Methods: Subjectivity of Information,” *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, ASME, Washington, DC, pp. DETC2011–47530.

[28] Widrow B., and Hoff M., 1960, “Adaptive switching circuits.,” *1960 IRE WESCON Conv. Rec.*, (4), pp. 96 – 104.

[29] Hagan M. T., Demuth H. B., and Beale M. H., 1996, “Neural network design,” *Bost. Massachusetts PWS*, **2**, p. 734.

[30] Rumelhart D. E., Hinton G. E., and Williams R. J., 2013, “Learning Internal Representations by Error Propagation,” *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, pp. 399–421.

[31] Sridhar S., Fazelpour M., Gill A., and Summers J. D., 2016, “Precision Analysis of the Graph Complexity Connectivity Method: Assembly and Function Model,” *CIRP CATS 2016*, CIRP, Gothenburg, Sweden, p. 01095.

[32] Phelan K. T., Summers J. D., Pearce B., and Kurz M. E., 2015, “Higher order interactions: Product and configuration study on DSM saturation,” *Int. Conf. Eng. Des.*, (July), pp. 1–10.