Comparison of Methods for Maximising Testing Accuracy of Classification Artificial Neural Networks Given Incomplete Input Data

“To what extent can the methods of network reduction and network imputation improve the testing accuracy of multi-class classification feed-forward artificial neural networks given incomplete input data?”

Computer Science Extended Essay

3999 Words

May 2020

Table of Contents:

Section Ⅰ - Introduction

1.1: Importance and Scope

The advent of machine learning, particularly over the previous two decades, has revolutionised a multitude of fields and accelerated the development of others. Manifesting in a number of forms, with applications ranging from engineering to healthcare, machine learning programs have ubiquitously augmented humans with the ability to identify patterns from data for classification and optimisation uses[1]. One form of machine learning, *feed-forward multi-class classification artificial neural networks* (MCC ANNs) (refer to figure 1.1), and their use in healthcare is of particular interest and research allure to me, having conducted previous independent research on the topic.

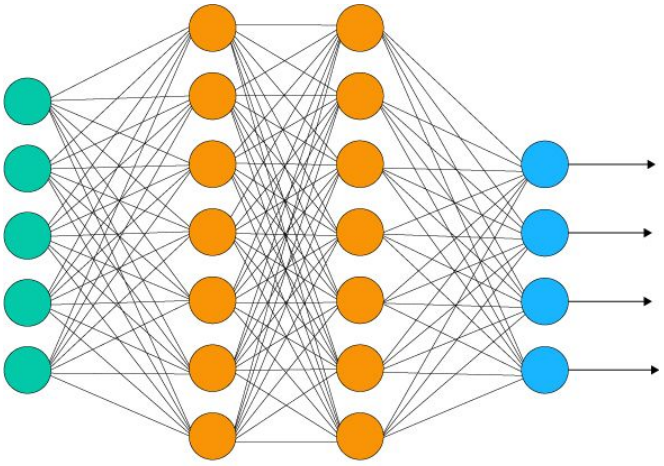

Figure 1.1: Visual representation of 4-layer MCC ANN

Source: https://www.xenonstack.com/blog/artificial-neural-network-applications/

Ironically, data, the metaphorical blood of a neural network, can also be its major downfall. A classification network trained across a dataset of specific dimensions can only be used on data of that particular dimension. Hence, conventional neural networks have an outstandingly poor ability to handle incomplete *testing* input *data instances* and resort to inbuilt *nullification*: assigning a constant, or value of '0', to *features* - individual data within a data instance - that is missing or corrupted[2]. Given the importance and individual meaning of

[1] "Machine Learning: What It Is and Why It Matters." *SAS*, www.sas.com/en_us/insights/analytics/machine-learning.html. Accessed: July 11, 2019

[2] Badr, Will. "6 Different Ways to Compensate for Missing Values In a Dataset (Data Imputation with Examples)." *Medium*, Towards Data Science, 12 Jan. 2019, towardsdatascience.com/6-different-ways-to-compensate-for-missing-values-data-imputation-with-examples-6022d9ca0779. Accessed: July 20, 2019.

each feature of an input data instance, nullification often leads to misclassification and an overall decline in the average classification accuracy of a network - to potentially disastrous effect in the domain of healthcare. A single failure in data acquisition (via a faulty sensor) or data transmission to classification networks in medical software can lead to ramifications including misdiagnoses[3]. It is this major caveat of conventionally poor ability to handle incomplete testing input data and the search for a better handling method, that will form the basis for this investigation.

This investigation will compare the testing accuracy of two increasingly commonplace active handling methods - *network reduction* and *network imputation* - in comparison with the conventional *network nullification* handling method on self-collected incomplete temperature-stress condition data. The scope of this investigation will be limited to testing data instances with only a single missing feature per instance and feed-forward MCC ANNs with a standard four-layer architecture. This ensures the conclusions drawn are valid within a domain and are a result of the handling methods being tested rather than by external unrelated factors. In this, my computer science extended essay, I aim to investigate and answer **'to what extent can the methods of network reduction and network imputation improve the testing accuracy of multi-class classification feed-forward artificial neural networks given incomplete input data?'.**

1.2: Inspiration and Prior Knowledge

My interest in this topic is following extensive research into MCC ANNs for my IB MYP personal project and subsequent independent research project conducted the following summer. Having developed an MCC ANN, sensor array and frontend mobile app capable of monitoring, predicting and preventing temperature-stress conditions in the elderly, I conducted live training and testing of the network with a number of clinics and hospitals abroad, in India.

Of the input features for the temperature-stress prediction MCC ANN, one was to be supplied by a temperature sensor as part of the sensor array. However, on a number of sensor arrays, this sensor failed to report values to the network. As a result, the default network nullification method meant the network defaulted to proceeding the missing features with a value of 0. Subsequently, a number of networks began returning dire risks of

[3] "Tackling the Misdiagnosis Epidemic with Human and Artificial Intelligence." *Healthcare Analytic News*, www.idigitalhealth.com/news/diagnostic-errors-human-and-artificial-intelligence. Accessed: July 20, 2019.

hypothermia for their assigned elders on warm summer days. It is this inability of MCC ANNs with conventional network nullification handling to classify with a high accuracy that this investigation aims to explore the relative performance of two alternatives to.

1.3: Neural Networks Premise and Relevant Terminology

MCC ANNs are a form of artificial neural network wherein the network learns to classify *input instances* - a set of features representing an object or event - into *classes* based on patterns present in data, often indistinguishable by a human[4]. As with all ANNs, to achieve this level of training, the network must be given much data for it to gradually 'learn' from across a number of *training iterations* and *training epochs*. It does so by adjusting the *weight matrices* and *bias matrices* linking layers of *nodes* to minimise overall *loss* - the mathematical justification for classification inaccuracy. Loss is determined by a mathematical function linking the disparity between the predicted class of an instance to the instance's *label* - the correct class.

In most applications of MCC ANNs, including this investigation, training is achieved through *supervised learning* and *gradient descent*. Supervised learning firstly entails splitting a dataset into *training data* and *testing data*. The individual *data instances* in both the training and testing datasets have a number of *features*, attributes of each instance, on whose basis the network must learn to classify the instance. The training of the network itself occurs in two stages, *forward-propagation* and *back-propogation*. *Forward-propagation* is the process by which the network produces classifications for an input *training batch* - a batch of training data instances - and calculates the total loss of the network on the batch. *Back-propagation* entails the network adjusting its weight and bias matrices in a loss-reducing (classification accuracy increasing) direction. A *training iteration* is a single forward and back propagation of the network on a single training batch. A single *training epoch* has completed once all batches in the testing dataset undergo forward and backpropagation of the network once [5].

Upon the completion of every training epoch, the network conducts a *testing epoch*, where the network is tested on all testing data instances. While these instances are still labelled,

---

[4] Marr, Bernard. "10 Amazing Examples Of How Deep Learning AI Is Used In Practice?" *Forbes*, Forbes Magazine, 12 Dec. 2018, www.forbes.com/sites/bernardmarr/2018/08/20/10-amazing-examples-of-how-deep-learning-ai-is-used-in-practice/, Accessed: July 28, 2019.

[5] "KDnuggets." *KDnuggets Analytics Big Data Data Mining and Data Science*, www.kdnuggets.com/2016/10/deep-learning-key-terms-explained.html/2. Accessed: 25 July, 2019

the labels are now only used to calculate the loss of the network, and to identify the percentage of input instances the network is identifying correctly - the network's current *testing accuracy*.

It is here, in these testing datasets, that the oddities of the real world can be simulated. In this investigation, the testing dataset will be filled with incomplete input instances, and the testing accuracy of the network, across testing epochs, will be recorded in order to answer the research question.

1.4: Nullification, Network Induction and Network Imputation

The nullification method of incomplete input instance features simply allows any missing feature of an instance to be assigned a value of zero and scaled to zero in the scaling step before the introduction of its dataset to a network - rendering the instance complete to the classifier[6]. While it is not often programmed into networks, a number of machine learning frameworks including TensorFlow and Keras have nullification as inbuilt exception call protection.

To assist this investigation, a research paper from the University of West England (UWE) was identified. The paper, titled 'Dealing with Missing Values in Neural Network-Based Diagnostic Systems' provided research into network reduction and network imputation (in the paper referred to as network substitution) as used in diagnosing medical scans of patients with incomplete health records[7]. Although not recent, the source is reliable with multiple citations and has been peer-reviewed multiple times. Thus, it was used to understand the handling methods in depth so as to allow their programming.

Network reduction is an approach to handling incomplete input data where the classifier is a system of trained MCC ANNs rather than a single trained classifier. Entitled 'multinet' in the paper from UWE, this system of sub-networks is made of networks of varying input dimensions. Thus, an input instance with any combination of missing features could be directed to a trained network within the multinet with the corresponding input features for its

[6] Ekami. "Best Way to Deal with Missing Values." *Deep Learning Course Forums*, 2 Dec. 2017, forums.fast.ai/t/best-way-to-deal-with-missing-values/8548. Accessed: 25 July, 2019.

[7] Sharpe, P. K., and R. J. Solly. "Dealing with Missing Values in Neural Network-Based Diagnostic Systems." *University of the West of England*, 1995. Accessed: 2 July, 2019.

classification[8]. Unlike nullification or network imputation, this method does not attempt to complete the input instance - but instead treat the incomplete testing instance as a complete instance for a trained classifier of corresponding dimensions.

Network imputation is an entirely different approach to handling incomplete input data. In it, the network aims to predict the value of the missing feature based on the present features of an incomplete data instance. This is accomplished by training a *regression neural network* - a network whose purpose is not to classify an input instance, but rather to produce a predicted value for the missing feature based on the present features of an instance. This predicted feature value can then be inserted in place of the missing value of the instance, and the now complete instance can be ordinarily passed onto the single main classifier[9].

[8] Sharpe, P. K., and R. J. Solly. “Dealing with Missing Values in Neural Network-Based Diagnostic Systems.” *University of the West of England*, 1995. Accessed: 2 July, 2019.

[9] Sharpe, P. K., and R. J. Solly. “Dealing with Missing Values in Neural Network-Based Diagnostic Systems.” *University of the West of England*, 1995. Accessed: 2 July, 2019.

Section Ⅱ - Investigation

2.1: Experiment Justification and Methodology

This investigation will involve a software-based experiment to answer the research question. The aim is to compare the testing accuracy of networks enhanced by the active methods of network reduction and network imputation on incomplete input data to that of a network with conventional nullification handling. Thus, the independent variable of the experiment is the handling method of the network. The dependent variable is subsequently the testing accuracy percentage of the network on a given testing epoch. Testing accuracy is given by the percentage of the testing dataset instances classified correctly from the total number of instances in the testing dataset.

The testing accuracy of each network will be recorded for the first 100 testing epochs of each network. This allows testing accuracy to be plotted against each testing epoch for each network at a number of stages over network training - a visualisation that would allow more detailed conclusions to be drawn for each handling method.

The data that will be used throughout this experiment for the training and testing of the networks will be the temperature stress condition data I collected over my independent research project in India. The data collected entails 13 attributes of subjects who fall within one of three categories of temperature-stress condition: no condition (class 0), heat exhaustion (class 1) and heatstroke (class 2). Figure 2.1 below is an excerpt of the dataset.

Time of Day	Time of Year	Environmental Temperature	Heat Index	Relative Humidity	Sex	Age	Weight	BMI	Body Temperature	Heart Rate	Systolic BP	Daily Water Intake	Stress Condition
10.6	7.0	39.0	107.3	0.40	1	38	48.4	24.0	40.8	166	60	2.8	2
10.6	6.8	37.7	105.4	0.10	0	50	52.4	19.6	38.5	68	120	5.0	2
14.4	4.6	37.7	101.8	0.10	0	64	46.3	21.1	39.8	96	100	5.0	2
9.5	2.6	37.7	95.9	0.10	0	19	42.1	20.8	39.3	70	116	10.5	2
12.5	9.9	37.7	101.7	0.10	0	21	41.9	22.0	39.4	88	130	5.0	2
14.0	12.0	37.7	101.2	0.10	0	52	53.1	18.7	36.4	88	107	1.5	2
11.9	2.3	37.7	122.7	0.10	0	45	45.4	21.5	42.1	76	116	8.5	2
9.2	9.4	37.7	106.3	0.10	1	23	47.6	20.9	37.2	85	140	5.0	2
11.0	7.0	34.3	110.0	0.10	1	27	152.0	22.5	43.0	111	120	4.1	2
17.0	8.6	26.1	103.4	0.24	1	12	45.5	21.1	38.1	168	126	5.0	2
14.7	4.4	33.7	110.8	0.10	0	78	53.4	21.7	39.6	165	117	5.3	2
12.5	10.3	44.2	105.2	0.10	0	78	41.4	21.0	39.1	144	111	1.6	2
16.2	5.7	38.4	110.3	0.10	0	78	47.1	22.9	43.2	98	119	4.7	2
13.1	2.7	35.6	96.2	0.10	0	78	52.5	20.0	39.4	117	114	3.9	2
12.8	3.5	38.2	113.4	0.10	0	78	51.9	18.9	40.0	177	120	1.3	2
13.0	4.3	38.0	106.5	0.10	0	78	52.6	18.7	41.7	153	112	1.7	2
10.4	6.2	34.1	99.5	0.10	0	78	47.5	21.7	40.9	77	119	3.2	2
15.2	7.0	39.4	102.3	0.43	1	67	46.7	19.4	39.3	144	119	1.5	2
11.8	7.0	39.4	102.4	0.43	0	25	43.1	19.3	40.4	108	100	5.6	2
17.0	7.0	39.4	109.9	0.43	1	64	51.4	19.3	39.7	120	110	2.5	2

Figure 2.1: Excerpt of Self-Collected Temperature-Stress Condition Classification Testing Dataset

To conduct this experiment, in Python, using a handful of libraries and frameworks, I will develop three identical network architectures according to the needs of the three handling methods. The 607 instance labelled temperature-stress dataset will then be split into training and testing data in the ratio of 9:1. I will then train each of the networks with the same complete temperature-stress training dataset of 543 instances. Importantly, in the network reduction method, each sub-network will be trained with these same instances, however having removed a single feature per network to match the sub-network's corresponding dimensions.

By removing the values for a single feature of the complete, but unlabelled, testing dataset, for each of its features, I will have created 13 separate testing data files - each file of which has 64 instances and is missing values of the same feature. By supplying each of these modified datasets to the networks to classify over 100 testing epochs and recording its testing accuracy over each epoch, a total of 13 sets of testing accuracy for 100 testing epochs will be recorded for each of the 3 networks - 3900 total data points. This entire process, from training to testing, will then be repeated for five trials for reliability.

Each network's performance will then be processed by taking the average of the best fit lines relating testing accuracy to testing epochs for each of the 13 testing datasets for each of the five trials. This produces a single best fit line of testing accuracy over testing epochs for each handling method that summarises its ability to handle incomplete input data.

By following this methodology, this investigation is capable of reliably answering the research question of 'to what extent can the methods of network reduction and network imputation improve the testing accuracy of multi-class classification feed-forward artificial neural networks given incomplete input data?'.

2.2: Experiment Results

2.2.1: Data Results and Analysis

Figure 2.2 below is an excerpt of the script written for the reduction network.

```
    from keras.models import Sequential
    from keras.layers import Dense

    model = Sequential()

    n_cols = train_X.shape[1]

    #add model layers
    model.add(Dense(12, activation='relu', input_shape=(n_cols,)))
    model.add(Dense(12, activation='relu'))
    model.add(Dense(1))

    model.compile(optimizer='adam', loss='mean_squared_error', metrics=['accuracy'])

    hist = model.fit(train_X, train_y, validation_data=(train_X, train_y), batch_size=64, epochs=100)

    test_y_predictions = model.predict(train_X)
    print(test_y_predictions)

    classSaves = pd.read_csv('TempStressData1CSV.csv')
    classSavedValues = classSaves.iloc[:, 13:14].values
    print(classSavedValues)

    classSavedValuesArray = []
    for i in range(0,len(classSavedValues)):
        classSavedValuesArray.append(classSavedValues[i])

    print(classSavedValuesArray)

    with open(fileIn,'r') as csvinput:
        with open(fileOut, 'w') as csvoutput:
            writer = csv.writer(csvoutput, lineterminator='\n')
            reader = csv.reader(csvinput)

            all = []
            row = next(reader)
            row.append(str(missing) +' Pred')
            # all.append(row)

            row.append('Stress Condition')
            all.append(row)
```

Figure 2.2: Excerpt of Network Reduction Enhanced Network Script

This experiment has produced a number of quantitative and qualitative results to be analysed and evaluated to answer the research question. As it would be infeasible to insert the 3900 dependent variable points collected per network per trial, the average testing accuracy across testing datasets of each testing epoch has been calculated for each network for each trial. These critical values for each network, and the processing of their average and standard deviation across trials, are summarised with figures for 6 epochs in figures 2.3 to 2.5 below. The full table of results for 100 epochs of each method is given in appendices 1 to 3. All values are given to three decimal places to preserve accuracy.

Figure 2.3: Nullification Method - Processed Testing Accuracy per Training Epoch

Network with Nullification Handling - Testing Accuracy per Testing Epoch							
	Testing Accuracy (%)						
Testing Epoch	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**	**Trial 5**	**Average**	**Standard Deviation**
1	25.473	29.256	32.282	32.535	36.192	31.148	4.014
20	36.948	43.632	42.875	42.497	48.045	42.799	3.954
40	20.933	27.491	25.977	25.725	24.464	24.918	2.474
60	20.303	20.555	15.889	21.564	21.438	19.950	2.334
80	18.916	18.411	15.637	17.654	19.042	17.932	1.394
100	16.520	18.285	15.511	18.033	18.916	17.453	1.397

Figure 2.4: Imputation Network - Processed Testing Accuracy per Training Epoch

Network with Imputation Handling - Testing Accuracy per Testing Epoch							
	Testing Accuracy (%)						
Testing Epoch	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**	**Trial 5**	**Average**	**Standard Deviation**
1	39.975	39.849	43.253	33.291	45.271	40.328	4.551
20	80.832	80.202	79.445	80.328	79.571	80.076	0.571
40	83.480	82.976	80.076	82.850	80.454	81.967	1.578
60	82.976	81.967	78.562	82.472	79.823	81.160	1.884
80	82.219	81.337	77.932	82.093	78.689	80.454	2.004
100	81.211	81.084	78.058	81.211	78.562	80.025	1.577

Figure 2.5: Reduction Network - Processed Testing Accuracy per Training Epoch

Network with Reduction Handling - Testing Accuracy per Testing Epoch							
	Testing Accuracy (%)						
Testing Epoch	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**	**Trial 5**	**Average**	**Standard Deviation**
1	39.754	51.776	26.639	48.087	38.251	40.902	9.768
20	86.202	86.202	85.109	84.016	84.016	85.109	1.093
40	88.525	89.617	89.344	90.574	87.158	89.044	1.283
60	89.754	90.847	90.301	91.120	88.661	90.137	0.978
80	90.301	91.393	90.574	92.486	89.891	90.929	1.030
100	90.847	91.803	91.120	92.760	90.574	91.421	0.877

The most prominent observation from these tables is the vast difference between the average testing accuracies of the networks at 100 epochs. The nullification network is, expectedly, the lowest with an average final testing accuracy of 17.453% with a standard deviation across the trials of 1.397 (%). This testing accuracy is incredibly low for any trained network and is far lower than even the random epoch 0 testing accuracy of all three of the networks. This is justified as by epoch 100 the network is trained to rely on a present feature of data that, in testing, has been nullified.

Network imputation has a significantly higher final testing accuracy of 80.025% but too is dwarfed by network reduction's final testing accuracy of 91.421%. For reference, a majority of MCC ANNs trained on complete datasets with four-layer architectures hover around 95% testing accuracy at epoch 100[10]. Additionally, the standard deviation values of the networks indicate a trend of network reduction having a generally lower standard deviation across the trials than the other methods by around 1-2%. The relatively higher standard deviation of network imputation testing accuracy can be inferred as a result of the nature of induction, and its inherent susceptibility to fallacy and factoring nonexistent trends relating present instance features to the one missing.

[10] "State-of-the-Art in Artificial Neural Network Applications: A Survey." *Heliyon*, Elsevier, 23 Nov. 2018, www.sciencedirect.com/science/article/pii/S2405844018332067. Accessed: 1 September, 2019.

2.2.2: Graphed Results and Analysis

Figures 2.6 to 2.8 below are the testing accuracy across testing epochs plots of the first trial of each network. Each plot entails the subsequent individual best fit lines for each of the 13 testing input datasets (each missing a single feature) and the average best fit of these individual best fits. The points of the scatterplot on which the best fit lines are modelled are hidden for clarity and the best fit lines are all quintic equations - defined and plotted by the Python library 'matplotlib'. It is important to note that the starting (epoch 0) accuracy of each testing dataset is entirely random (a property of all neural networks) and is not a function of the handling method in the network.

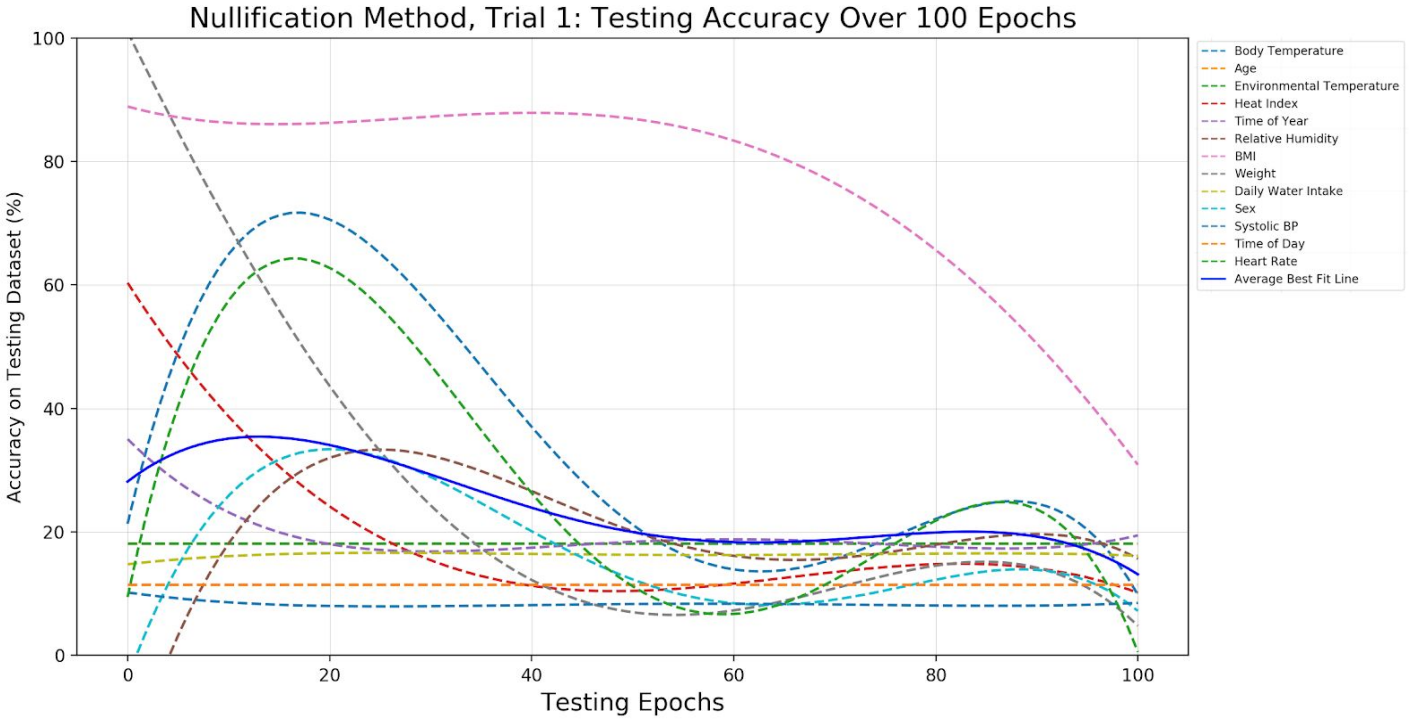

Figure 2.6: Nullification Method Best Fit Lines of Individual Datasets- Trial 1

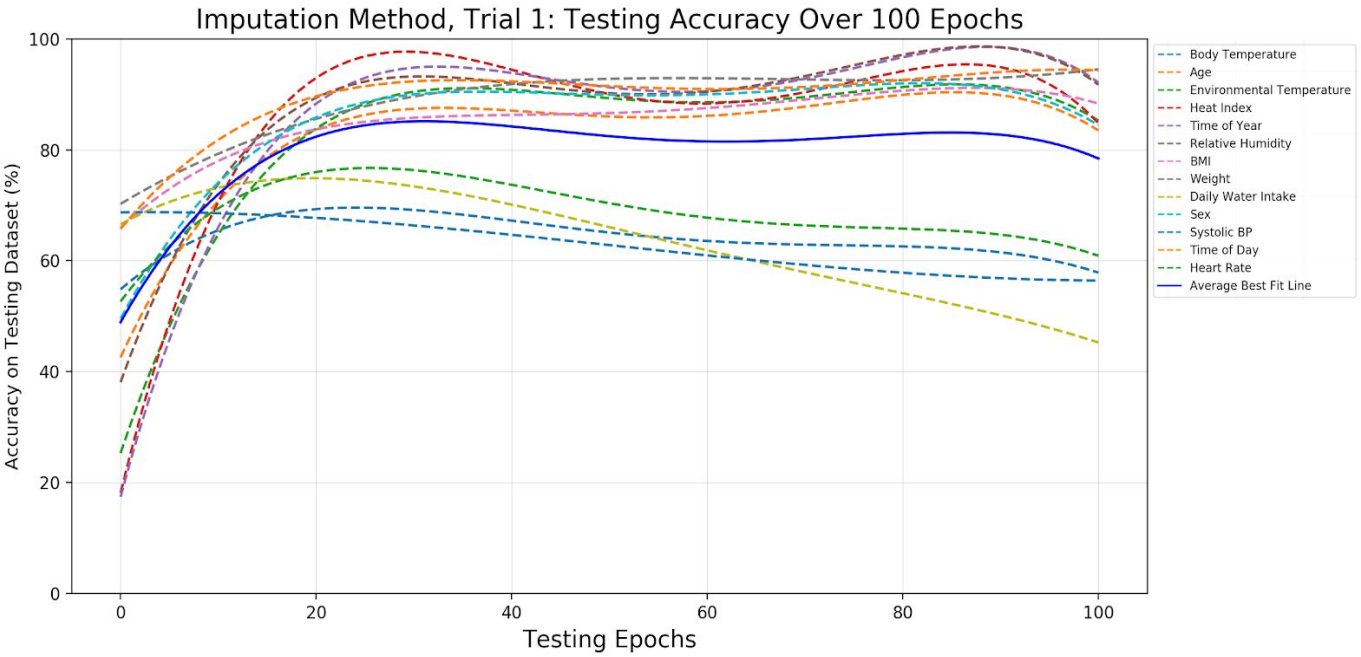

Figure 2.7: Imputation Method Best Fit Lines of Individual Datasets - Trial 1

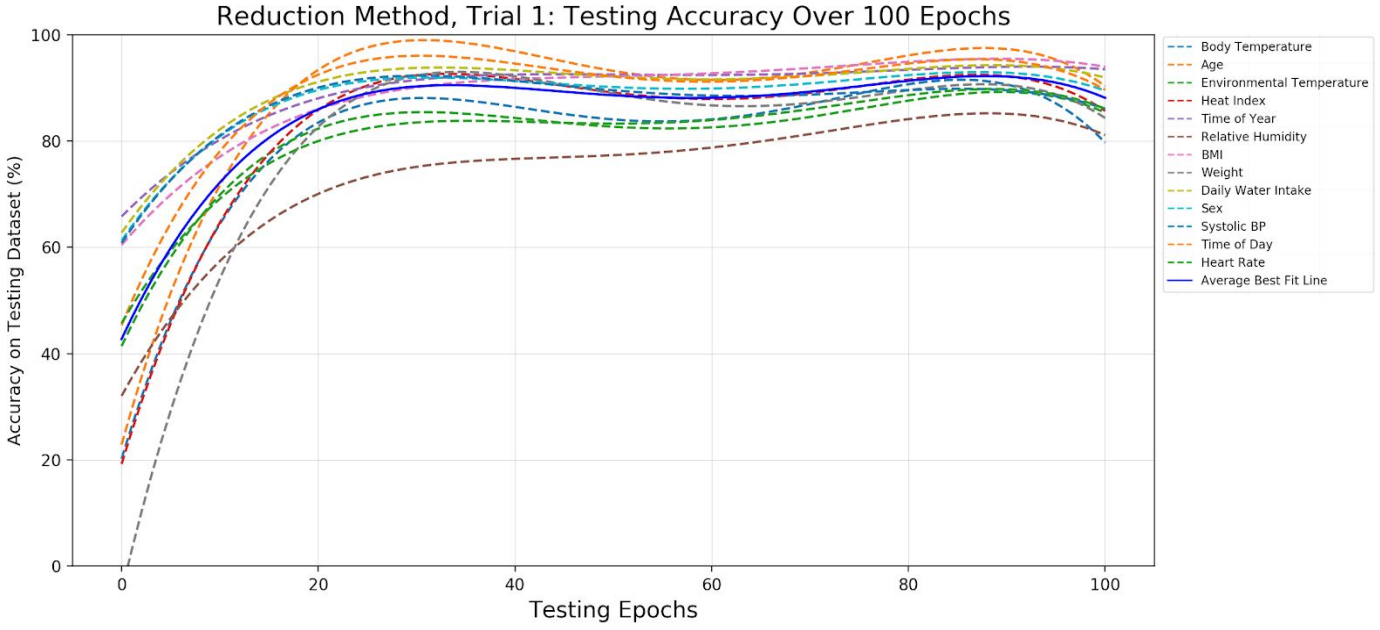

Figure 2.8: Reduction Method Best Fit Lines of Individual Datasets - Trial 1

The most prominent of the observable qualitative differences between nullification and the two active handling methods lay in the inconsistency of the nullification network's best fit lines of individual testing datasets. This is likely a consequence of the fact that nullifying certain features have a greater impact on the network's ability to classify than nullifying other

features. In the case of classifying instances between no condition, heat exhaustion and heatstroke, the nullification method shows networks trained without ‘BMI’ and ‘time of year’ features tested with the highest accuracy percentages of the datasets - although still extremely poorly with ~33% and ~19% respectively - while datasets without ‘environmental temperature’ and ‘relative humidity’ tested worst with 2% and 5% accuracy respectively at epoch 100. This is indicative of the importance of these values to the classification of temperature-stress conditions - that ‘environmental temperature’ and ‘relative humidity’ are highly influential features as opposed to ‘BMI’ and ‘time of year’. This inconsistency is not seen to the same extent in the other method’s graphs - however network imputation follows a somewhat bimodal distribution given the two broad groupings of accuracy values at epoch 100 - with the resultant average best fit line returning centrally. This can be explained by the main caveat of network imputation deduced earlier - weak or misleading correlations between the present features and (individually) each of the four missing and imputed features of the grouping with lower testing accuracy at epoch 100: ‘daily water intake’, ‘body temperature’, ‘systolic BP’ and ‘heart rate’.

Network reduction is clearly the most consistent, however given that it must similarly handle classification without features that would otherwise contribute greatly, the datasets missing ‘relative humidity’ and ‘body temperature’ trained worst. Yet, by not introducing a placeholder value and not imputing a prediction value, the reduction method achieved the highest final testing accuracy and within the smallest range of deviation.

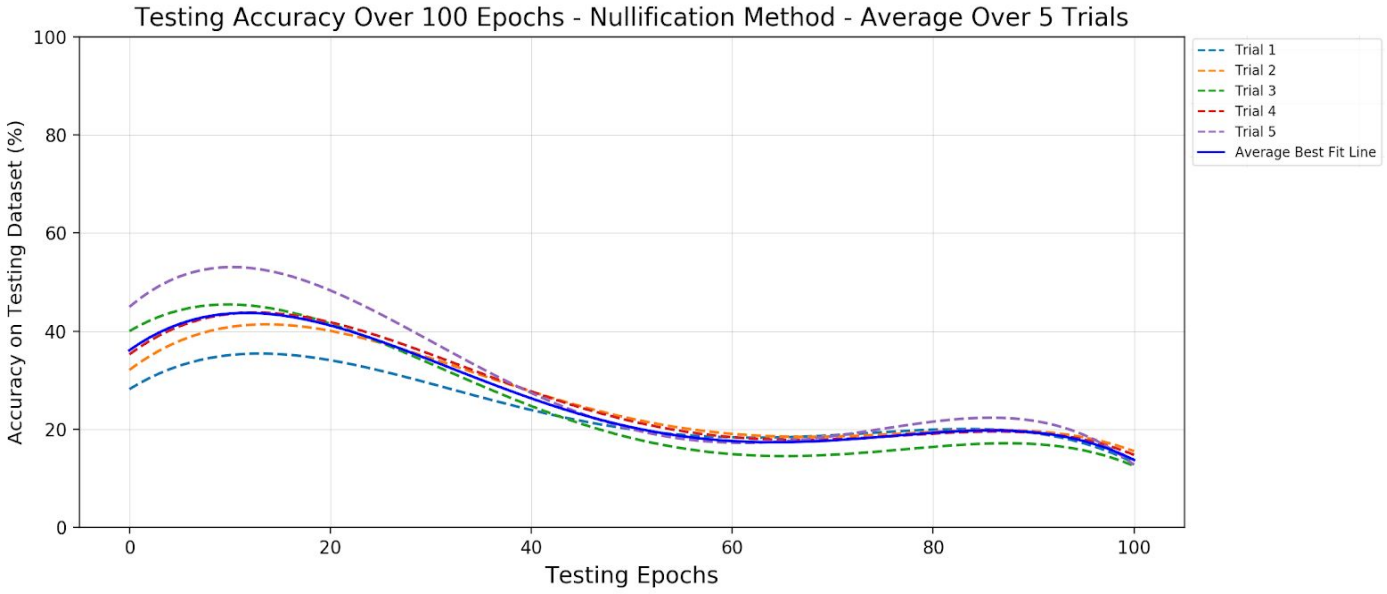

Figure 2.9: Nullification Method - Average of Average Best Fit Lines Across 5 Trials

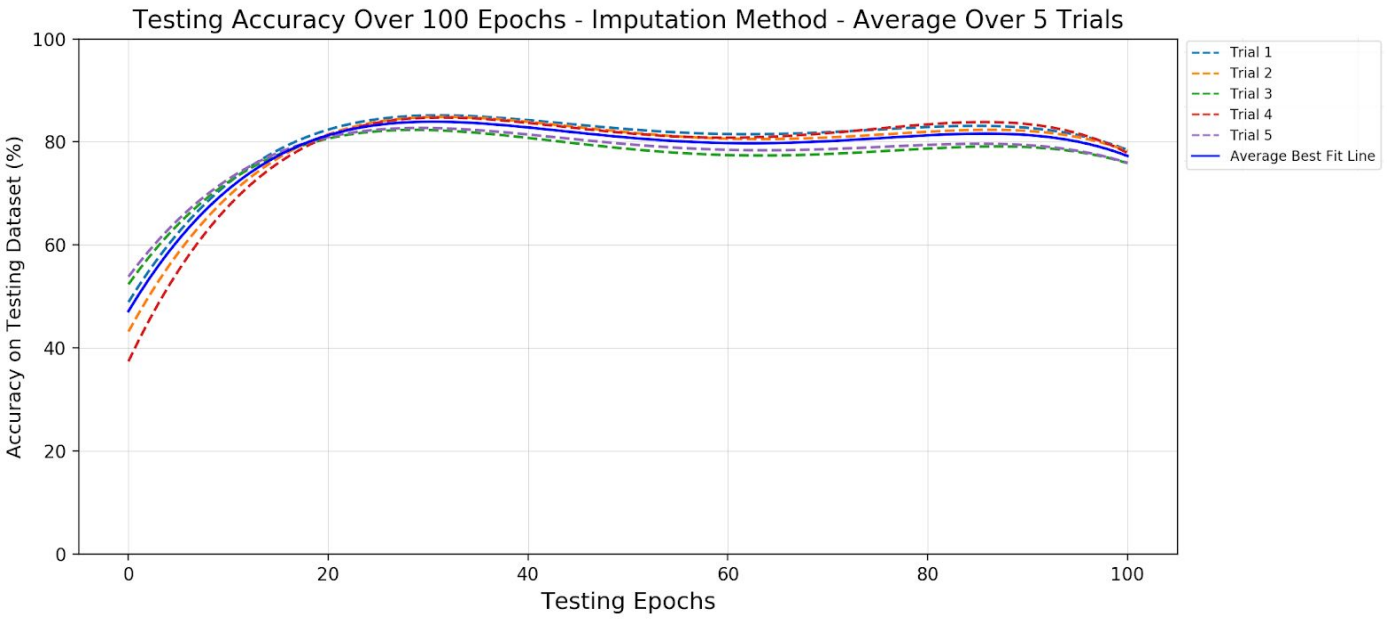

Figure 2.10: Imputation Method - Average of Average Best Fit Lines Across 5 Trials

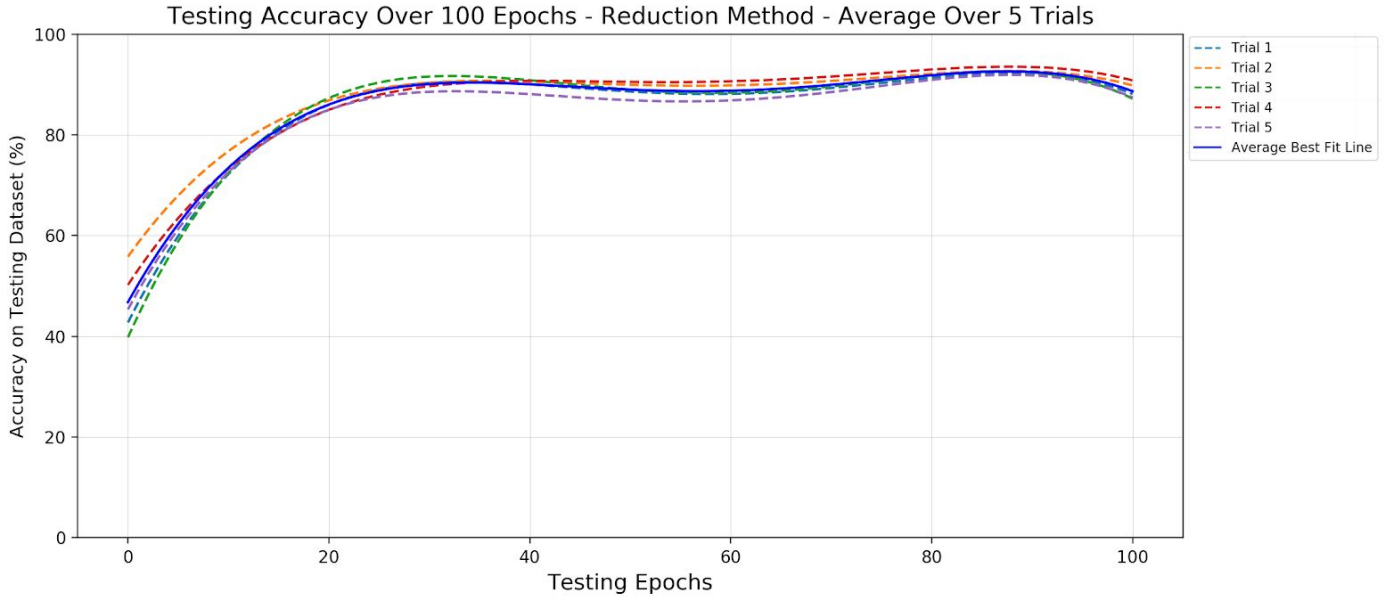

Figure 2.11: Reduction Method - Average of Average Best Fit Lines Across 5 Trials

Figures 2.9 through 2.11 above reinforce the qualitative and quantitative observation made previously and produce further reliable average lines of best fit as the average of the average best fit lines of each network's 5 trials. It demonstrates the relatively higher testing accuracy standard deviation of the nullification network than the other networks and highlights the high testing accuracy of the reduction method.

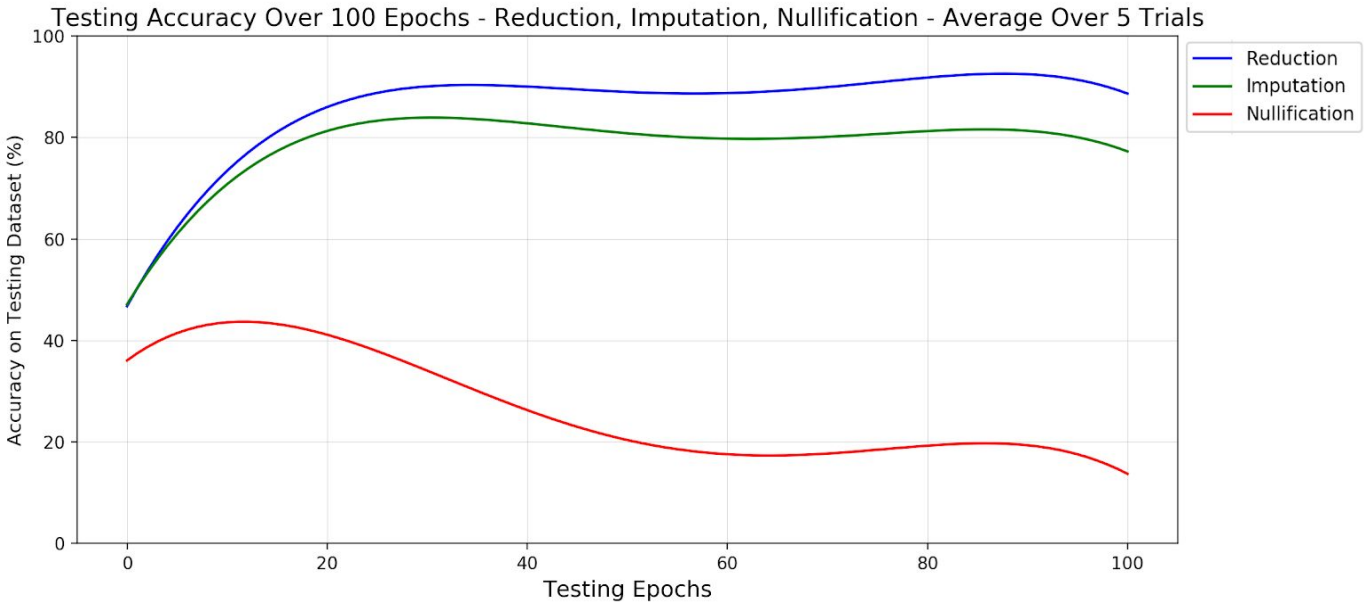

Figure 2.13: Nullification, Imputation, Reduction Methods Average Best Fit Line Comparison

To visually summarise and support all this quantitative and qualitative data, the average testing accuracy over epochs of each network can be placed in a final conclusive graph - figure 2.13 above.

2.2.3: Network-Network Comparison and Analysis

The research question specifies exploring the testing accuracy improvement of an MCC ANN, conventionally on nullification handling, when instead using network imputation and network reduction handling. The table below, figure 2.12, summarises with 6 epochs the percentage increase over the nullification network's testing accuracy that each of the active handling methods were producing. The full table of results for 100 epochs of each method is given in appendix 4.

Figure 2.12: Imputation and Reduction Methods Testing Accuracy Increase to Nullification

	Nullification Method	Imputation Method		Reduction Method	
Testing Epoch	**Testing Accuracy (%)**	**Testing Accuracy (%)**	**Accuracy Increase to Nullification (%)**	**Testing Accuracy (%)**	**Accuracy Increase to Nullification (%)**
1	31.148	40.328	29.474	40.902	31.316
20	42.799	80.076	87.095	85.109	98.856
40	24.918	81.967	228.947	89.044	257.346
60	19.950	81.160	306.827	90.137	351.823
80	17.932	80.454	348.664	90.929	407.079
100	17.453	80.025	358.526	91.421	423.820

Till epoch 100, the trend of network reduction's highest accuracy percentage clearly continues at which point:
The baseline network nullification method has a testing accuracy of 17.453%.

Network imputation has the next highest testing accuracy of 80.025%.
This is a ~359% accuracy increase over nullification at epoch 100.

Network reduction has the highest testing accuracy of 91.421%.
This is a ~424% accuracy increase over nullification at epoch 100.

Section Ⅲ - Conclusion

3.1: Experiment Methodology Limitations and Results Scope

The preceding processed results are conclusive in demonstrating and justifying the major testing accuracy improvements over nullification of the active handling methods. However, there are some intuitive limitations to the universality of the testing accuracy figures and best fit lines identified. Importantly, the investigation results must only be considered within the scope declared at the onset: as applicable only to feed-forward MCC ANNs following a classical four-layer architecture and being input datasets missing a single feature.

The first key limitation, as seen in the results, is the nature of both active handling methods to rely on the correlation between the present features of an instance. If a single feature is solely important to classification, it is possible for a network with an active handling method to classify as inconsistently as one with nullification handling. However, the features of any dataset are recorded with purpose and such a sole determinant is likely rare in a professional healthcare dataset given the interconnectedness of all bodily aspects to medical conditions. To address the extent of this limitation, this investigation could be conducted again on a different dataset - one with different features, but the same dimensions.

The results of the network imputation network is critically also dependant on the accuracy of the regression network used to identify the trends between the present features. This investigation made use of a simple four-layer regression network trained to an accuracy of ~94%. Should a better regression network architecture or embedded algorithm be implemented and increase this accuracy, network imputation may perform better than observed in this investigation.

3.2: Investigation Summary and Findings

This investigation aimed to evaluate methods to reduce the impact that missing or corrupted data can have on the classification ability of a multi-class classification artificial neural network. Inspired by an observation during my past independent research into the use of MCC ANNs in temperature-stress condition monitoring, I looked to compare conventional nullification incomplete input handling to the more complex active handling methods of network imputation and network reduction. By developing scripts in Python of three networks, each enhanced with one of the three methods, training them on my previously-collected complete temperature-stress condition data and finally testing them on multiple datasets with single features removed, I was able to plot and process key figures

and trends to answer the research question: ‘to what extent can the methods of network reduction and network imputation improve the testing accuracy of multi-class classification feed-forward artificial neural networks given incomplete input data?’.

The incomplete input handling method network of nullification has been demonstrated to produce a low calibre of testing accuracy and with large deviations in its testing accuracy on datasets with a single missing feature. This mounted to producing a majorly downward-sloping average best fit line relating testing accuracy to testing epochs. Without further need for justification, the 17.453% accuracy produced by its best-fit line at testing epoch 100 highlights its insufficiency for use in the real-world - especially in fields like healthcare and engineering where correct diagnoses and predictions can have major impacts on human life.

Network imputation proved to be far better at handling incomplete inputs with a testing accuracy at epoch 100 of 80.025%. However, its nature of relying on the correlation of present features in a dataset to predict the missing feature’s value had the consequence of increasing its standard deviation across trials - a reminder of its potentially volatile testing accuracy nature as a function of the dataset being classified.

Network reduction handling ultimately demonstrated the most favourable testing accuracy characteristics. Having a high testing accuracy at epoch 100 of 91.421% and maintaining a considerably higher testing accuracy than the other method-enhanced networks through 100 epochs, reduction handled the incomplete input dataset effectively. Having achieved this final testing accuracy with a low standard deviation of best fit lines between datasets of different missing features, reduction can also be seen as the method by which the uncertainty of a network’s reaction to differing datasets is far lower than that of imputation and nullification.

In conclusion, default network nullification handling performs exceedingly poorly on incomplete input datasets. Network imputation handling significantly enhances this testing accuracy, in this case around 358% by testing epoch 100, although is sporadic depending on the missing feature. Network reduction enhances testing accuracy further, increasing around 423% to nullification in the same epoch, and thus is the best-suited MCC ANN incomplete input handling method. Devoid of nullifying and imputing missing values, four-layer MCC ANN classifiers enhanced with network reduction handling see the most

consistent and highest improvement in testing accuracy of these methods given incomplete input data missing a single feature.

In the context of my healthcare research project in India which inspired this investigation, developing an MCC ANN enhanced with network reduction handling rather than defaulting to nullification would have provided far more accurate temperature-stress risk calculations and alerts given sensor or transmission failure. In a broader context, network reduction handling would greatly benefit testing accuracy anywhere with the possibility of a missing or corrupted data feature.

References:

Badr, Will. “6 Different Ways to Compensate for Missing Values In a Dataset (Data Imputation with Examples).” *Medium*, Towards Data Science, 12 Jan. 2019, towardsdatascience.com/6-different-ways-to-compensate-for-missing-values-data-imputation-with-examples-6022d9ca0779. Accessed: July 20, 2019

Ekami. “Best Way to Deal with Missing Values.” *Deep Learning Course Forums*, 2 Dec. 2017, forums.fast.ai/t/best-way-to-deal-with-missing-values/8548. Accessed: August 25, 2019

“KDnuggets.” *KDnuggets Analytics Big Data Data Mining and Data Science*, www.kdnuggets.com/2016/10/deep-learning-key-terms-explained.html/2. Accessed: August 25, 2019

“Machine Learning: What It Is and Why It Matters.” *SAS*, www.sas.com/en_us/insights/analytics/machine-learning.html. Accessed: July 11, 2019

Marr, Bernard. “10 Amazing Examples Of How Deep Learning AI Is Used In Practice?” *Forbes*, Forbes Magazine, 12 Dec. 2018, www.forbes.com/sites/bernardmarr/2018/08/20/10-amazing-examples-of-how-deep-learning-ai-is-used-in-practice/. Accessed: July 28, 2019.

Sharpe, P. K., and R. J. Solly. “Dealing with Missing Values in Neural Network-Based Diagnostic Systems.” *University of the West of England*, 1995. Accessed: July 2, 2019.

“State-of-the-Art in Artificial Neural Network Applications: A Survey.” *Heliyon*, Elsevier, 23 Nov. 2018, www.sciencedirect.com/science/article/pii/S2405844018332067. Accessed: September 1, 2019.

“Tackling the Misdiagnosis Epidemic with Human and Artificial Intelligence.” *Healthcare Analytic News*, www.idigitalhealth.com/news/diagnostic-errors-human-and-artificial-intelligence. Accessed: July 20, 2019

Appendix:

Appendix 1:

Network with Nullification Handling - Testing Accuracy per Testing Epoch							
	Testing Accuracy (%)						
Testing Epoch	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Average	Standard Deviation
1	25.473	29.256	32.282	32.535	36.192	31.148	4.014
2	25.725	30.391	34.174	34.300	39.723	32.863	5.194
3	27.617	31.652	40.227	35.939	43.632	35.813	6.425
4	32.030	34.805	44.515	36.822	47.163	39.067	6.480
5	34.426	37.327	47.793	39.218	49.433	41.639	6.616
6	36.192	38.966	49.937	41.362	50.946	43.480	6.622
7	38.083	41.740	50.946	42.749	51.576	45.019	5.961
8	37.957	42.749	50.189	43.632	53.720	45.649	6.273
9	37.957	43.632	49.937	45.397	55.233	46.431	6.528
10	37.957	44.010	48.676	47.919	58.638	47.440	7.564
11	37.957	44.136	47.289	48.424	62.421	48.045	9.006
12	36.066	44.262	44.893	48.172	62.926	47.264	9.825
13	33.039	43.758	44.010	48.676	63.808	46.658	11.170
14	32.913	42.749	43.632	48.928	62.926	46.230	10.980
15	32.661	43.506	43.632	47.163	62.169	45.826	10.635
16	32.535	44.893	44.136	46.154	60.025	45.549	9.769
17	32.535	44.767	45.019	44.262	58.260	44.968	9.108
18	32.913	43.884	45.776	43.506	54.603	44.136	7.724
19	34.426	43.632	45.145	43.253	50.820	43.455	5.888
20	36.948	43.632	42.875	42.497	48.045	42.799	3.954
21	36.318	44.262	41.236	41.866	45.776	41.892	3.613
22	37.453	42.245	39.849	41.362	44.388	41.059	2.601
23	37.453	39.344	38.462	40.101	42.623	39.596	1.960
24	37.453	37.579	36.948	38.966	41.992	38.588	2.045

25	36.570	34.805	36.318	38.335	40.858	37.377	2.314
26	35.939	33.165	34.678	37.201	40.353	36.267	2.730
27	34.048	30.895	33.417	36.570	38.335	34.653	2.883
28	31.274	30.391	32.282	35.813	36.318	33.216	2.692
29	29.508	30.013	31.904	34.552	34.426	32.081	2.374
30	27.743	29.256	30.895	33.039	31.778	30.542	2.084
31	25.725	28.752	30.139	31.274	29.760	29.130	2.106
32	24.464	28.373	29.256	29.508	27.869	27.894	2.028
33	23.960	28.373	28.373	29.382	26.230	27.264	2.176
34	22.951	27.995	27.869	29.004	25.977	26.759	2.393
35	22.573	28.121	27.238	28.373	25.725	26.406	2.380
36	22.194	27.995	26.860	27.617	25.347	26.003	2.358
37	21.816	27.995	26.482	26.734	24.968	25.599	2.373
38	21.185	27.617	26.608	26.482	25.095	25.397	2.520
39	20.933	27.491	26.356	25.851	24.590	25.044	2.523
40	20.933	27.491	25.977	25.725	24.464	24.918	2.474
41	20.807	27.491	24.968	25.221	23.960	24.489	2.430
42	20.807	27.238	22.699	24.590	23.707	23.808	2.378
43	20.807	27.112	21.942	23.960	23.707	23.506	2.397
44	20.807	26.482	21.311	23.455	23.203	23.052	2.237
45	20.807	26.230	20.807	23.203	23.329	22.875	2.243
46	20.681	25.095	20.303	23.077	23.077	22.446	1.970
47	20.555	24.716	19.672	22.825	22.951	22.144	2.023
48	20.555	23.707	19.042	23.077	22.825	21.841	1.967
49	20.555	23.329	18.916	22.573	22.699	21.614	1.833
50	20.555	23.077	18.537	22.194	22.320	21.337	1.815
51	20.555	22.951	18.285	22.320	22.194	21.261	1.884
52	20.555	22.825	17.276	22.320	22.320	21.059	2.284
53	20.555	22.194	17.024	21.816	22.194	20.757	2.193

54	20.429	21.942	16.393	21.942	22.068	20.555	2.422
55	20.429	21.311	16.393	21.942	22.194	20.454	2.370
56	20.429	21.185	16.393	21.942	22.068	20.404	2.336
57	20.429	20.933	16.267	21.942	21.816	20.277	2.328
58	20.429	20.807	16.267	21.690	21.564	20.151	2.233
59	20.303	20.555	16.015	21.438	21.438	19.950	2.258
60	20.303	20.555	15.889	21.564	21.438	19.950	2.334
61	20.303	20.303	16.015	21.311	21.311	19.849	2.202
62	20.303	20.177	16.015	20.429	21.059	19.596	2.031
63	20.177	20.050	16.015	19.672	20.429	19.269	1.839
64	20.050	20.050	15.889	19.672	20.177	19.168	1.843
65	20.050	19.924	15.889	19.294	20.050	19.042	1.790
66	20.050	19.420	15.763	19.042	20.177	18.890	1.809
67	19.924	19.420	15.763	19.294	20.177	18.916	1.799
68	19.924	19.420	15.889	19.168	20.177	18.916	1.738
69	19.798	19.168	16.015	18.916	20.177	18.815	1.643
70	19.798	18.916	16.015	19.042	20.050	18.764	1.611
71	19.798	18.916	15.889	18.411	19.924	18.588	1.634
72	19.672	18.916	15.889	18.411	19.546	18.487	1.538
73	19.672	18.663	15.889	18.411	19.546	18.436	1.525
74	19.546	18.663	15.637	18.033	19.546	18.285	1.612
75	19.420	18.663	15.637	18.159	19.420	18.260	1.561
76	19.546	18.663	15.637	18.159	19.042	18.209	1.525
77	19.546	18.663	15.637	18.159	19.168	18.235	1.543
78	19.042	18.411	15.637	17.781	19.042	17.982	1.412
79	19.042	18.411	15.637	17.654	19.042	17.957	1.417
80	18.916	18.411	15.637	17.654	19.042	17.932	1.394
81	18.789	18.411	15.637	17.654	19.168	17.932	1.400
82	18.789	18.411	15.637	17.402	19.168	17.881	1.417

83	18.537	18.411	15.637	17.402	19.168	17.831	1.380
84	18.663	18.411	15.637	17.402	19.042	17.831	1.369
85	18.411	18.411	15.637	17.528	19.042	17.806	1.327
86	18.285	18.411	15.511	17.528	18.916	17.730	1.336
87	18.159	18.411	15.511	17.528	18.916	17.705	1.324
88	17.781	18.411	15.511	17.654	18.916	17.654	1.301
89	17.654	18.411	15.637	18.033	19.042	17.755	1.290
90	17.402	18.411	15.637	18.033	19.168	17.730	1.333
91	17.402	18.411	15.637	18.033	19.042	17.705	1.300
92	17.150	18.411	15.637	18.033	19.168	17.680	1.353
93	17.150	18.411	15.637	18.033	19.168	17.680	1.353
94	16.898	18.411	15.385	18.033	19.168	17.579	1.475
95	16.898	18.411	15.385	18.033	19.168	17.579	1.475
96	16.898	18.285	15.385	18.033	18.916	17.503	1.391
97	16.898	18.285	15.637	18.033	18.789	17.528	1.264
98	16.772	18.285	15.511	18.033	18.789	17.478	1.327
99	16.772	18.285	15.511	18.033	18.789	17.478	1.327
100	16.520	18.285	15.511	18.033	18.916	17.453	1.397

Appendix 2:

Network with Imputation Handling - Testing Accuracy per Testing Epoch							
	Testing Accuracy (%)						
Testing Epoch	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Average	Standard Deviation
1	39.975	39.849	43.253	33.291	45.271	40.328	4.551
2	46.532	45.397	50.694	38.335	51.324	46.456	5.214
3	52.837	48.928	56.747	42.875	56.494	51.576	5.814
4	59.899	53.846	62.673	50.063	62.421	57.781	5.590
5	65.574	56.873	67.591	55.485	67.591	62.623	5.960
6	70.240	61.034	70.744	59.395	71.122	66.507	5.782
7	73.392	65.322	73.392	64.187	73.644	69.987	4.795
8	75.788	68.600	74.653	68.726	75.662	72.686	3.699
9	76.923	72.888	75.788	72.257	77.301	75.032	2.324
10	77.049	75.032	76.166	73.897	77.680	75.965	1.525
11	77.301	77.175	77.049	74.779	78.310	76.923	1.298
12	78.562	78.562	77.427	76.293	79.193	78.008	1.151
13	79.193	79.445	77.932	77.049	79.571	78.638	1.102
14	79.319	79.950	78.436	78.058	80.076	79.168	0.898
15	79.319	80.328	78.562	78.436	80.202	79.369	0.885
16	79.950	80.454	78.941	79.067	80.202	79.723	0.681
17	80.328	80.832	79.193	79.950	79.950	80.050	0.601
18	80.328	80.454	79.193	79.823	79.571	79.874	0.525
19	80.580	80.202	79.319	80.076	79.697	79.975	0.483
20	80.832	80.202	79.445	80.328	79.571	80.076	0.571
21	80.958	80.202	79.950	80.328	79.823	80.252	0.442
22	81.337	80.706	79.950	80.202	79.950	80.429	0.594
23	82.219	80.706	80.202	80.328	80.328	80.757	0.839
24	82.093	80.958	80.202	80.580	80.202	80.807	0.784
25	82.472	81.463	80.454	80.958	80.580	81.185	0.819

26	82.346	81.967	80.706	81.211	80.202	81.286	0.881
27	82.598	82.219	80.454	81.337	80.076	81.337	1.088
28	82.850	82.219	80.454	81.715	80.202	81.488	1.136
29	82.850	82.472	80.454	82.219	80.202	81.639	1.221
30	83.102	82.219	80.328	82.472	80.076	81.639	1.354
31	83.354	82.093	80.454	82.472	80.202	81.715	1.349
32	83.607	82.219	80.076	82.472	80.202	81.715	1.532
33	83.607	82.219	79.950	82.724	80.202	81.740	1.601
34	83.607	82.472	79.571	82.724	80.454	81.765	1.684
35	83.354	82.850	79.697	82.850	80.328	81.816	1.674
36	83.480	82.976	80.076	82.976	80.580	82.018	1.566
37	83.480	82.976	80.076	82.976	80.706	82.043	1.538
38	83.228	83.102	80.328	83.102	80.454	82.043	1.510
39	83.354	82.976	80.202	82.850	80.328	81.942	1.543
40	83.480	82.976	80.076	82.850	80.454	81.967	1.578
41	83.354	83.228	80.202	82.724	80.706	82.043	1.480
42	83.607	83.354	80.328	82.598	80.454	82.068	1.576
43	83.480	83.354	80.328	82.472	80.580	82.043	1.504
44	83.354	83.228	80.202	82.472	80.454	81.942	1.514
45	83.354	83.102	80.328	82.598	80.832	82.043	1.374
46	83.354	83.102	80.328	82.724	80.832	82.068	1.388
47	83.480	82.850	80.328	82.724	81.211	82.119	1.303
48	83.480	82.850	79.950	82.850	81.084	82.043	1.472
49	83.733	82.976	79.697	82.598	80.706	81.942	1.679
50	83.607	82.976	79.571	82.472	80.958	81.917	1.636
51	83.480	83.102	79.571	82.346	80.832	81.866	1.635
52	83.480	82.850	79.193	81.967	80.706	81.639	1.719
53	83.228	82.724	79.193	81.841	80.832	81.564	1.608
54	83.102	82.598	78.815	82.093	80.328	81.387	1.778

55	82.976	82.472	78.941	82.093	80.076	81.311	1.723
56	82.850	82.472	78.815	82.219	80.202	81.311	1.732
57	82.850	82.346	78.689	82.346	80.202	81.286	1.776
58	82.850	82.472	78.689	82.472	80.202	81.337	1.814
59	82.976	82.093	78.562	82.850	79.823	81.261	1.969
60	82.976	81.967	78.562	82.472	79.823	81.160	1.884
61	82.976	82.093	78.689	82.472	79.823	81.211	1.855
62	82.976	81.715	78.689	82.093	79.571	81.009	1.802
63	82.976	81.967	78.689	82.093	79.319	81.009	1.884
64	82.976	81.841	78.689	82.472	79.445	81.084	1.904
65	82.850	81.967	78.310	82.724	79.193	81.009	2.111
66	82.850	81.463	78.184	82.976	79.067	80.908	2.189
67	82.598	81.337	77.932	83.102	79.193	80.832	2.215
68	82.598	81.084	77.932	83.228	79.319	80.832	2.215
69	82.346	81.211	77.932	83.228	79.193	80.782	2.194
70	82.346	81.337	77.932	82.976	79.193	80.757	2.134
71	82.219	81.084	77.932	82.724	79.193	80.631	2.029
72	82.093	81.211	78.058	82.472	78.941	80.555	1.956
73	82.093	81.084	78.184	82.598	79.067	80.605	1.913
74	82.093	81.084	78.058	82.598	79.067	80.580	1.954
75	81.967	81.211	77.932	82.598	78.941	80.530	2.005
76	82.093	81.084	77.680	82.724	78.689	80.454	2.182
77	82.219	81.211	77.806	82.598	78.941	80.555	2.094
78	82.219	81.084	77.806	82.346	78.689	80.429	2.075
79	82.219	81.211	77.806	82.093	78.689	80.404	2.031
80	82.219	81.337	77.932	82.093	78.689	80.454	2.004
81	82.219	81.337	77.932	82.093	78.689	80.454	2.004
82	81.967	81.211	77.806	81.841	78.562	80.277	1.951
83	81.715	81.084	77.932	81.967	78.436	80.227	1.901

84	81.589	81.211	77.932	81.841	78.184	80.151	1.926
85	81.715	81.084	77.932	81.841	78.058	80.126	1.967
86	81.463	80.958	77.932	81.967	78.058	80.076	1.933
87	81.589	80.706	77.932	81.715	77.806	79.950	1.939
88	81.715	80.832	77.932	81.967	77.932	80.076	2.002
89	81.589	80.706	78.058	82.093	77.806	80.050	1.999
90	81.589	80.832	78.184	81.841	77.932	80.076	1.881
91	81.463	80.706	78.184	81.967	77.806	80.025	1.912
92	81.463	81.084	78.184	81.841	78.058	80.126	1.850
93	81.337	80.832	78.058	81.967	78.310	80.101	1.798
94	81.211	80.832	78.310	81.841	78.562	80.151	1.609
95	81.337	80.958	78.058	81.841	78.436	80.126	1.749
96	81.337	81.084	78.184	81.463	78.310	80.076	1.675
97	81.337	80.832	78.058	82.093	78.436	80.151	1.800
98	81.211	80.832	78.058	81.337	78.815	80.050	1.509
99	81.211	81.084	77.932	80.958	78.689	79.975	1.545
100	81.211	81.084	78.058	81.211	78.562	80.025	1.577

Appendix 3:

Network with Reduction Handling - Testing Accuracy per Testing Epoch							
	Testing Accuracy (%)						
Testing Epoch	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Average	Standard Deviation
1	39.754	51.776	26.639	48.087	38.251	40.902	9.768
2	44.536	57.240	37.432	52.732	45.355	47.459	7.697
3	49.727	61.339	45.355	56.011	51.913	52.869	6.100
4	56.011	65.027	55.328	60.792	57.240	58.880	4.031
5	61.066	68.443	63.115	64.617	62.842	64.016	2.777
6	64.891	71.995	68.169	67.350	67.350	67.951	2.573
7	68.579	75.273	73.224	69.399	71.585	71.612	2.743
8	71.448	77.322	77.459	71.995	73.907	74.426	2.856
9	74.044	78.689	79.235	74.180	76.503	76.530	2.433
10	75.820	80.055	80.874	75.820	78.005	78.115	2.341
11	77.732	80.738	81.148	77.049	79.098	79.153	1.798
12	78.962	81.831	81.694	78.689	80.601	80.355	1.479
13	80.601	82.514	82.377	79.918	80.874	81.257	1.141
14	81.694	83.607	83.333	81.011	81.284	82.186	1.201
15	82.377	84.153	84.016	81.148	82.240	82.787	1.278
16	83.060	84.699	84.153	82.787	82.514	83.443	0.939
17	84.290	85.246	84.153	83.333	83.197	84.044	0.828
18	84.973	85.656	84.290	83.470	83.333	84.344	0.988
19	85.246	85.929	84.699	83.743	83.743	84.672	0.953
20	86.202	86.202	85.109	84.016	84.016	85.109	1.093
21	86.749	86.612	85.792	84.973	84.153	85.656	1.101
22	87.432	86.749	86.066	86.202	84.290	86.148	1.170
23	87.295	87.022	86.339	86.202	84.699	86.311	1.010
24	87.568	87.432	87.158	86.339	85.246	86.749	0.966
25	87.842	87.842	87.432	86.475	85.519	87.022	1.009

26	87.842	87.842	87.842	86.749	85.656	87.186	0.978
27	87.978	88.251	87.705	87.432	85.656	87.404	1.024
28	87.978	88.525	87.978	87.705	85.929	87.623	0.993
29	87.978	89.071	88.251	88.115	86.066	87.896	1.108
30	88.115	89.071	88.661	88.388	86.202	88.087	1.112
31	88.251	89.344	88.798	88.525	86.475	88.279	1.086
32	88.388	89.617	88.934	88.798	86.612	88.470	1.129
33	88.388	89.617	89.208	88.798	86.749	88.552	1.107
34	88.388	89.754	89.208	89.208	86.749	88.661	1.175
35	88.388	89.754	89.208	89.208	86.885	88.689	1.120
36	88.388	89.891	89.208	89.481	86.885	88.770	1.189
37	88.388	89.891	89.208	90.164	87.022	88.934	1.271
38	88.388	89.891	89.208	90.301	87.022	88.962	1.305
39	88.388	89.754	89.208	90.574	87.022	88.989	1.358
40	88.525	89.617	89.344	90.574	87.158	89.044	1.283
41	88.661	89.891	89.481	90.710	87.158	89.180	1.350
42	88.661	89.891	89.754	90.710	87.295	89.262	1.320
43	88.525	90.164	90.164	90.710	87.432	89.399	1.372
44	88.661	90.164	90.164	90.574	87.568	89.426	1.269
45	88.525	90.027	90.437	90.574	87.705	89.454	1.272
46	89.481	90.164	90.437	90.574	87.568	89.645	1.235
47	89.344	90.301	90.710	90.710	87.568	89.727	1.329
48	89.481	90.437	90.574	90.710	87.705	89.781	1.257
49	89.754	90.437	90.574	90.847	87.705	89.863	1.272
50	89.754	90.574	90.574	90.984	87.705	89.918	1.315
51	89.481	90.574	90.574	90.984	87.705	89.863	1.329
52	89.617	90.574	90.574	91.120	87.705	89.918	1.350
53	89.891	90.710	90.574	91.120	87.705	90.000	1.357
54	89.891	90.710	90.437	91.120	87.705	89.973	1.344

55	89.754	90.710	90.437	91.120	87.978	90.000	1.235
56	89.754	90.710	90.437	90.984	88.115	90.000	1.149
57	89.617	90.710	90.437	90.984	88.251	90.000	1.103
58	89.891	90.710	90.301	90.984	88.388	90.055	1.020
59	89.891	90.710	90.301	90.984	88.661	90.109	0.909
60	89.754	90.847	90.301	91.120	88.661	90.137	0.978
61	89.891	90.847	90.301	91.393	88.525	90.191	1.090
62	89.754	90.847	90.437	91.530	88.388	90.191	1.196
63	89.754	90.847	90.437	91.530	88.388	90.191	1.196
64	89.617	90.984	90.437	91.667	88.798	90.301	1.127
65	89.754	90.984	90.301	91.667	88.798	90.301	1.106
66	89.754	90.984	90.301	91.803	88.798	90.328	1.149
67	89.617	90.984	90.301	92.077	88.934	90.383	1.217
68	89.754	90.984	90.301	92.077	88.934	90.410	1.196
69	89.891	90.984	90.437	92.077	88.934	90.464	1.177
70	89.891	90.984	90.437	92.077	89.208	90.519	1.091
71	89.891	91.120	90.301	91.940	89.208	90.492	1.065
72	90.027	91.120	90.301	92.213	89.344	90.601	1.103
73	90.301	91.393	90.301	92.350	89.344	90.738	1.157
74	90.164	91.393	90.437	92.350	89.344	90.738	1.161
75	90.301	91.393	90.437	92.350	89.617	90.820	1.064
76	90.301	91.393	90.437	92.350	89.617	90.820	1.064
77	90.164	91.393	90.437	92.486	89.754	90.847	1.097
78	90.164	91.393	90.437	92.486	89.891	90.874	1.064
79	90.164	91.393	90.574	92.486	89.754	90.874	1.086
80	90.301	91.393	90.574	92.486	89.891	90.929	1.030
81	90.301	91.530	90.437	92.486	90.027	90.956	1.029
82	90.574	91.667	90.301	92.486	90.027	91.011	1.033
83	90.574	91.667	90.301	92.623	90.301	91.093	1.024

84	90.574	91.667	90.574	92.623	90.437	91.175	0.949
85	90.574	91.667	90.710	92.623	90.437	91.202	0.930
86	90.437	91.530	90.574	92.623	90.574	91.148	0.934
87	90.574	91.803	90.574	92.623	90.301	91.175	0.997
88	90.847	91.803	90.847	92.760	90.301	91.311	0.974
89	90.984	91.530	90.710	92.623	90.164	91.202	0.935
90	90.984	91.667	90.710	92.486	90.301	91.230	0.861
91	90.984	91.667	90.710	92.486	90.301	91.230	0.861
92	90.984	91.803	90.574	92.486	90.437	91.257	0.869
93	90.984	91.803	90.710	92.486	90.437	91.284	0.844
94	90.984	91.940	90.847	92.486	90.301	91.311	0.883
95	90.984	91.803	90.847	92.623	90.710	91.393	0.808
96	90.984	91.803	90.847	92.623	90.574	91.366	0.839
97	90.984	91.530	90.847	92.623	90.574	91.311	0.812
98	90.984	91.667	91.120	92.623	90.574	91.393	0.791
99	90.984	91.803	91.120	92.760	90.437	91.421	0.893
100	90.847	91.803	91.120	92.760	90.574	91.421	0.877

Appendix 4:

	Nullification Method	Imputation Method		Reduction Method	
Testing Epoch	**Testing Accuracy (%)**	**Testing Accuracy (%)**	**Accuracy Increase to Nullification (%)**	**Testing Accuracy (%)**	**Accuracy Increase to Nullification (%)**
1	31.148	40.328	29.474	40.902	31.316
2	32.863	46.456	41.366	47.459	44.417
3	35.813	51.576	44.014	52.869	47.623
4	39.067	57.781	47.902	58.880	50.716
5	41.639	62.623	50.394	64.016	53.740
6	43.480	66.507	52.958	67.951	56.279
7	45.019	69.987	55.462	71.612	59.071
8	45.649	72.686	59.227	74.426	63.039
9	46.431	75.032	61.597	76.530	64.824
10	47.440	75.965	60.128	78.115	64.660
11	48.045	76.923	60.105	79.153	64.746
12	47.264	78.008	65.048	80.355	70.015
13	46.658	78.638	68.541	81.257	74.153
14	46.230	79.168	71.249	82.186	77.778
15	45.826	79.369	73.198	82.787	80.655
16	45.549	79.723	75.028	83.443	83.195
17	44.968	80.050	78.015	84.044	86.895
18	44.136	79.874	80.971	84.344	91.100
19	43.455	79.975	84.039	84.672	94.849
20	42.799	80.076	87.095	85.109	98.856
21	41.892	80.252	91.571	85.656	104.470
22	41.059	80.429	95.885	86.148	109.813
23	39.596	80.757	103.949	86.311	117.978
24	38.588	80.807	109.412	86.749	124.809
25	37.377	81.185	117.206	87.022	132.822

26	36.267	81.286	124.131	87.186	140.398
27	34.653	81.337	134.716	87.404	152.226
28	33.216	81.488	145.330	87.623	163.800
29	32.081	81.639	154.481	87.896	173.985
30	30.542	81.639	167.300	88.087	188.412
31	29.130	81.715	180.519	88.279	203.052
32	27.894	81.715	192.948	88.470	217.164
33	27.264	81.740	199.815	88.552	224.800
34	26.759	81.765	205.561	88.661	231.331
35	26.406	81.816	209.838	88.689	235.864
36	26.003	82.018	215.422	88.770	241.392
37	25.599	82.043	220.493	88.934	247.414
38	25.397	82.043	223.039	88.962	250.281
39	25.044	81.942	227.190	88.989	255.329
40	24.918	81.967	228.947	89.044	257.346
41	24.489	82.043	235.015	89.180	264.161
42	23.808	82.068	244.703	89.262	274.921
43	23.506	82.043	249.034	89.399	280.329
44	23.052	81.942	255.470	89.426	287.938
45	22.875	82.043	258.655	89.454	291.051
46	22.446	82.068	265.618	89.645	299.373
47	22.144	82.119	270.843	89.727	305.201
48	21.841	82.043	275.635	89.781	311.066
49	21.614	81.942	279.113	89.863	315.762
50	21.337	81.917	283.924	89.918	321.424
51	21.261	81.866	285.053	89.863	322.667
52	21.059	81.639	287.665	89.918	326.976
53	20.757	81.564	292.953	90.000	333.597
54	20.555	81.387	295.951	89.973	337.720

55	20.454	81.311	297.534	90.000	340.012
56	20.404	81.311	298.517	90.000	341.100
57	20.277	81.286	300.871	90.000	343.843
58	20.151	81.337	303.630	90.055	346.892
59	19.950	81.261	307.332	90.109	351.686
60	19.950	81.160	306.827	90.137	351.823
61	19.849	81.211	309.149	90.191	354.394
62	19.596	81.009	313.385	90.191	360.242
63	19.269	81.009	320.419	90.191	368.074
64	19.168	81.084	323.026	90.301	371.107
65	19.042	81.009	325.430	90.301	374.227
66	18.890	80.908	328.304	90.328	378.171
67	18.916	80.832	327.333	90.383	377.822
68	18.916	80.832	327.333	90.410	377.967
69	18.815	80.782	329.357	90.464	380.820
70	18.764	80.757	330.376	90.519	382.404
71	18.588	80.631	333.786	90.492	386.839
72	18.487	80.555	335.744	90.601	390.086
73	18.436	80.605	337.209	90.738	392.168
74	18.285	80.580	340.690	90.738	396.241
75	18.260	80.530	341.022	90.820	397.376
76	18.209	80.454	341.828	90.820	398.753
77	18.235	80.555	341.770	90.847	398.213
78	17.982	80.429	347.265	90.874	405.353
79	17.957	80.404	347.753	90.874	406.063
80	17.932	80.454	348.664	90.929	407.079
81	17.932	80.454	348.664	90.956	407.232
82	17.881	80.277	348.942	91.011	408.968
83	17.831	80.227	349.929	91.093	410.868

84	17.831	80.151	349.505	91.175	411.327
85	17.806	80.126	350.000	91.202	412.205
86	17.730	80.076	351.636	91.148	414.083
87	17.705	79.950	351.567	91.175	414.969
88	17.654	80.076	353.571	91.311	417.214
89	17.755	80.050	350.852	91.202	413.660
90	17.730	80.076	351.636	91.230	414.545
91	17.705	80.025	351.994	91.230	415.278
92	17.680	80.126	353.210	91.257	416.167
93	17.680	80.101	353.067	91.284	416.322
94	17.579	80.151	355.954	91.311	419.440
95	17.579	80.126	355.811	91.393	419.907
96	17.503	80.076	357.493	91.366	421.998
97	17.528	80.151	357.266	91.311	420.935
98	17.478	80.050	358.009	91.393	422.908
99	17.478	79.975	357.576	91.421	423.064
100	17.453	80.025	358.526	91.421	423.820