5-2018

Discriminative Feature Extraction of Time-Series Data to Improve Temporal Pattern Detection using Classification Algorithms

David Stolze
dws4077@rit.edu

Discriminative Feature Extraction of Time-Series Data to Improve Temporal Pattern Detection using Classification Algorithms

by

David Stolze

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Science in Industrial and Systems Engineering

Department of Industrial and Systems Engineering
Kate Gleason College of Engineering

Rochester Institute of Technology
Rochester, NY
May, 2018

DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING

KATE GLEASON COLLEGE OF ENGINEERING

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NY

M.S. DEGREE THESIS

The M.S. Degree thesis of David Stolze has been examined and approved by the thesis committee as satisfactory for the thesis requirements for the Master of Science Degree

Approved by:

Dr. Katie McConky, Thesis Advisor Date

Dr. Michael E. Kuhl Date

Abstract

Time-series data streams often contain predictive value in the form of unique patterns. While these patterns may be used as leading indicators for event prediction, a lack of prior knowledge of pattern shape and irregularities can render traditional forecasting methods ineffective. The research in this thesis tested a means of predetermining the most effective combination of transformations to be applied to time-series data when training a classifier to predict whether an event will occur at a given time. The transformations tested on provided data streams included subsetting of the data, aggregation over various numbers of data points, testing of different predictive lead times, and converting the data set into a binary set of values. The benefit of the transformations is to reduce the data used for training down to only the most useful pattern containing points and clarify the predictive pattern contained in the set. In addition, the transformations tested significantly reduce the number of features used for classifier training through subsetting and aggregation. The performance benefit of the transformations was tested through creating a series of daily positive/negative event predictions over the span of a test set derived from each provided data stream. A landmarking system was then developed that utilizes the prior results obtained by the system to predetermine a "best fit" transformation to use on a new, untested data stream. Results indicate that the proposed set of transformations consistently result in improved classifier performance over the use of untransformed data values. Landmarking system testing shows that the use of prior knowledge results in selection of a near best fit transformation when using as few as 3 reference transformations.

Acknowledgements

This research is supported by the Office of the Director of National Intelligence (ODNI) and the Intelligence Advanced Research Projects Activity (IARPA) via the Air Force Research Laboratory (AFRL) contract number FA875016C0114. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ODNI, IARPA, AFRL, or the U.S. Government.

I cannot overstate the contribution that my primary advisor Dr. Katie McConky and committee member Dr. Michael E. Kuhl made to the research conducted in this thesis. Their advice and guidance were key in both conducting an effective experiment and helping me become a better data scientist and student.

I would also like to acknowledge the help and guidance of Dr. Shanchieh Jay Yang, whose leadership on the CAUSE team was invaluable in the completion of this research project.

Table of Contents

List of Tables

List of Figures

1. Introduction

Cyber-attacks have been the cause of massive loss of capital and damage to vital infrastructure in both the private and public sectors over the last few years (Bronk & Tikk-Ringas, 2013;Lee, Assante, & Conway, 2014). A large scale cyber-attack has the potential to cost the United States government upwards of $121 billion dollars, more than the total damages of Hurricane Katrina (Maynard & Ng, 2017). The rise of the "Internet of Things" has created wireless connections and access between systems that have never before been remotely tethered. These vast networks can result in very real risks to connected individuals (Roman, Zhou, & Lopez, 2013). In late-2016, malicious software known as Mirai was able to take down multiple major sites such as Twitter and PayPal by utilizing the computing power of millions of vulnerable devices connected through the "Internet of Things" (Kolias et al., 2017). Although these connections are created for ease-of-use and streamlining purposes, they drastically increase the risk of a devastating infiltration.

If businesses could see these types of attacks coming before they occurred, even without absolute certainty of the time and place, they may drastically reduce their losses incurred. Forewarning would allow for the focusing of security efforts and heightened defenses around a given network during the expected window of the attack. While current approaches rely on network sensor activity to detect cyber-attacks, the method is limited by the origin of the data being utilized. There must already be unwanted access of or attempts on the network for these sensors to generate readings, meaning damage may have already been done by the time detection occurs.

The future of cyber-defense may lie in providing warning before an attempt on a system is even made. Research has been conducted regarding the use of traditional forecasting methods such as multi-correlation and ARIMA models in predicting the number of cyber-attacks that may occur in a given time period (Pontes et al., 2011;Werner, Yang, & McConky, 2017). Forecasting models utilize the rates at which cyber-attacks have transpired and attempt to decipher patterns or trends that have predictive value moving forward. These models function by exploiting autocorrelations in the data set, which may make generating predictions difficult for events which are very sparse or follow no detectable trend or seasonality. Instead of trying to find a pattern within the timing of the events themselves, more benefit may be found in examining external data, such as public outrage or availability of malicious software tools, suspected of providing evidence or latent indicators of attacks (Jordan, 2001;Ashford, 2012). Outside data sets which may contain predictive value are known as leading indicators. The greatest difficulty lies in determining which data streams function as leading indicators for cyber-attacks.

There has never been more data readily available to researchers than there is now. The types of data available range from historical weather and environmental data to counts of traffic violations issued in a geographical region (NOAA, 2017;MCoM, 2015). One major source of data which appeared in 2006 and has grown rapidly ever since is Twitter. As of 2016, more than 500 million tweets were created each day, with the number increasing every year (InternetLiveStats, 2017). Numerous analytical techniques have been employed, both traditional and novel in their methods, in attempts to make use of the wealth of available data. Tools to derive information such as public sentiment and trending topics amongst populations from Twitter usage data have become popular amongst business analysts (Culnan, McHugh, & Zubillaga, 2010). The value gained from analysis

of Twitter data has been promising in some fields, such as social sciences and business, while lacking in others, such as cyber security (Lerman & Ghosh, 2010;Chae, 2015). The promise of valuable insight continues to drive research utilizing Twitter as well as other data sources for predictive purposes in the realm of cyber-defense. The study in this thesis aims to approach the predictive problem through the application of a novel method which may be used to extract predictive patterns from many of the data sources mentioned.

Even though an individual or organization may have access to large amounts of data, how best to derive useful information from a given data set is not always clear. Transforming a temporal data stream containing millions of entries into a feature set which may be used to train a machine learning algorithm involves numerous challenges. The difficulties faced include questions such as how much historical data to incorporate, which classification algorithm to use, and how these factors may influence the time required to train a desired algorithm. Simply training on all available data is almost never the best approach and is often not possible due to constraints on run-time and processing power. In the case of a time-series data set, many features of the data beyond just the raw values may be used, such as trends and statistical characteristics (Meina et al., 2015). The "Curse of Dimensionality" often comes into play, with each additional feature extracted from a data set creating a more complex training process while potentially adding little to no value (Friedman, 1997). Without active reduction, a feature set can rapidly become too complex to train a classifier within a reasonable amount of time.

The practice of reducing the data used for training and extracting useful features may be referred to as feature engineering. The purpose behind feature engineering is to transform a set of data that

is too large or complex to use into a manageable form. Feature engineering may involve aggregating multiple values into one, removing unneeded features that provide little or no value to the task at hand, or finding value in the interactions of features. Unfortunately, the best approach to engineering the features of a data set to make them more useful is not always clear.

While many approaches to improve the predictive power of a given data set exist, no best fit method will apply to more than a specific assortment of similar data sets. Single-featured, time-series data sets may be particularly difficult to extract information from if they contain no trend or seasonality. A given time-series data set may contain predictive information over any range of time or type of pattern. The challenge is now determining an efficient and automated method to extract maximum predictive value from a range of single-feature time-series data streams. Once a method is developed, available time-series streams which meet the necessary input format may be utilized in a more effective form to attempt to forecast oncoming cyber-attacks. Even without high precision, any method to provide forewarning of cyber-attacks could be very useful. In the realm of predicting a major attack, a false positive warning is far less harmful to a target than failing to predict an event altogether.

2. Problem Statement

All the relevant data in the world is of no use when solving a problem if the data cannot be practically applied. Large data sets regularly require extensive dimensionality reduction to be used in the training of machine learning classifiers. Although the problem of extracting value exists for all data types, time-series data can be particularly challenging to use efficiently. Time-series data, such as the sample shown in Figure 1, is defined as a sequence of observations recorded at distinct time intervals and stored in chronological order, usually recorded over successive time intervals with consistent spacing. A time-series data stream can represent anything from barometric pressure to occupants of a building at various points in time.

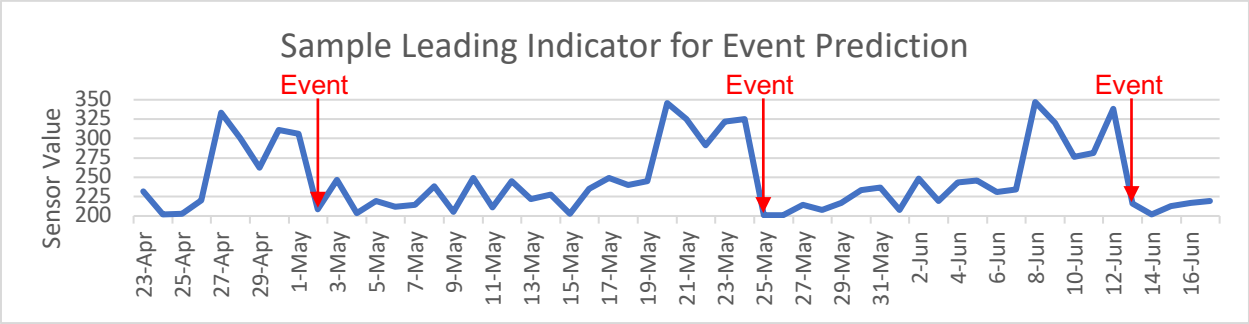

Figure 1 - Leading Indicator Time-Series Example

An acquired data stream may be used to generate predictions against a provided Ground Truth (GT). The GT is a data series containing the time-stamps of occurrences of an event type to be predicted. Training a classifier on the GT series requires converting the provided time-series data stream into a useable feature set. Training on the entire time-series is often infeasible and almost never the best approach, as each data point would be handled as a separate feature by the classifier and dimensionality could become a problem. The challenge becomes developing an algorithm to reduce time-series data in order to condense the resulting feature set while extracting significant predictive value and retaining the temporal nature of any trends or patterns.

The purpose of the research moving forward is to determine an efficient, automated, and effective method to extract useful feature sets from time-series data streams for use in predicting if an event will occur on a given day or not. A series of transformations will be performed on single-feature time-series data streams to determine how each transformation must be applied for best performance. The transformations are meant to clarify or retain patterns that exist in the data while reducing the size of the feature set being used to train a classifier.

The transformation parameters to be determined by the process include lead time of predictions (*lead time*), length of data used for feature set creation (*tail length*), the length of aggregation periods used (*bin size*), and sensitivity to abnormally high values, referred to as spikes, in the data (*sensitivity*). The resulting feature set extracted from the transformed data will be used to train classifiers for event prediction. The process has been designed to both reduce the feature set for the purpose of faster classifier training as well as extract maximum predictive value from any pattern that exists within the provided data set.

Testing the transformation method will involve measuring the performance of classifiers trained on the transformed feature sets versus classifiers trained on a baseline feature transformation. The baseline features will be a set containing the untransformed time-series values. The performance of baseline features will be compared to the performance of the transformed feature sets to determine if performance was improved. The significance of the performance differences will clarify any benefit the proposed transformation method provides for event prediction.

A system to utilize prior knowledge to predetermine a best fit transformation set and classifier type will also be developed and tested. Testing of the prior knowledge selection system, further referred to as the landmarking system, will be conducted using generated data stream and event set pairs. The generation of artificial testing sets will allow for a set of controllable generation parameters determining event frequency and distribution, as well as the shape of predictive patterns in the data stream. The transformation and classifier combination selected by the landmarking system will then be compared to the performance of all transformation sets recorded to determine the significance of the tradeoff in classifier performance. The effect of varying the number of top performing similar transformations selected by the landmarking system will be examined as well.

The testing process will determine the landmarking system's capability to accelerate transformation set selection without significant performance loss. If the landmarking system is able to select a transformation set that creates a classifier close enough to the best observed transformation while also saving a significant amount of run time versus total enumeration, the system will be considered successful. The other significant performance measure is whether the best observed classifier produces a statistically significant performance improvement over the baseline feature set. The results of the experiment will determine whether the transformation process paired with the landmarking system yield a significant improvement or not.

The purpose of the transformation method is to convert single-feature time-series data into a form which clarifies predictive patterns and improves classifier performance. The method may be used to find the predictive power of a provided time-series data stream. The ability to do so would be valuable to improving predictions of a provided event set such as cyber-attacks. The ability to

complete data reduction, feature extraction, and classifier selection in an accelerated way using the landmarking system would allow for testing to be conducted on more data streams in less time than could be achieved through total enumeration. The likelihood of producing a useful feature set from a provided time-series data stream would be increased.

3. Literature Review

Review of literature pertaining to cyber-attacks, feature engineering and classification methods was conducted to establish a "best-practice" wherever possible in developing an automated feature engineering system. Understanding what methods have been conducted in creating better cyber-attack prediction methods ensures that the proposed research does not retrace any previously tested models. While the system focuses on the intelligent feature engineering of time-series data streams, methods used previously on both time-series and non-time-series data can reveal techniques that may be tested, and if successful, included in the final system. The final piece of the proposed method is the classification method being employed to create a prediction from the transformed data stream. While finding the best values of the classifier parameters themselves is not the primary focus of the proposed research, being a data transformation experiment, confirming that the classifiers tested are appropriate for the task at hand, and will yield useful testing results, is important to the experiment as a whole.

The literature review begins with an evaluation of current methods being applied to the forecasting of cyber-attacks. Models and methods which are shown to be effective in the realm of cyber-security prediction are used to help develop the approach of the proposed system. In addition to determining research methods that have been shown to be effective, determining where they have fallen short assists in guiding the newly developed method towards a novel objective. The next section of the literature review examines current feature engineering techniques to determine what methods yield performance improvement. The following section provides background on the application and performance of the classifiers that are to be included in the automated feature engineering system. The final section of the review examines techniques used to select a best fit

classifier, which will be adopted for use in the proposed system. The section also discusses the use of meta-features to improve selection of the transformations and classifiers used in the proposed research.

3.1. Event Prediction Using Forecasting and Machine Learning

Forecasting techniques are commonly used to generate predictions using a time-series data set. One study reviewed tested the application of an ARIMA model to the field of cyber defense in event prediction(Werner et al., 2017). In the model, time-series data containing information regarding the time and type of cyber-attacks is used to train a forecasting method meant to predict the number of cyber-attacks on a given day.

When tested on historical attack data, the ARIMA model followed spikes in occurrence data, albeit with some lag behind the initial spike, indicating that using ARIMA forecasting leads to better performance than simply following the mean. Prediction error for less frequently occurring events such as denial of service or malicious URL was twice that of attacks on internet facing servers, in part due to forecasting methods performing better when the data set being examined contains a fewer number of zeros. One of the limitations of ARIMA forecasting is that increasing sparsity in data has a negative performance impact. The data patterns contained in the test data used in the proposed thesis may also be sporadic and difficult to identify on a large scale, which can lead to reduced performance using traditional forecasting methods.

Experimentation has also been conducted using a Bayesian Network to attempt to predict cyber-attacks (Okutan, Yang, & McConky, 2017). The tested method involves incorporating outside data sources such as social media and open source projects consolidated over a set period of time as

signals into the training of a Bayes Net. Once all the signals are formatted, an individual Bayes Net is trained for each of the five types of attacks being investigated.

Through 5-fold cross validation testing Bayes Net was found to perform well on more frequent attack types while failing to predict any of the sparse DDoS events. The results reflect an issue of the Bayes Net valuing precision in negative event prediction over positive event prediction when trained on sparse data (less than 9% occurrence rate).

Research conducted using a Support Vector Machine and the traffic of malicious IPs showed the method effective at predicting cyber-attack incidences up to 3 months prior to their occurrence (Liu et al., 2015). A feature set was generated to capture the behavior of malicious IP addresses in a way which reveals anomalies in their behavior as individuals as well as a group. Testing was performed using subsets of these features to determine which features yielded the greatest performance gain. The study validated the use of a separate set of leading indicators to predict the timing of a major attack on the Univ. of Maryland. All 3 classifiers were capable of indicating a high risk of an oncoming attack, however, an interesting result was that the highest performing classifier was that trained only on Nov-Dec-Jan data rather than Nov-Dec-Jan-Feb data. The results indicate that the classifier may achieve highest performance with a certain amount of lead time >1 day.

Another research team developed a method to create a general cyber-attack forecast built on Japanese tweet counts and the activity of Twitter accounts of 90 know hacktivist groups (Munkhdorj & Yuji, 2017). An experiment was conducted using the frequency and sentiment of

the tweets of hacktivist groups. An artificial neural network (ANN) was trained and tested using these factors on 34 unique cyber-attack events.

Results of the experiment show that the activity of accounts known to be linked to cyber-attack display a pattern of higher than average activity during the month prior to an attack occurring for all events tested. The experiment also stated that a limitation of examining the hacktivist data was the sparsity of activity, where the data examined in the proposed thesis is more densely populated, providing the capability to examine the time frame preceding each cyber-attack with more granularity.

Major issues, such as predicting infrequent events, faced with each of these methods are addressed through the transformation method applied in the proposed thesis. Predicting infrequent events is difficult using the aforementioned forecasting or Bayes Net approaches because of their innate tendency to predict all negative for a sufficiently sparse event type, resulting in high precision, but providing little real value. By using the new method being investigated to utilize recognizable data patterns as opposed to single aggregated values, the classifier will be better able to predict sparse events. Creating a method meant to accurately predict when sparse events occur could complement a traditional method by being used on attacks below a set occurrence rate. Liu et al., (2015) shows SVM's validity for predicting cyber-attack and for utilizing large feature sets. Liu et al. also suggest that prediction accuracy may be maximized through testing on lead times > 1 day, which will be incorporated into the developed transformation system. Munkhdorj & Yuji, (2017) shows that analysis of Hacktivist groups provided some predictive value. The data to be used for the proposed thesis closely resembles the hacktivist activity set in that only tweets containing

information known to coincide with malicious activity are contained. The combination of utilizing a more correlated, closely populated data set as well as the testing of multiple classification methods is expected to clarify the predictive value of the detected activity increase prior to attack occurrences.

3.2. Feature Reduction and Extraction

The data sets examined through the proposed research are numerical counts in the form of time-series data streams. The temporal data provides a basis from which forecasts or predictions may be generated, often used to estimate future events or behaviors (Antunes & Oliveira, 2001). Most time-series data streams contain a vast number of values, many of which may provide little predictive values in their initial form. For example, a data set containing the readings of 10 sensors within a system each providing a reading every 2ms would generate 5000 unique values/ second. Perhaps the user is interested in predicting an event which is influenced by the difference in 2 pairs of these sensors as opposed to the raw values of all 10. In addition, the event being predicted may be preceded by spikes every 30 seconds. An improved data set may contain only the differences between the 2 sensor pairs and an aggregated value for each second. Through the proposed transformation, the data set would be reduced from 5000 features and 10 unique values/second to 2 features and 1 value/second. Reducing data sets through similar feature reduction techniques, such as aggregation, removal, and similar transformations, is a key focus of the proposed transformation method.

The goal of feature engineering techniques is to remove features which provide little or no predictive value and to combine any *n* features that may provide equal or greater value when fused to produce a single feature. The removal of low-value features aids in speeding up training and run

time of classifiers, as well as reducing "noise" which negatively affects the accuracy of predictions generated. The aggregation of numerical features may be achieved through a variety of reduction techniques such as summation, averaging, or many other methods.

Another study examined utilized multiple data transformation techniques in order to create useful features out of 2 large data sets (Yu et al., 2010). The method used involved assessing performance metrics of a set of students in order to predict their future performance. After creating categorical features to identify question type, student id, and other identifying characteristics, the first data set had approximately 1,000,000 unique features and the second data set had approximately 200,000 unique features. Training classifiers on these data sets remained infeasible due to training time/processing power restrictions.

To reduce the number of features in the student data, past performance metrics on identical question types were averaged to create a single past performance measure for each question type. In addition to averaging matching features, single features were generated for each student, each identical problem type, etc. which measured the percentage of attempts which were correct the first time. Another transformation technique employed was creating a set of 4 binary features indicating student familiarity with a given question type, indicating if a student had encountered a given problem type within a prior window of time. The final data sets used in the research project had been reduced from more than 100,000 features to only 17.

Performance was measured first using the untransformed data set, followed by the 17-feature data. The results of classifier testing showed that with a random forest classifier better predictive

performance was achieved using both of the condensed data sets compared to using the untransformed data. The results display that a massive reduction in features, 99.9983% and 99.9915% respectively, was achieved without sacrificing significant performance. These types of transformations extract the useable information from patterns within a data set and are similar to those being employed in the transformation system being developed. The performance improvement achieved shows the promise of being able to extract and simplify the meaningful value of a large data set down to a small set of features through feature engineering.

The Heritage Health Prize (HHP) competition focused on predicting the likelihood of patients to be admitted to a hospital based on historical data("Heritage Health Prize," 2012). The first-place team used two distinct methods of feature reduction in order to create multiple testing data sets (Brierley, Vogel, & Axelrod, 2011). The first method employed by the researchers was yearly aggregation, resulting in (# of years * # of unique patients) instances to be trained on. The second method tested was aggregating all claims by patient. The combined results yielded a .0035 improvement in root-mean-square error over what any one model could achieve on the original data sets, a significant improvement with the difference between the 1st and 5th place teams being <.002. Using the described method of aggregating multiple data sets over varied aggregation periods could yield both higher performance than using a single test set in the proposed transformation system.

In the AAIA'15 Data Mining Competition competitors were tasked with classifying the positioning and activities of firefighters by using real-time tracking data (Meina et al., 2015). A

research team used histograms of occurrence frequency within the 42 unique time-series data streams per firefighter to train a Support Vector Machine (SVM) (Zdravevski et al., 2015).

Testing results showed that when the team aggregated the provided data streams into small windows of time, with the total number of aggregation periods referred to as B, the use of more, and therefore smaller, aggregation windows frequently resulted in overfitting of the SVM. The top 14 performing classification methods used only $B = 30$ or $B = 50$ while $B = 100$ resulted in reduced predictive performance.

The results of the study suggest that training on only the most valuable portion of the training data rather than the full data stream improves classifier performance while reducing complexity and training time. The method of generating training histograms employed in the experiment sacrifices the temporal component of the data set, which the proposed thesis system seeks to retain through utilizing a similar binning process aggregated by time as well as value.

3.3. Classification Methods

An algorithm, or machine learning model, may be trained to classify an outcome based on a set of values provided. The value set provided are related to a set of features common to the data set upon which a given classifier was trained. Classification algorithms allow predictions to be made regarding the likelihood of an event occurring through examining one or more related data sets.

Classification methods use a variety of algorithms to determine how best to classify a given instance based on the values of its associated features. The way in which classifiers produce a

prediction differs, which may result in different classifier types producing different results when provided the same data set for training and testing.

The classifiers used in the proposed thesis are binary classifiers which predict 0 or 1 when given a set of features. An algorithm for classifying new instances is generated based on the correlation between values of the features of examined instances and their respective categories. The use of labeled examples for training is known as supervised learning, as opposed to unsupervised learning in which the algorithm must generate classes within the data (Brownlee, 2016).

The "No Free Lunch" theorem stated that one cannot determine a best fit classification method based on prior knowledge alone, the classifier must be tested against a ground truth. Different classification algorithms may be more effective in handling factors adverse to effective machine learning such as sparse data sets or vast feature quantities. Because of this, multiple distinct classification methods will be incorporated into the system to allow for the widest range of classifiers tested per dataset. In order to conduct the study on a large number of classifiers in a practical way, the WEKA machine learning library will be utilized (Frank et al., 2009). The WEKA classifiers to be tested in the experiment are an Unpruned Classification Tree, a K-Nearest Neighbors algorithm, a Bayes Net and an SVM classifier.

The Classification Tree and KNN algorithms will be included as both have shown to provide differing results with reasonably consistent performance in a previously discussed meta-learning experiment (Reif, Shafait, & Dengel, 2011). Because these two classifiers utilize fundamentally different algorithms while having straight-forward parameters, *K* for KNN and pruning level for

Decision Tree, they will function well early in the research for comparison. Random Forest, an extension of the Decision Tree algorithm will not be used for the research case as the training demand would be much higher than a Decision Tree. Any additional predictive performance Random Forest would provide is unneeded as measuring performance changes across different data transformations is more important to the experiment than overall precision of the classifier.

The Bayes Net classification method was selected due to robustness when handling many features which have little predictive value as well as use in similar research examined (Okutan et al., 2017). The SVM classifier has been shown to be effective in previous research regarding data transformation testing so will be included in the proposed thesis (Liu et al., 2015;Meina et al., 2015).

3.4. Meta-Features for Classifier Selection

Determining a best fit classification method has been a problem since the origin of machine learning. According to the "No Free Lunch" theorem, first proposed by David Wolpert, one cannot determine what supervised classification method will yield the best results based on prior results alone (Wolpert, 1996). One method proposed for solving the problem of finding a best performing classification method is through landmarking (Bernhard, Hilan, & Christophe, 2000). Landmarking was developed through a study in which the landmarking system was used to predict a best fit classifier through the performance results of a small set of experiments, or "landmarks". Landmarks are algorithms which have performed successfully on a specific data set, or better yet across a broad category of tasks (i.e. image detection, traffic pattern recognition, etc.).

Meta-features are features which exist as a quality of the data set as a whole. Research into a field referred to as “meta-learning” aims to study the effect of “meta-features” on the performance run times of classifiers (Doan & Kalita, 2016). Simple meta-features refer to features pertaining to a data set such as number of features, number of classes, number of instances, etc. which do not require extensive calculation to be derived from the data set. Applying transformation methods in a novel way to time-series data as well as utilizing meta-features to improve performance is the primary objective of the proposed research.

A study was also conducted with the intent of examining the effects of simple meta-features on the run time of different classification methods (Reif et al., 2011). The results gathered through the testing of 5 classifiers (KNN, SVM, MLP, Ripper, Decision Tree) on a traditional grid search problem suggest that there is significant correlation between the values of a set of “simple meta-features” and the run-time of a given classifier. The normalized absolute error of the predicted runtimes of each of these classifiers using only simple meta-features was ~.6, resulting in 40% lower error than when not considering these features. In addition to demonstrating that reducing the factor set assists with altering run-time, the results show that examining meta-features with relation to transformed data sets in the proposed system may allow for identification of data sets with beyond feasible run-times before they are tested.

The meta-features of the data sets which result in their similarities are then identified. By grouping a new data set with which there is no prior testing with other data sets based on its meta-features, the classifier which has the best performance on the similar data sets may provide a good starting point for testing. Bernhard et al., (2000) tested the aforementioned method in the landmarking

study and discovered promise for application of the system in improving classifier selection. A similar method can be employed in attempting to determine which classifier and transformation values may work best on a given data stream, using features such as the number of events in the event set, variance of the values within the data set, length of historical data available for training, etc.

4. Methodology

The methodology first introduces the proposed transformation developed and tested for the study. Once all steps of the transformation process have been examined and their application as a single process is presented, the experiment for testing the transformation process is introduced. The process of generating test data sets to measure the performance of the transformation process is explained in depth followed by the procedures used to narrow down these data sets and experimental parameters for more efficient testing. The proposed method of measuring the performance of the transformation process and validation testing using real data is then proposed.

Section 4.3 of the methodology introduces the landmarking system and explains how the landmarking system is intended to improve the efficiency of the proposed transformation method. The functionality of the landmarking system is then explained as a step-by-step process to detail the purpose of the system. Finally, an approach for measuring how effectively the landmarking system performs and an experiment designed to measure the system's performance is laid out. The last section covers exactly what performance measures were collected and how they were used to gauge the usefulness of the landmarking system.

4.1. Transformation Method

The method used in the proposed research requires a specific format of data to perform certain transformations. The restrictions on the format of the data, requiring a time-series numerical data stream and set of event time stamps, are key to the operation of the process as a whole. Some approaches are used to modify the provided data streams to enhance the performance of the method.

The proposed method applies all transformations put forth by the study. The transformations include spike conversion, which involves transforming the data set from a data stream with a continuous range of numerical values to a set of binary values. Additional transformations include reducing the number of data points considered for training, known as tail length, as well as testing differing spans of time intervals between the data to be trained on and the event to be predicted, known as lead times. The final transformation used involves aggregating these data points into bins of a predefined width.

4.1.1. Transformation Method Input

Input Terminology

D	Continuous single-feature time-series data stream to be used as a leading indicator
GT	Series of time-stamps of events to be predicted by the trained classifier

The instances within the provided time-series data stream each contain a timestamp and corresponding value. These sets may contain 0 values, however the "Time" column must be uniform with no missing values. Having no missing values allows the transformation method to better identify temporal patterns without inconsistent spacing of time-intervals. Formatting of data streams and backfilling of missing values was not considered part of the core program tested and was handled through external scripts. Each instance D_i consists of a time $D_{i,t}$ and level $D_{i,l}$.

A data set containing the timings of events to be predicted must be provided in order to train the classifiers as well as generate performance metrics. After removing any duplicates, the stream of unique times, each corresponding to a single time within *D*, are imported as *GT*. *GT* is made up of

individual instances GT_i. Partial examples of appropriate supplied D and GT data sets, prior to removal of GT duplicates, are shown in Figure 2.

a)

DDoS Attack Dates

```
1  2016-05-05
2  2016-05-06
3  2016-07-22
4  2016-07-25
5  2016-07-25
6  2016-07-25
```

b)

Daily Twitter Mentions

```
1  "2016-03-04",11
2  "2016-03-05",336
3  "2016-03-06",322
4  "2016-03-07",497
5  "2016-03-08",1652
6  "2016-03-09",2217
```

Figure 2 - Sample GT(a) and D(b) Data Streams

GT is then subdivided into a training set, GT_{Train}, and a testing set, GT_{Test}, such that:

$$GT_{Train} \cup GT_{Test} \leq GT \tag{1}$$

after removal of duplicate event dates. In order to train the classifiers using GT_{Train} both positive, $GT_{Positive\ Train}$, and negative, $GT_{Negative\ Train}$, event instances must exist within the data set such that:

$$GT_{Positive\ Train} + GT_{Negative\ Train} = GT_{Train} \tag{2}$$

Having close to a 1:1 ratio between positive and negative occurrences in GT_{Train} is preferred in order to avoid generating a biased classifier (Kubat & Matwin, 1997). Therefore, simply using all dates not contained in $GT_{Positive\ Train}$ as negative occurrences, although valid, was not the chosen method for the experiment. The approach taken for the purpose of the research is to calculate a time centrally positioned between each $GT_{Positive\ Train,i}$, $GT_{Positive\ Train,i+1}$ timestamp pair where:

$$GT_{Positive\ Train,i} = time\ of\ i^{th} event\ occurance\ in\ training\ set \tag{3}$$

$$GT_{Negative\ Train,i} = \left\{ \frac{GT_{Positive\ Train,i} + GT_{Positive\ Train,i+1}}{2} \right. \tag{4}$$

$$\forall\ i\ in\ GT_{Positive\ Train}\ where\ i \neq i+1$$

These times are used as the $GT_{Negative_{Train}}$ for training purposes so as to provide maximum disparity between the spike profiles of positive and negative occurrences. The process of creating an equal number of positive and negative GT_{Train} events is referred to as leveling the training set. In the case of a $GT_{Positive_{Train,i}}$, $GT_{Positive_{Train,i+1}}$ pair occurring at sequential times, no negative event is generated and the $GT_{Positive_{Train,i+1}}$ event is removed.

The process for creating the negative events of GT_{Test} involves simply labeling every time instance not contained within the provided event set as a negative occurrence. The creation of these negative occurrences is done under the assumption that the provided *GT* is complete for the associated span of time contained. GT_{Test} cannot have empty or irregular time-intervals in order to allow the classifier to test on each time instance in sequence.

4.1.2. Spike Transformation

The first transformation applied to each data stream is a conversion from the individual raw values, referred to as $D_{i,l}$, to individual "spikes", referred to as P_i. The most basic application of the spike transformation utilizes the formula:

$$P_i = \begin{cases} 1 \; if \; D_{i,l} \geq s\sigma + \mu \\ 0 \; otherwise, \end{cases} \qquad (5)$$

where *s* is sensitivity or number of standard deviations and σ is the standard deviation of the values of D_l and μ is the mean of D_l in the training set. The transformation to spikes is meant to clarify patterns of abnormally high values in the data set and reduce the raw values to a set of binary values, resulting in a simplified training process. An example of an application of the spike transformation and the resulting pattern clarification are shown in Figure 3.

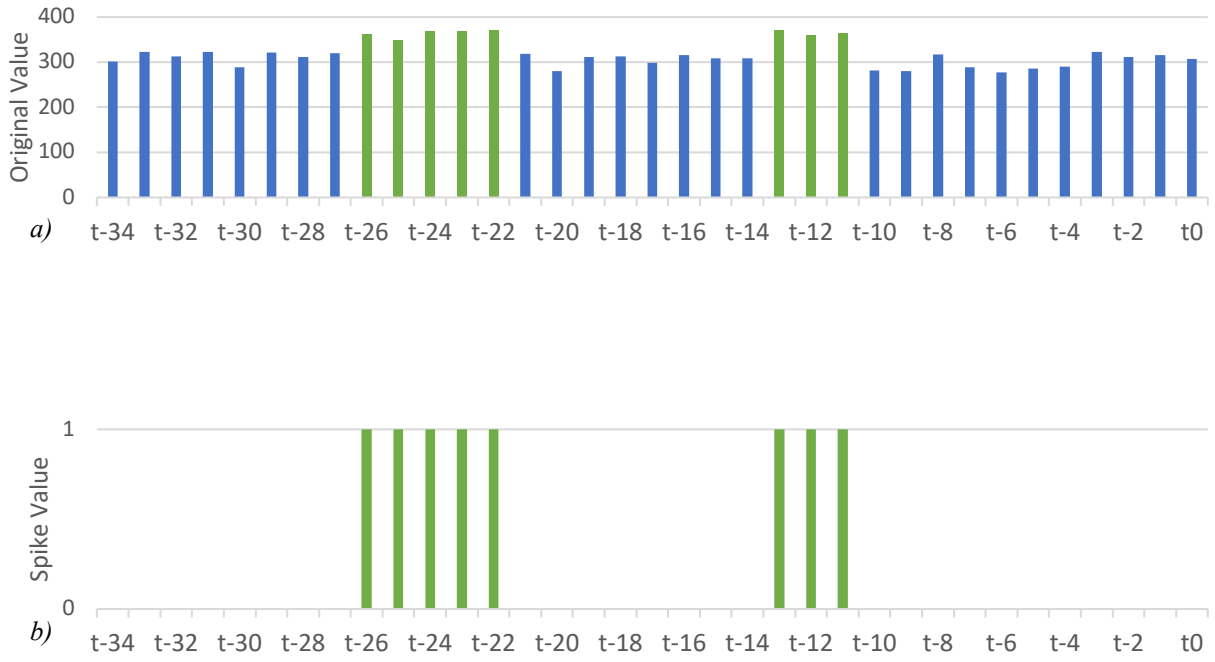

Figure 3 - Sample Spike Transformation before (a) and after (b) at s = 1

The spike transformation is also used to normalize values across multiple data sets, allowing for easier inclusion of additional data streams and potential cross-set transformations such as combining spikes with matching time-stamps. The transformation may be modified through varying the number of standard deviations (σ), referred to as sensitivity s, above the mean necessary for a value to be transformed into a "spike", as well as testing numerical transformations on values based on how many standard deviations over the mean they are.

4.1.3. Determining Tail Length and Lead Time

Further transformations to be tested include adjusting the number of data points included in each training set, referred to as Tail Length (T_L) and testing the method over a variety of lead times (N), the number of data points prior to the event being predicted at which the training set concludes.

The lead time parameter is illustrated in Figure 4 as N and allows a classifier to train on patterns occurring greater than 1-time interval prior to an event.

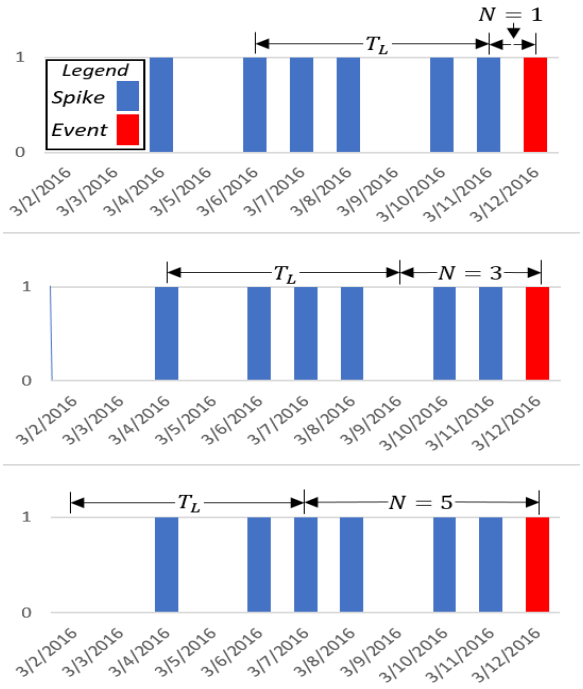

Figure 4 - Lead Time Profiles

T_L is varied as a parameter of the transformation method to determine the minimum amount of data to be included while improving predictive performance. In addition to the specified tail length T_L, the value of the lead time N for a given test affects which data points are analyzed leading up to an event time. For $N = 1$, the spike profile used to train consists of the data points immediately preceding the event time. If $N > 1$, the spike profile must be shifted to account for the gap between the event to be predicted and the time at which a prediction is generated.

4.1.4. Creating Binned Spike Profiles

Once the data stream has been subsetted into T_L length training sets with lead time N, the spikes are aggregated into a set number of bins (B_i), where i = *bin location in array*. The width of each

bin, A, is another parameter of the transformation method which was varied in testing. All P_i within the bin are summed following the formula:

$$B_i = \sum_{i=1+A(i-1)}^{Ai} P_i \qquad (6)$$

to generate the value of B_i, which is then appended to the training set. An example transformation using the parameters in Table 1 is shown in Figure 5.

Table 1 - Sample Transformation Parameters

Parameter	Value
T_L	8
A	4

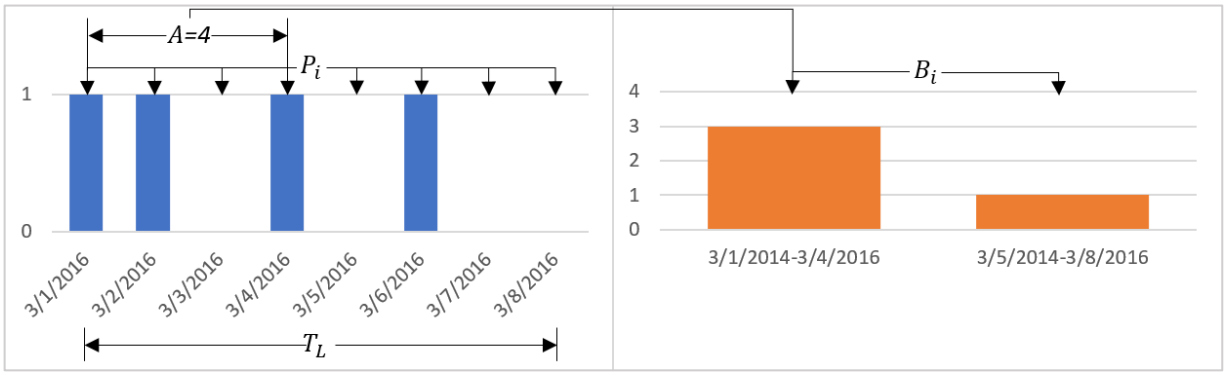

Figure 5 - Spike Binning Transformation Example

Because the generation of B_i's relies on the aggregated values at each time stamp, only factors of T_L are used as values of A in the testing process. The process of binning P_i's into B_i's to construct a spike profile B with length $\frac{T_L}{A}$ is repeated for every value in GT_{Train} and GT_{Test}.

The binning process is meant to both reduce the number of features required to train a classification engine, as well as expose underlying trends or patterns in the training data. The bins are expected to capture irregular patterns in a form more useful for classification than raw values would. Within a given pattern, activity at a particular point in time may matter less than activity over a span of time, which a bin captures.

The binning process also holds a distinct advantage over total aggregation of a training set through maintaining temporal patterns. An example of how a pattern may be lost through aggregation is shown in Figure 6.

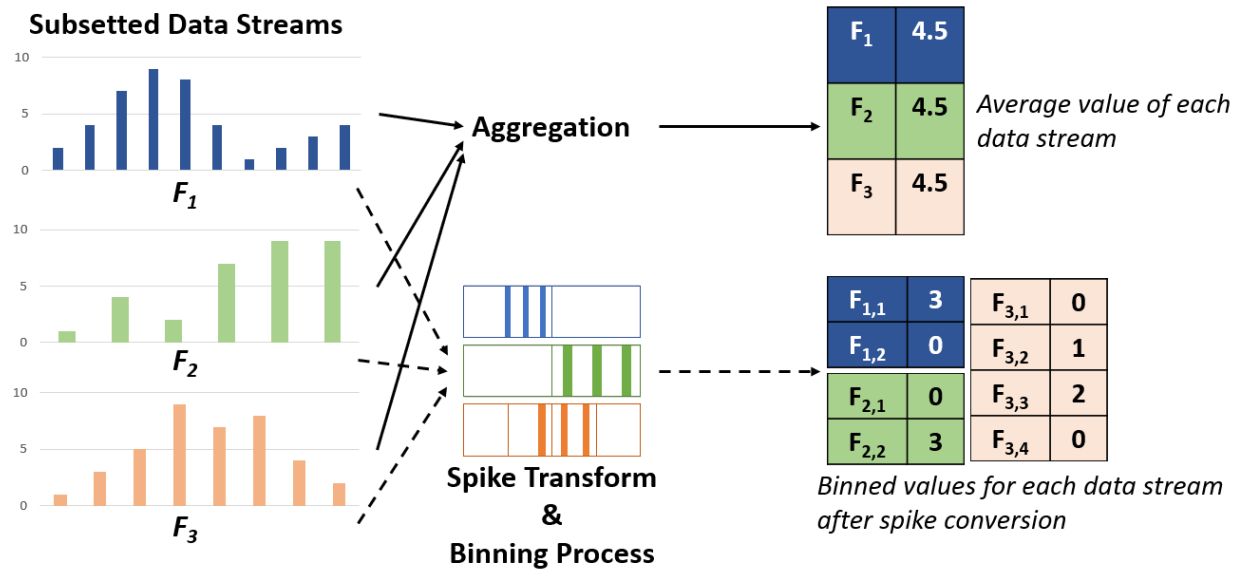

Figure 6 - Maintaining Temporal Patterns: Transformation Method vs Aggregation

As shown through the conversion of the three subsetted data streams to feature sets, total aggregation views all data streams identically while binning maintains the temporal patterns while reducing the number of features.

Table 2 shows an example of two consecutive instances that could be contained in GT_{Test}, along with the values of each feature of these instances both before and after their spike values have been binned using $A = 2$.

Table 2 - Sample Consecutive GTTest Instances Before and After Binning

F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	Event
0	1	0	0	1	1	1	0	1
1	0	0	1	1	1	0	0	0

F_1	F_2	F_3	F_4	Event
1	0	2	1	1
1	1	2	0	0

4.1.5. Transformation Process Overview

A process to identify the best set of transformation parameters is shown in Figure 7 and functions through being provided with both a time-series data set D as well as a set of event times known as the ground truth GT. The GT is then leveled resulting in an equal number of positive and negative events to provide an unbiased training set, after which transformations are applied to D_l, the values of D, based on the timing of both the positive and negative events in GT. The transformations involve first converting the time-series values into a binary set using the spike transformation. The tail length T_L and lead time L_T are then used to subset the data and adjust the number of data points used for each event prediction. The values are then aggregated into bins B_i containing a predefined number of data points A as a reduction technique to create both a training and testing set to be used for classifier testing.

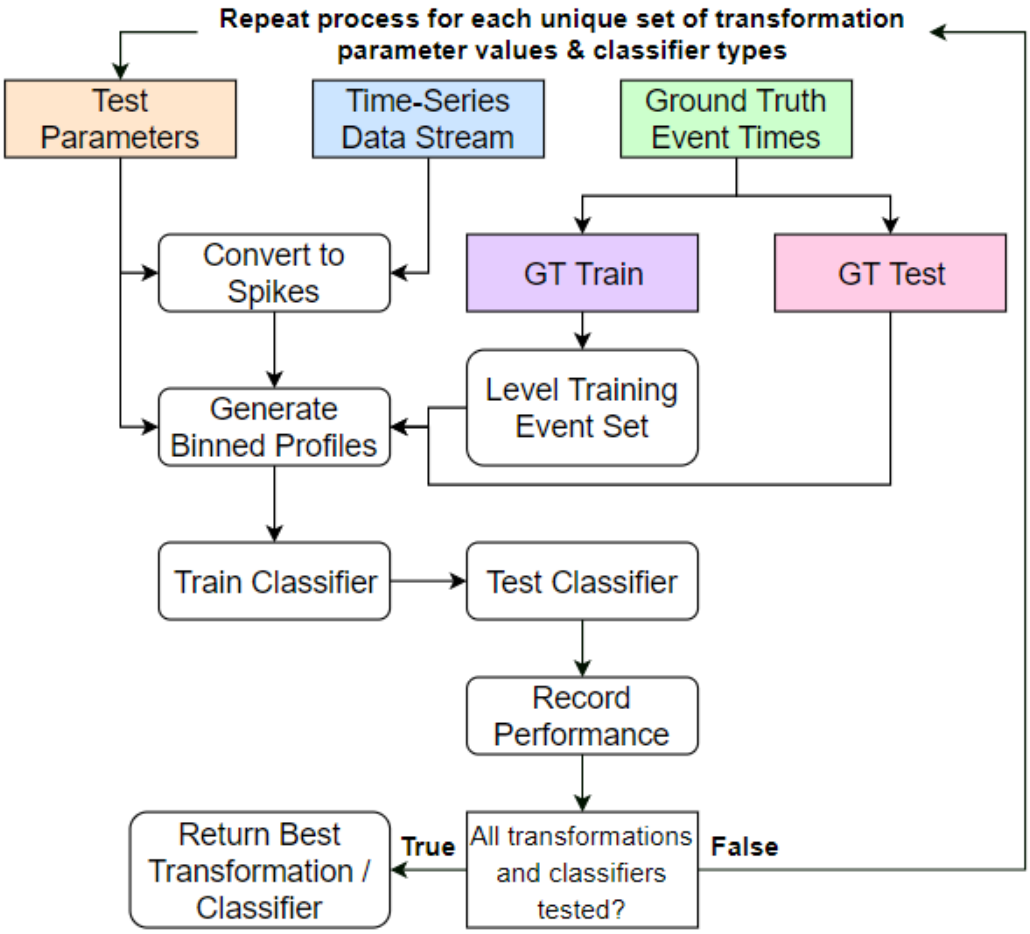

Figure 7 - Transformation System Overview

After all transformations have been performed, the transformed training sets are used to train a classifier using the GT_{Train} event set. After training, the classifier is then used to predict events. In the case of the testing process, the events being predicted are those contained in GT_{Test}. Each time-interval of GT_{Test} is used to create a unique training set based on the defined transformation parameters. The training set is then used by the trained classifier to create a prediction of whether the time-interval in question contains an event. Performance is then measured based on how well the classifier performs over the entirety of GT_{Test}.

4.2. Transformation Method Experimentation

The following sections cover in detail how the data used to test the proposed transformation method and landmarking system was generated. Performance metrics were gathered through the process's predictive accuracy when trained on all points of each generated GT_{Test}. Methods for reducing the number of generation parameters through a screening experiment and levels at which these parameters must be tested through a limit experiment are discussed. The approach to testing the transformation process used in the study is explained, as well as how the transformation process was tested on a real data stream for validation of the proposed method. The effectiveness of the proposed process when tested on real data was measured by performance improvement compared to when no spike transformation is applied and $A = 1$, representing the raw values.

4.2.1. Data Set Generation

In order to test the developed transformation method, testing data sets were generated using a known set of parameters. These parameters will allow for analysis of the effects of particular variables on the method's ability to detect and utilize patterns. The effect of each of these generation parameters were analyzed through a screening experiment to allow for the elimination of any parameters with little effect or no further value to the research being conducted. Following the screening experiment, a limit experiment was conducted with the purpose of establishing effective upper and lower values at which to test the generation parameters.

4.2.1.1. Data Set Generation Procedure

A controlled experiment was conducted to determine how well the transformation process performs at detecting unique pattern types using data sets and ground truth sets with varied

characteristics. The experiment first involved generating *GT* event sets with different distributions as to when and how often events occur. By testing on a variety of *GT*s determining limitations or highest performing event distributions in terms of prediction accuracy of the generated classifiers is possible. For each of the *GT*s, multiple *D* data sets were generated containing different predictive patterns and noise levels in D_l. By testing on multiple *D*s for each *GT*, features of D_l and the contained patterns may be altered and their effects on predictive performance measured.

Artificial *GTs* were generated to allow for performance testing of the developed method on a variety of event distributions and frequencies. Each artificial *GT* was generated by applying a probability distribution over a specified span of time to determine the likelihood of each time stamp containing an event. The parameters affecting the *GT* set are shown in Table 3.

Table 3 - GT Generation Parameters

Parameter	Description	Test Values
$\boldsymbol{GT_{TL}}$	Time intervals covered by *GT*	*50, 100, 200, 500*
$\boldsymbol{GT_{Dist}}$	Probability distribution of an event occurring on a given day	*Uniform, Sine*
$\boldsymbol{GT_{Prob}}$	Average likelihood of an event across the entire *GT*	*.05, .10, .20*
$\boldsymbol{GT_{FP}}$	Likelihood of any generated event being a "false positive"	*.10, .25, .50*

The number of points in time considered for the *GT* is determined by the value of GT_{TL}. The frequency of *GT* events and their distribution along the length of the *GT* are determined by GT_{Prob} and GT_{Dist}. GT_{Prob} is a probability distribution spanning the length of the entire *GT*. The probability distribution determines the likelihood of any data point within *GT* being an event, which is dependent on where the data point falls with relation to the start and end of the *GT*.

GT_{Prob} determines the mean probability of GT_{Dist}, providing a way of estimating the number of events to be generated based on the equation:

$$Total\ GT\ Events \approx GT_{TL} \times GT_{Prob} \quad (7)$$

In addition to the generated *GT* events, a certain number of negative events are created to generate uncertainty amongst the *GT* set. Each negative event is included in the *GT* when the predictive pattern is applied to *D*, but then removed prior to testing. Following the given procedure, the predictive pattern appears in *D,* but the corresponding event does not exist. GT_{FP} is the probability that any generated event is treated as a negative event. Using these artificial *GTs* allows for a set of controllable parameters to be varied in order to determine the effect of various factors on predictive performance.

From each of these artificial *GTs* multiple test data sets *D* are generated. The generated D_l contains a specified number of data points which follow either a Uniform or Normal distribution. Because the process functions through detecting anomalous values based on standard deviation, the specific mean and standard deviation of each data set does not affect performance. Each artificial *D* contains a defined pattern occurring prior to each event contained in the associated *GT*.

The parameters contained in Table 4 control the generation of the base data points contained in D_l as well as many of the characteristics of the pattern applied.

Table 4 - D Generation Parameters

Param	Description	Sample Values
$D_{Pattern}$	Predictive pattern used	*Upward, Downward, V*
$D_{PatternType}$	How the predictive pattern is applied	*Magnitude, Probability*
D_{Amp}	Size of generated pattern spikes in standard deviations	*.5, 1, 2*
D_{Dur}	Number of data points over which predictive pattern is applied	*5,10,25*
D_{LT}	Number of data points prior to event at which predictive pattern ends	*1,3,5*
D_{Rand}	Chance of each spike in a given pattern occurring in *D*	*.25, .5, .75, 1*
D_{Dist}	Likelihood distribution used to generate values between [0,100]	*Uniform, Normal*

D_{Dist} determines the random distribution applied to the base values generated for D_l before any predictive pattern is applied. The distribution determines how many "spikes" naturally occur within the data set, with a uniform distribution producing ~16% spikes and normal distribution producing ~29% spikes at $s = 1$. $D_{Pattern}$ is the predictive pattern applied to each event.

Table 5 describes the distributions defined by $D_{Pattern}$. The patterns are used to create spikes during generation of each artificial *D* using the defined parameters. Each of these distributions produce a probability curve spanning D_{Dur}. The probability distributions of each $D_{Pattern}$ allow for controllable variation between patterns within the same data set.

Table 5 - $D_{Pattern}$ Values

$D_{Pattern}$	Shape	Probability of Spike at $D_{GT_{Train_i}+j,t}$ while $0 \leq j \leq D_{Dur}$
Upward	Linear Slope	$P\left(D_{GT_{Train_i}+j,t}\right) = \frac{1}{D_{Dur}} * j$
Downward	Linear Slope	$P\left(D_{GT_{Train_i}+j,t}\right) = 1 - \frac{1}{D_{Dur}} * j$
V	2xLinear Slope	$P\left(D_{GT_{Train_i}+j,t}\right) = \begin{cases} 2 - \frac{2}{D_{Dur}} * j \; where \; j = [1{:}\frac{D_{Dur}}{2}] \\ \frac{2}{D_{Dur}} * j \; where \; j = (\frac{D_{Dur}}{2}{:}D_{Dur}] \end{cases}$

Figure 8 shows each of the values of $D_{Pattern}$ listed in Table 5 as they would appear applied to a sample data set D.

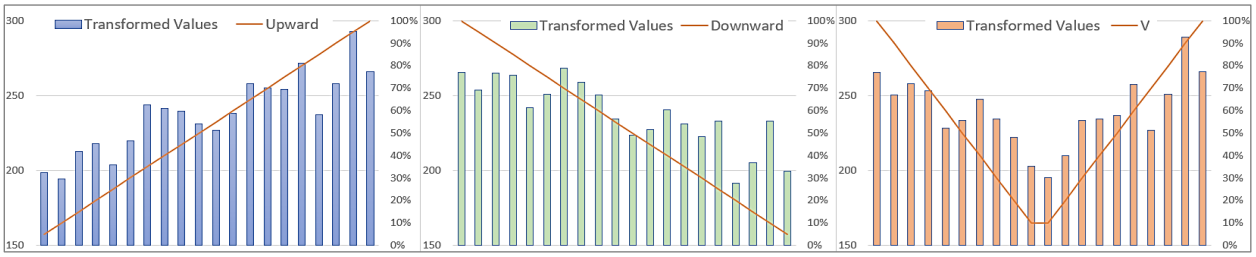

Figure 8 - $D_{Pattern}$ Samples

D_{Amp} controls a multiplier determining the amplitude of the predictive pattern being used. The value of D_{Amp} sets the number of standard deviations of each spike leading up to an event. A sample of what two patterns applied using different values of D_{Amp} may look like in the same D are shown in Figure 9.

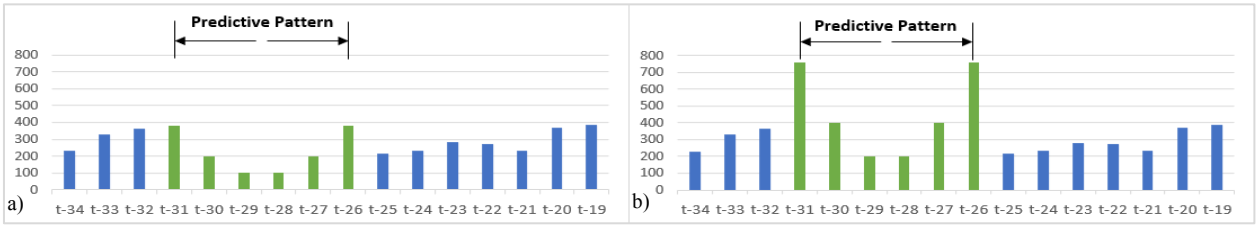

Figure 9 - Sample Patterns Applied at DAmp = 1 (a) & DAmp = 4 (b)

D_{Dur} determines pattern duration by specifying the number of data points over which the predictive pattern is applied prior to each event. Examples of two different values of D_{Dur} are shown in Figure 10.

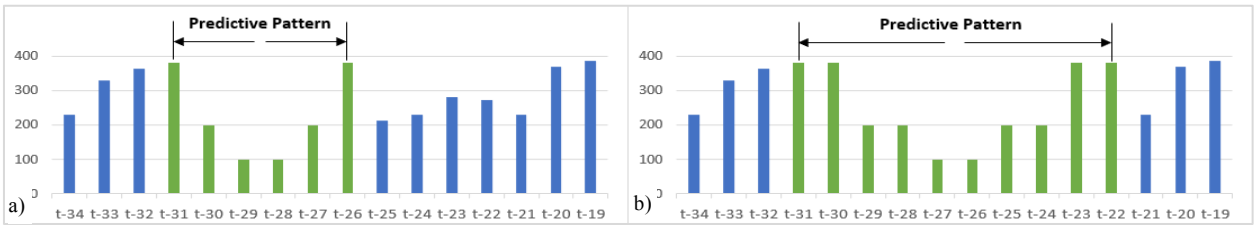

Figure 10 - Sample Patterns Applied at DDur = 6 (a) & DDur = 10 (b)

D_{LT} defines the lead time of each pattern, or the number of data points prior to each event that the predictive pattern ends. D_{LT} should always correlate with a best lead time for event prediction.

D_{Rand} creates variability within the predictive patterns. D_{Rand} is a likelihood that any given point in a pattern occurrence is not generated. If not generated, the value is left as the original generated value of the given point.

Table 6 defines the methods through which the distribution defined by $D_{Pattern}$ is used to apply a pattern to the given data stream, referred to as $D_{PatternType}$.

Table 6 - DPatternType Values

$\mathbf{D_{PatternType}}$	**Description**
Magnitude	*P(Di)* used as multiplier for amplitude of spike, defined by D_{Amp}, occurring at point *i*
Probability	$P(D_i)$ used as probability of spike occurring at point *i*

The first $D_{PatternType}$ method is magnitude, in which case the pattern value determined by $D_{Pattern}$ at a given point defines the amplitude of the spike occurring at the specified point. In the case of a magnitude pattern, each spike has a 100% chance of occurring, but the amplitude of each spike will vary. The second $D_{PatternType}$ value is probability, in which case the values generated by the $D_{Pattern}$ applied determine the likelihood of a spike occurring at a given point. In the case of a probability pattern, each spike with either occur with full amplitude or not occur at all. The two methods are meant to simulate two drastically different pattern types for each of the three possible $D_{Pattern}$ distributions. Examples of the two values of $D_{PatternType}$ used in the experiment are shown in Figure 11.

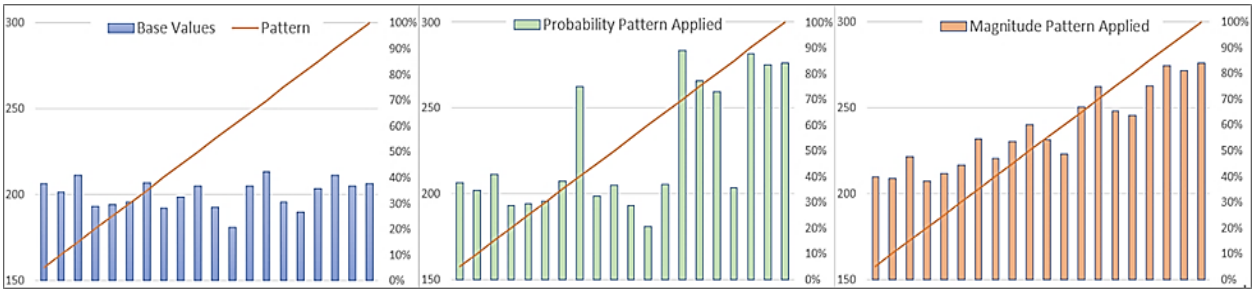

Figure 11 - $D_{PatternType}$ Sample

Figure 12 shows a sample *GT* generated using the parameters specified in Table 7. The total *GT* set contains 252 days spanning 2016-04-23 to 2016-12-30. The sine probability curve GT_{Dist} is shown as a black line in Figure 12. Because the curve determines the probability of an event being generated on each given day, a clear grouping of events forms near each peak in the curve. The GT_{Prob} of .05 results in 13 of the 252 days creating *GT* events, roughly 5.2% of all possible days. The GT_{FP} rate of .1 causes 1 of these 13 events to be classified as a false positive to be used for pattern generation but not as a positive event in testing.

Table 7 - Sample GT Parameter Values

Parameter	Description
GT_{TL}	*252 days*
GT_{Dist}	$Sin\ Period = \frac{GT_{TL}}{4}$
GT_{Prob}	.05
GT_{FP}	.1

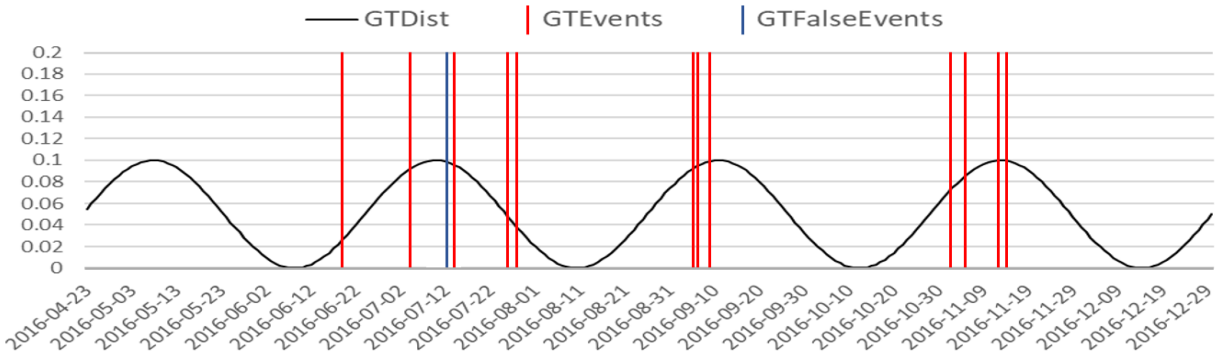

Figure 12 - Sample Event Distribution using GTProb = .05 & GTDist = Sine

Figure 13 shows another sample *GT* generated using the parameters shown in Table 7 but using a uniform GT_{Dist} instead of sinusoidal. As shown by the event spacing in Figure 13, altering the GT_{Dist} value to uniform removes the clustering of events and creates more even spacing between event occurrences.

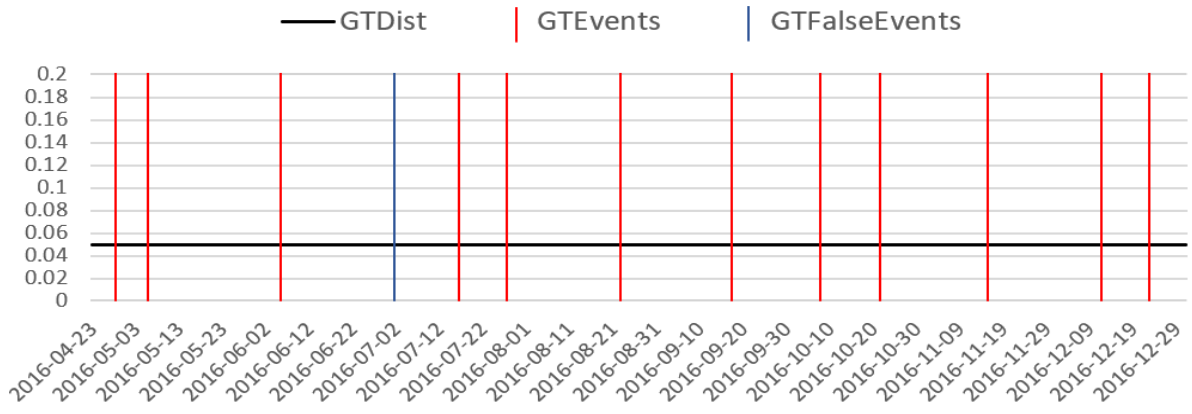

Figure 13 - Sample Event Distribution using GTProb = .05 & GTDist = Uniform

Figure 14 shows the raw values of a D_l set used with the *GT* generated in Figure 12. The data set is generated using the G_{TL} and D_{Dist} values shown in Table 8. The G_{TL} value directly determines the number of data points contained in *D*.

Table 8 - Sample D Parameters

Parameter	Description
D_{Dist}	Uniform
$D_{Pattern}$	Upward
$D_{PatternType}$	Probability
D_{Amp}	1
D_{Dur}	50
D_{LT}	1
D_{Rand}	.25

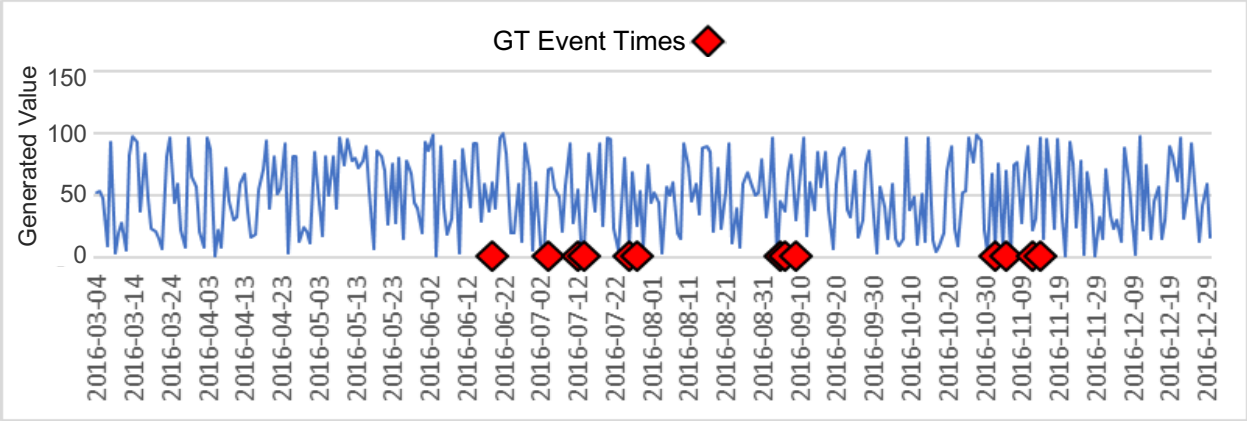

Figure 14 - Generated D Raw Values

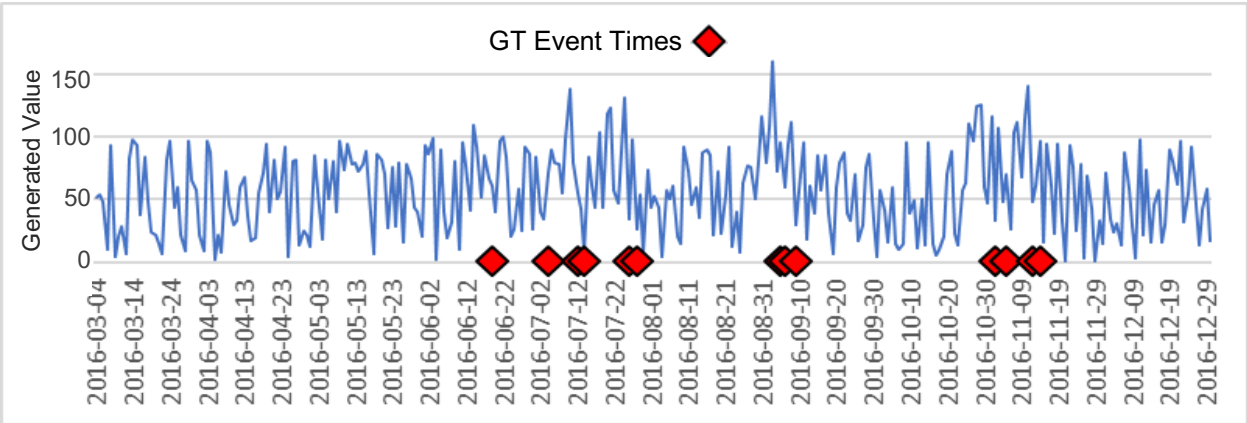

Figure 15 - Generated D with Pattern Applied

The D_{Dist} specified creates the Uniform distribution of values between [0,100] seen in Figure 14 with no discernable pattern. These data points then have the pattern defined by $D_{Pattern}$, D_{Amp}, D_{Dur}, D_{LT}, and D_{Rand} applied. As defined in Table 8, a $D_{Pattern}$ value of Upward and $D_{PatternType}$ of Probability result in a pattern of increasing spike likelihood until the provided event time. The pattern is defined to be 50 data points long, with a spike amplitude of 1 standard deviation, a lead time of 1 day, and a 25% chance of any given spike within the pattern not occurring. The *D* set shown in Figure 15 has the pattern described applied, seen as higher values prior to *GT* events including the negative event.

GT/D pairs generated using the method shown above determine exactly how well the method performs when provided with different patterns and base data sets. The results of the testing performed on these data sets were used to draw final conclusions on the abilities and limitations of the transformation process.

4.2.1.2. Feature Screening Experiment

A fractional-factorial screening experiment meant to capture all primary as well as 2 and 3 factor interactions was conducted with the intent of filtering out insignificant data generation parameters for removal from further testing. In addition to removing insignificant factors, the remaining factors were used to create the landmarking set used in the landmarking system testing. The factors and levels examined in the experiment are shown in Table 9.

Table 9 - Screening Generation Parameters

Factor	Number of Levels	Low	High
Ground Truth Characteristics			
GT_{TL}	2	600	1200
GT_{Dist}	2	Sinusoidal	Uniform
GT_{Prob}	2	2%	10%
GT_{FP}	2	0%	10%
Data Stream/Pattern Characteristics			
D_{Amp}	2	2	4
D_{Dur}	2	5	20
D_{LT}	2	1	5
D_{Rand}	2	0%	5%
D_{Dist}	2	Sinusoidal	Uniform
$D_{Pattern}$	3	Downward Upward	V
$D_{PatternType}$	2	Magnitude	Probability

For each of these parameters, a statistically significant effect resulting in an average effect of >5% change on F-Score, precision or recall of the generated classifiers resulted in consideration for landmarking and included in the final experiment. Because $D_{Pattern}$ was the only parameter tested at 3 values, the effect of varying between any 2 of these levels on the F-Score of the overall process was examined. The transformation parameter values tested in the screening experiment are shown in Table 10.

Table 10 - Screening Transformation Parameters

Parameter	Experiment Values
TailLength T_L	10, 20, 30
BinSize *A*	1, 2, 3, 4, 5, 6, 10, 15, 30
LeadTime *N*	1, 5
Sensitivity *s*	0.0 (No Spike Transform), 1.0, 2.0, 3.0
Classifier Type *type*	Tree, Bayes, KNN, SVM

The range of transformation values such as Tail Length and Lead Time are meant to provide the process with the capability to correctly capture all generated predictive patterns. Bin Size values are determined to test every possible bin size at each tail length and allow for generation of results under both total aggregation and raw value conditions. Sensitivity is currently the only transformation parameter with an unclear range of effective values, and what interval size is appropriate, so test values were selected to test the widest range currently believed to be applicable and were confirmed in later experiments.

A standard t-test was used to generate a p-value which determined the statistical significance of each parameter's effect on system-level performance. A Pearson's correlation coefficient was calculated to measure the strength of the correlation between the parameter and process response. The results of the experiment will provide the P-Value and estimated coefficient examining the relationship between each data generation parameter and the average F-Score of the overall system. The experiment utilized 3,084 unique data stream/ground truth pairs to capture each combination, resulting in a total of ~1,500,000 unique test results after testing all feasible transformation parameter sets.

4.2.1.3. Initial Transformation Method Testing

Once the total enumeration required for the screening experiment was complete, an initial observation on the value of the transformation method was made. A brief analysis was conducted to determine how the transformation method performed in terms of production of the top performing transformation/classifier set when compared to the baseline method as well as aggregation of the total training set into a single value. The relative performance of the

transformation method was assessed to validate the performance gains shown so far before continuing to the limit experiment.

4.2.1.4. Feature Limit Experiment

The purpose of the limit experiment, following successful completion of the screening experiment, was to determine appropriate test values for the generation parameters being tracked through the landmarking features. Factors found to have a statistically insignificant effect on the performance of the generated classifiers were held constant while those found to have a significant 2-factor interaction with the parameter being tested were varied for that particular experiment. Each of the parameters being tracked through landmarking were tested at a wide range of values until the effect the parameter has on the F-Score of the overall system became stable. Test values were then determined which capture the parameter at values which have a known, unchanging effect on performance. The effect of a higher or lower value of the data generation parameter can be inferred based on the slope equation of the nearest value tested.

In addition to determining the appropriate limit for landmark associated generation parameters, limit testing was also conducted for the sensitivity (s) transformation parameter. Using the data generated for the screening experiment, the effect of sensitivity (s) on the average F-Score of the process was examined to determine what range of values to use for the further experiments.

4.2.2. Transformation Method Testing

Classifier performance was calculated based on the algorithm's ability to correctly classify each unique time interval provided in GT_{Test} as either positive for containing an event or negative for

not containing an event. The performance of the method was compared to baseline performance metrics.

The baseline metric was provided by testing against the method run using the transformation parameters contained in Table 11. The baseline using these metrics determined whether the transformation process yields any noticeable difference over using the raw data value set in terms of classifier performance.

Table 11- Raw Values Transformation Parameters

Parameter	Value
s	*No spike transformation is performed when generating baseline*
$\boldsymbol{T_L}$	*Max T_L included in the transformation parameters tested*
A	1

The baseline metric was compared to the best result yielded by a total enumeration of all transformation parameters tested with the transformation system. The testing range of *s* for total enumeration testing was [0,2] to cover a range converting ~50% to ~2.3% of instances to spikes, assuming the values of *D* follow a normal distribution. The hypothesis was that converting less than ~2.3% of instances to spikes would not provide sufficient dissimilarity of B_i's within binned spike profiles for training of a classifier. The range of test values used for T_L were determined using the equation:

$$Test\ Set\ T_L = \begin{cases} [0{,}50]\ if\ GT_{Train,1} - D_{1,1} - N\ \geq 50 \\ [0, GT_{Train,1} - D_{1,1} - N] \quad otherwise \end{cases} \tag{8}$$

where *N* is the lead time value being tested. The test set for *A* at each value T_L contained all factors of T_L in order to maximize the range of the test set while retaining useable values for *A*. The total enumeration test produced the highest performing classifier possible within the given range of

values and showed how far from maximum performance the final result selected by the landmarking system was in terms of F-Score.

After each competed iteration of classification using the proposed transformation process and a specific set of transformation parameter values the responses were recorded for comparison against later transformation and classifier combinations. Once every specified transformation/classifier combination was complete, the highest performing combination was returned as output along with a set of performance metrics.

One metric used to track the performance of each classifier was precision, shown in equation 9. Precision is a measure of the number of accurate event predictions made compared to the total number of predictions made, correct or incorrect. The other metric used was recall, shown in equation 10. Recall is a measure of the number of correct event predictions made weighed by the total number of events, both predicted and missed by the classifier. Classifier performance was primarily determined based on the harmonic mean of precision and recall, referred to as the F-Score and shown in equation 11. The F-Score provides a means of combining two of the most informative metrics when comparing classifier performance.

Terminology

True Positive: Event correctly predicted **False Positive:** Event predicted when none occurred

False Negative: Missed event prediction **True Negative:** Predicted "no event" correctly

$$precision = \frac{\#\ of\ true\ positives}{\#\ of\ true\ positives + \#\ of\ false\ positives}, \quad (9)$$

$$recall = \frac{\#\ of\ true\ positives}{\#\ of\ true\ positives + \#\ of\ false\ negatives}, \quad (10)$$

$$F - Score = 2 \times \frac{precision \times recall}{precision + recall}. \quad (11)$$

The testing to generate an F-Score was based on an 70-30 split of the data set into training and testing sets based on occurrence of Ground Truth events. The training set contained all time intervals up to the point at which 70% of ground truth events have occurred, which ensures a sufficient number of events to be trained and tested on. Testing occurs beginning with the first time-interval contained in GT_{Test} for which the transformation process creates a prediction of positive or negative for if the time-interval contains an event. The process was then repeated for each following point in GT_{Test}. At the end of the testing, the performance of the process was calculated based on average performance over the entirety of GT_{Test}.

The results of the testing include any statistically significant findings regarding the performance of the best transformation method versus the baseline method. The final conclusion of the research also determined whether spike conversion and binning methods are capable of maintaining or enhancing the predictive value of a time-series data set and what future research could be conducted with regards to improvement of the final process.

4.2.3. Real Data Validation Experiment

In order to demonstrate that the benefits of the proposed transformation process carry over to real-world application the method was tested on a validation data set. Although the performance of the

validation test did not conclusively prove or disprove the value of the method, the experiment provided a means of testing some of the conclusions produced from the testing of the method on generated data.

The data stream (*D*) used for the experiment was an aggregated daily count of tweets containing both the name of a specific entity, referred to as KNOX, and at least one malicious keyword. The malicious keywords used to create the set came from a collection of 50 words determined to have significant correlation with cyber-attacks within 2016. The data set *D* contained counts for all dates from 3/3/2016 to 12/31/2016.

The ground truth (*GT*) used in the experiment was a record of DDoS attack occurrence dates against a known target. The *GT* file was split following the 70-30 guideline to provide 14 training attack and 5 test attack dates. The ground truth file contained attack dates ranging from 5-5-2016 to 12-29-2016.

The classifiers used in the test were a C4.5 decision tree and a support vector machine (SVM). Both algorithms were trained using their default parameters as defined in the WEKA package. No alterations were made to the parameters of these classifiers as the experiment required consistency across all runs.

Baseline results were generated using the parameter values specified in Table 11 in order to determine performance without the application of any transformation or feature reduction methods

or using total aggregation. The parameter values tested in the transformation runs are shown in Table 12.

Table 12 - Test Parameter Values

Parameter	Values
s	*.5, 1*
T_L	*10, 20, 30, 40*
N	*1, 3, 5*
A	*2, 5, 10*

The results gathered from the experiment include the F-Score, Precision, and Recall of the baseline transformation set as well as the performance of these methods at each combination of transformation parameters contained in Table 12. Based on the results, any performance increase gained through the application of the transformation process were made apparent and were compared against the baseline results.

4.3. Landmarking System

The strategy that the proposed system uses for selecting a high performing set of transformation parameters involves generating landmarks. Landmarks are the landmarking system values of each *D/GT* set on which the system has been tested. The goal of recording landmarks is to build a knowledge base through which the landmarking system may determine transformation values which were effective on a *D/GT* pair similar to a new *D/GT* which the system has not tested prior. The hypothesis was that utilizing a transformation parameter set that was found to be useful on similar data would yield higher than average performance when compared against all transformation parameter sets tested. Selecting a classifier based on performance on a similar data set has been shown to be effective for parameter selection for classifiers (Reif et al., 2011;Syarif, Prugel-Bennett, & Wills, 2016). If the system performed as expected, a high performing

transformation for a given *D/GT* could be determined while testing only a small portion of the total possible transformations that could be applied.

4.3.1. Identifying Landmark System Features

The meta features used to differentiate *D*/*GT* combinations, or landmarking features, were determined based on both their ability to be calculated using the *D*/*GT* pair without requiring extensive preprocessing as well as their correlation with the F-Score and values of transformation parameters. Once the screening experiment was completed, the generation parameters which had the greatest effect on the F-Score of the system were considered as candidates for being landmarked using a landmarking parameter. The generation parameters selected were then reduced to only those generation parameters which are likely to be capable of being predetermined through minimal processing. The remaining generation parameters were then tested against multiple meta-features of D_I and *GT* for strong correlation. Correlation indicated if a given meta-feature could potentially be used to estimate the generation parameter value of a provided *D* or *GT*. The meta-features with the strongest correlation were then incorporated into the system as landmarking features.

For each landmarking feature selected through the process, a correlation with the F-Score verified that variation of the landmarking feature affects system performance in the same way as the associated generation parameter. A correlation with the best transformation parameter set showed that the landmarking feature affects what set of transformations produce the highest performing result from the given *D*/GT. These tests determined whether using these landmarking feature values to search for a nearest landmark yields a high performing transformation to apply to a *D/GT*.

4.3.2. Landmarking System Functionality

A landmark is defined as a combination of a set of landmarking feature values associated with a specific *D/GT* pair. Each landmark is stored along with every transformation set tested, as well as the performance achieved with each transformation set, in the knowledge base of the landmarking system. Once a set of transformation parameters have been found effective when applied to a *D/GT* pair yielding a particular set of landmarking feature values, the same transformation set is expected to yield comparable performance when tested on *D*/*GT* combinations with similar landmarking feature values.

The closest landmark to the provided *D/GT* pair is determined by a Linear Nearest Neighbors search algorithm. The algorithm determines the closest match based on the linear distance of the values of each feature provided. The values of the landmarking features of the knowledge base as well as the *D/GT* being tested are normalized from [0,1] to ensure equal weighting of landmarking values by the KNN algorithm. A sample landmarking search, with the resulting landmark match underlined and the corresponding set of top performing transformation values, is shown in Table 13. In the example, the transformation parameters of *Test Instance* have been matched to the nearest landmark. An overview of the landmarking system is shown in Figure 16.

Table 13 - Sample KNearestNeighbors Landmark Search

	Landmarking Features			**Transformation Parameters**		
	L_1	L_2	L_3	T_1	T_2	T_3
Test Instance	5	2	1	<u>5</u>	<u>4</u>	<u>3</u>
Landmark 1	2	4	5	2	1	2
Landmark 2	<u>5</u>	<u>1</u>	<u>1</u>	<u>5</u>	<u>4</u>	<u>3</u>

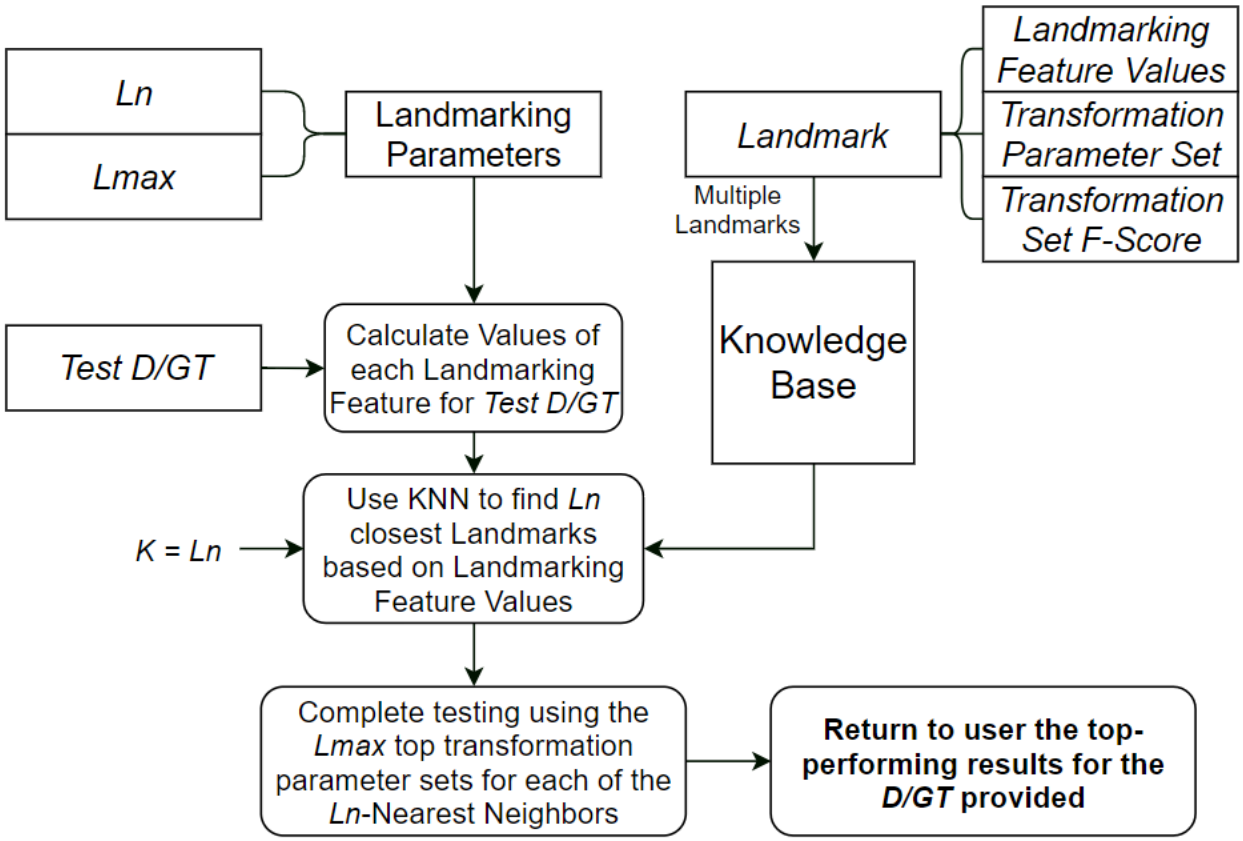

Figure 16 - Landmarking System Overview

For each set of landmarking feature values, there may be multiple landmarks each paired with their unique transformation parameters and the resulting F-Score of using that particular transformation set. Using the data provided, the system may find a nearest neighbor landmark and calculate which transformation parameter values resulted in the highest F-Score when applied. A sample knowledge base containing 2 unique landmarking feature sets with 2 transformation parameter sets each is shown in Table 14.

Table 14 - Sample Landmarking Knowledge Base

Landmarking Features			**Transformation Parameters**			**Response**
L_1	L_2	L_3	T_1	T_2	T_3	*F-Score*
2	4	5	2	1	2	.80
2	4	5	3	3	1	.58
5	1	1	6	3	2	.79
5	1	1	4	2	2	.54

The total number of landmarks utilized by the system is specified as L_N. The number of top performing transformation combination sets tested for each nearest landmark is determined by the parameter $L_{Max.}$ The total of transformation sets the landmarking system tests in a single run is given by the equation:

$$Total\ Transforation\ Sets\ Tested = L_N \times L_{Max} \quad (12)$$

For example, using the landmarking parameter values given in Table 15 would result in the landmarking system determining the three closest landmarks based on landmarking feature values to the *D/GT* provided and testing both the transformation set with the highest F-Score as well as the transformation set with the second highest F-Score for each closest landmark.

Table 15 - Sample Landmarking System Parameter Values

Parameter	Value
L_N	*3*
L_{Max}	*2*

4.3.3. Landmarking System Performance

The distance from the best transformation set's performance for a given landmark prediction may be calculated based on performance percentile when compared with the F-Score of all other transformation sets tested on the same *D/GT*. The response of the landmarking system measured for performance is the F-Score percentile of each *D/GT* pair tested. In order to calculate the response, the performance of all transformation sets previously tested on the *D/GT* pair associated with a given set of landmarking feature values must be examined. The F-Score percentile for a test run $L_{D/GT}$ may be calculated using equations 13 and 14.

Terminology:

$\boldsymbol{F_{T,D,GT}}$: $F\ Score\ of\ transformation\ set\ for\ D, GT\ pair$

$\boldsymbol{F_{Landmark}}$: $F\ Score\ of\ landmark\ for\ each\ D, GT\ pair$

$$F_{compare,T,Landmark} = \begin{cases} 1\ if\ F_{Landmark} > F_{T,D,GT}\ \forall\ T \\ 0\ otherwise \end{cases} \quad (13)$$

$$L_{F-Score\ Percentile,D,GT} = \frac{\sum_{\forall\ T} F_{compare,T,Landmark}}{T_{Count}} \quad (14)$$

where T is a set of transformation parameters and T_{Count} is the total number of test runs conducted using a given T. The F-Score Percentile is calculated based on the F-Scores of all runs previously conducted using the same *D/GT* pair. After running the landmarking system, the highest resulting F-Score from all landmark transformation sets tested is returned along with the F-Score percentile when compared to every transformation set previously tested on the same *D/GT*.

4.3.4. Landmarking System Experiment

Descriptions of the features L_N, L_{Max}, and L_{Split} tested in order to measure their effect on landmarking system performance are provided in Table 16.

Table 16 - Landmark System Parameters

Parameter	Description
$\boldsymbol{L_N}$	Number of unique landmark transformation sets tested
$\boldsymbol{L_{Max}}$	Number of transformation sets tested for each closest landmark
$\boldsymbol{L_{Split}}$	Fraction of generated data used for knowledge base generation

The set of parameters associated with the landmarking system allow for an assessment of how factors related to the number of runs conducted and amount of prior knowledge provided affect the performance of the system.

The number of transformation parameter sets to be tested is defined by the parameter L_N. The effect of varying the value of L_N was examined through the experiment. Determining an effective L_N based on a provided data set allows the system to identify whether the potential exists within a data set to provide predictive value early on and prevent frivolous testing from continuing once a certain confidence of the classifier's potential performance is established.

Using the results of the Screening and Limit experiments, a final set of *D/GT* pairs was generated for the purpose of testing the proposed landmarking features. The purpose of the final training set was to test the effectiveness of the landmarking system when trained using the proposed landmarking features. Once generated, all *D/GT* pairs were tested using a range of transformation parameters in order to build a base performance reference from which to calculate F-Score Percentile for each test run. Testing was completed using a wide enough range of transformation values to capture any of the patterns to be generated in the data sets based on their duration and lead time.

Upon completion of data generation and performance testing, the landmarking features were tested to ensure their significant correlation with the transformation parameters used. The experiment clarified whether varying the landmarking parameters to the values determined through the limit

experiment had a significant effect on the highest performing transformation set for a given *D/GT* pair.

The generated data was separated into a training set, with which the knowledge base of the landmarking system was generated, and a testing set. The percentage of the generated data used for training is defined by the parameter L_{Split}. L_{Split} was used in the experiment to determine the portion of the generated data to be used as a knowledge base. Ordinarily the landmarking system would be trained on all relevant results available to the user.

The splitting of the total generated data sets was based on unique combinations of landmarking features, ensuring that no data streams in the testing set would have corresponding matches in the knowledge base with identical landmark features. Preventing results generated using the same *D* parameters, *GT* parameters, or both from being split into the train and test sets was important to ensuring that an exact match between landmarking features did not occur. In a real application, the chance of the provided landmarking features matching exactly would be highly improbable. A landmark retrieved from the exact *D/GT* of the testing instance would likely provide unusually high performance, and would likely simply match with the highest performance transformation set known to the system for the particular *D/GT*.

5. Results and Discussion

Section 5 shows the results of the Screening, Limit and Landmarking Experiment as detailed in the Methodology. The resulting information gained from conducting these experiments and analyzing their results was sufficient to determine which features were appropriate to use as landmarks and at what levels they should be tested to fully understand their effect on performance of the transformation process. The final section analyzes the performance of the landmarking system and the effects of the associated parameters.

5.1. Screening Experiment

The purpose of the screening experiment was to determine which factors have a significant effect on the F-Score, precision and recall of the trained classifiers when a specified set of transformation values are used. The primary effects detected through the experiment are shown in Table 17. The effects of each parameter are significant with a > 99% certainty on the F-Score, precision, and recall with the exception of GT_{Dist} on precision. However, the effects of many of these parameters on each response variable are of insignificant magnitude, such as $D_{Pattern}$ causing an average change of less than 1% when varied from a value of *upward* to *v*. Therefore, in addition to screening out the parameter changes which have no statistically significant impact, any parameters which fail to result in a change of >5% in any response variable, shown underlined in Table 17, were deemed insignificant.

Table 17 - Screening Generation Parameters Results

			F-Score		***Precision***		***Recall***	
Factor	**Low**	**High**	**P-Val**	**Avg. Effect**	**P-Val**	**Avg. Effect**	**P-Val**	**Avg. Effect**
Ground Truth Characteristics								
GT_{TL}	600	1200	0.00	1.98	0.00	1.62	0.00	8.10
GT_{Dist}	Sinusoidal	Uniform	0.00	5.05	0.60	-0.01	0.00	1.78
GT_{Prob}	2%	10%	0.00	14.80	0.00	5.45	0.00	-17.16
GT_{FP}	0%	10%	0.00	-5.58	0.00	-1.33	0.00	-2.83
Data Stream/Pattern Characteristics								
D_{Amp}	2	4	0.00	8.74	0.00	2.22	0.00	5.10
D_{Dur}	5	20	0.00	-0.47	0.00	-2.52	0.00	-5.31
D_{LT}	1	5	0.00	4.45	0.00	2.90	0.00	9.31
D_{Rand}	0%	5%	0.00	-1.94	0.00	-0.70	0.00	-2.87
D_{Dist}	Sinusoidal	Uniform	0.00	5.05	0.00	-0.31	0.00	1.78
$D_{Pattern}$	Down	Up	0.00	-1.13	0.00	-1.17	0.00	-2.13
$D_{Pattern}$	Down	V	0.00	-1.08	0.00	-0.92	0.00	-2.78
$D_{Pattern}$	Up	V	0.00	0.05	0.00	0.25	0.00	-0.35
$D_{PatternType}$	Magnitude	Probability	0.00	1.07	0.00	-1.86	0.00	-0.63

D_{Rand}, $D_{Pattern}$, and $D_{PatternType}$ were the 3 factors found through the analysis to have no significant impact on the F-Score of the overall system and were therefore removed from consideration as potential parameters for creating landmarking features. Although GT_{FP} does have a significant single factor interaction with average F-Score, the negative impact has little impact on the other parameters. Generating negative pattern instances was hypothesized and confirmed to negatively

impact performance. The factor however holds no significant value warranting further testing. GT_{FP} will therefore not be considered for the creation of a landmarking feature.

In addition to testing each factor's individual impact on the performance of the transformation method, the effects on the average F-Score caused by 2-factor interaction of the parameters were recorded and are shown in Table 18. The purpose of analyzing the 2-factor interaction was that any factor which significantly interacts with a factor to be varied must also be varied to accurately capture performance effects on the process.

Table 18 - 2-Factor Interaction of Generation Parameters

	GT Tail	GT Prob	GT False Prob	D Amp	D Lead	D Rand Prob	D Duration	D Dist - Uni	D Pattern Type - Prob	D Pattern - Up	D Pattern - V	GT Dist - Uni
GT Tail		-0.73%	1.57%	1.68%	0.00%	-0.04%	0.00%	-0.36%	-0.36%	-0.42%	-0.42%	1.14%
GT Prob	-0.73%		0.09%	-1.27%	-0.47%	0.12%	-5.58%	-0.44%	-1.86%	-1.45%	-0.67%	-0.55%
GT False Prob	1.57%	0.09%		1.24%	-1.03%	-0.99%	0.80%	0.40%	0.04%	0.35%	0.24%	-0.81%
D Amp	1.68%	-1.27%	1.24%		3.64%	-0.03%	-4.94%	0.21%	-0.85%	-0.88%	-0.34%	-0.79%
D Lead	0.00%	-0.47%	-1.03%	3.64%		0.23%	-5.78%	-0.27%	-0.65%	1.21%	1.04%	-0.87%
D Rand Prob	-0.04%	0.12%	-0.99%	-0.03%	0.23%		0.04%	0.01%	0.29%	-0.01%	0.04%	0.06%
D Duration	0.00%	-5.58%	0.80%	-4.94%	-5.78%	0.04%		0.01%	-0.95%	4.87%	2.64%	-1.77%
D Dist - Uni	-0.36%	-0.44%	0.40%	0.21%	-0.27%	0.01%	0.01%		-0.10%	0.19%	0.12%	-0.03%
D Pattern Type - Prob	-0.36%	-1.86%	0.04%	-0.85%	-0.65%	0.29%	-0.95%	-0.10%		0.83%	0.19%	0.03%
D Pattern - Up	-0.42%	-1.45%	0.35%	-0.88%	1.21%	-0.01%	4.87%	0.19%	0.83%			-0.10%
D Pattern - V	-0.42%	-0.67%	0.24%	-0.34%	1.04%	0.04%	2.64%	0.12%	0.19%			-0.06%
GT Dist - Uni	1.14%	-0.55%	-0.81%	-0.79%	-0.87%	0.06%	-1.77%	-0.03%	0.03%	-0.10%	-0.06%	

All factors above which have a 2-factor interaction >4%, including GT_{Prob}, $D_{Duration}$, D_{Amp}, D_{Lead}, and $D_{Pattern}$, were varied for testing of the parameters with which they have a significant interaction. Factors lacking a significant interaction may still be varied for the purpose of creating additional replications for each generation parameter set.

5.2. Initial Transformation Method Performance

Based on the results of the total enumeration testing, the limits and effects of all parameters of the transformation process are better understood. In terms of specific transformation values, the transformation method seems to excel in less noisy data sets with fewer training instances. When D_l begins to have a higher number of spikes naturally occurring in the data set, as determined by $D_{Dist} = Uni$, the highest performing binning transformation requires far smaller bins resulting in a training set more similar to the raw values provided. Additionally, the smaller the number of data points in D and fewer event occurrences the more likely an s level >0 is to yield the highest performing transformation set. These results suggest that the less data available for classifier training and the lower the noise and pattern amplitude of the data set the more likely the transformation process is to outperform the raw values.

Figure 17 shows for what percentage of all *D/GT* pairs tested each of the 3 methods examined produced the best performing transformation/classifier combination.

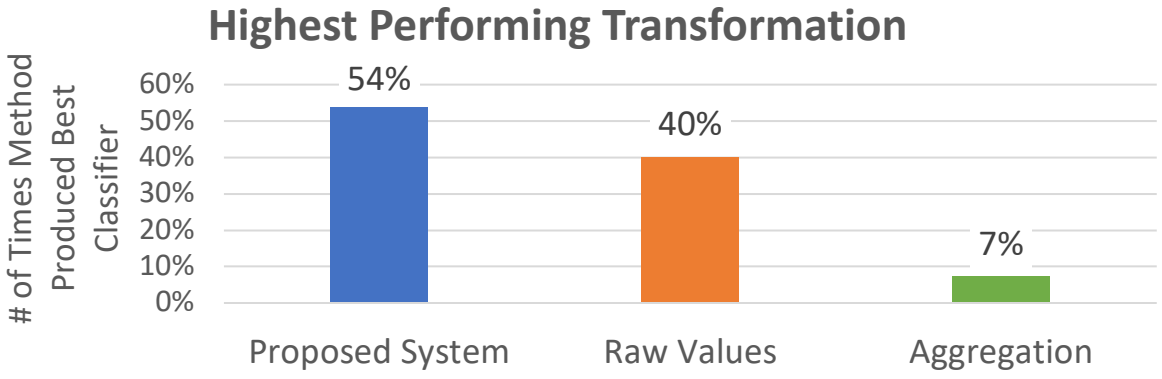

Figure 17 - Total Enumeration Test - Best Method

Raw values represent when a bin size of 1 and no spike transformation were used. Aggregation represents when the tail length (T_L) selected was equal to the bin size (A). The transformation method being tested produced the highest performing result 14% more often than the raw values and 47% more frequently than aggregation.

The improvement gained by utilizing the transformation method in the cases where the method produced the highest performing transformation is shown in Figure 18. In cases were the transformation method yielded the highest performing transformation, the improvement gained was greater than 50% over the F-Score produced by the raw values and aggregation feature sets.

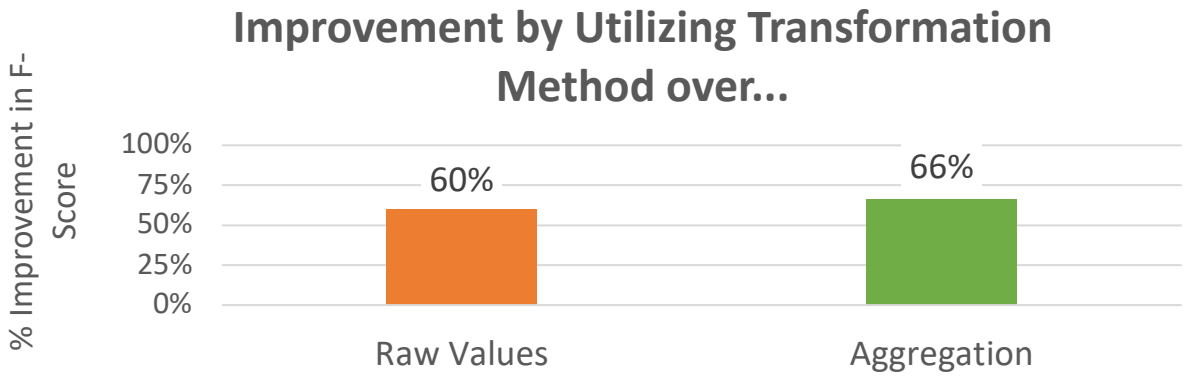

Figure 18 - F-Score Improvement from use of Transformation Method

The results of the observation suggest that the proposed transformation method produces a higher performing result than either training on the raw values or the aggregated value of the training set for at least 50% of the data sets provided. The results also show that when the transformation method produces the best performing feature set, it is by a wide margin of performance.

5.3. Landmark Selection

Table 19 shows the results of the screening experiment for features which are predeterminable for the system. As detailed in the methodology, these features are capable of having an accurate value

calculated with minimal processing of the provided data stream and ground truth. In addition to being predeterminable, each of these parameters has a > 5% effect on at least one response of the system, indicating that a change in the value of each parameter may be used to predict a significant change in system output.

Table 19 - Predeterminable Features

Term	P-Value	Response	Effect %
$\boldsymbol{GT_{TL}}$	0.00	Recall	8.10%
$\boldsymbol{GT_{Dist}}$	0.00	F-Score	5.05%
$\boldsymbol{GT_{Prob}}$	0.00	F-Score	14.8%
$\boldsymbol{D_{Amp}}$	0.00	F-Score	8.74%
$\boldsymbol{D_{Dist}}$	0.00	F-Score	5.05%

Some of these parameters, such as GT_{TL} and GT_{Prob}, can be calculated directly from the provided data stream and ground truth. The parameters GT_{Dist}, D_{Dist}, and D_{Amp}, required the development of a correlated landmarking feature because they cannot be easily extracted from the data sets without significant processing or prior knowledge. To attempt to predict the values of these parameters, characteristics of *D* and *GT* were extracted and tested for correlation with the original parameters. For testing of each proposed landmarking feature, 24 *D* or *GT* files were created, with 12 at each level of the parameter with which the landmark feature is being tested.

The characteristics to be tested for correlation with each of the listed key parameters are coefficient of variation (*equation 15*), spikiness (*equation 16*), and NRange (*equation 17*). Coefficient of variation is calculated for each data set as:

$$coefficient\ of\ variation(D) = \frac{\sigma(D)}{\mu(D)} \qquad (15)$$

The use of the coefficient of variation allows for a metric reporting the ratio of the standard deviation to the mean, which is inherently normalized across data sets consisting of different average values. The measure is expected to accurately display the "noisiness" of the data set, which was hypothesized to correlate well with all parameters being investigated.

Spikiness is a measure of the number and severity of the outliers within a data set. Spikiness is meant to capture how drastic the outliers in a data set are and weigh data points which fall a substantial distance from the mean heavily. Spikiness is calculated as:

$$P_i = \begin{cases} 1 \; if D_i > a \times \sigma_D + \mu_D \\ 0 \; otherwise, \end{cases}$$

$$spikiness(D) = \frac{\sum_{a=1}^{4} a \times P_i}{\#\ of\ data\ points(D)} \qquad (16)$$

Equation 16 multiplies the number of data points above a given threshold by the number of standard deviations that they lie above the mean of the data set.

NRange was used as a normalized method of capturing the range of each data set. The equation for NRange is:

$$NRange(D) = \frac{max\left(D_{i,l}\right) - min\ (D_{i,l})}{mean(D_{i,l})} \qquad (17)$$

5.3.1. GT_{Dist} Landmark

In order to create a landmarking feature which accurately captures the value of GT_{Dist} without having to fit a distribution to the data points contained in *GT,* the spacings of the contained event timings were examined as a data set. Prior to testing, the assumption was that a *GT* data set's event

spacing when following a uniform distribution would have less variance than that of a *GT* set generated following a sinusoidal distribution. The system may therefore determine how clustered an event set is by examining the values of these spacings. Using coefficient of variation allows the measure to be normalized between data sets with a significantly different GT_{Prob} value. The correlation coefficients of GT_{Dist} with each potential landmark are shown in Table 20.

Table 20 - GTDist Landmark Correlation Testing

	Coeff. Var.	***Spikiness***	***NRange***
$\boldsymbol{GT_{Dist}}$	-0.87	0.49	-0.74

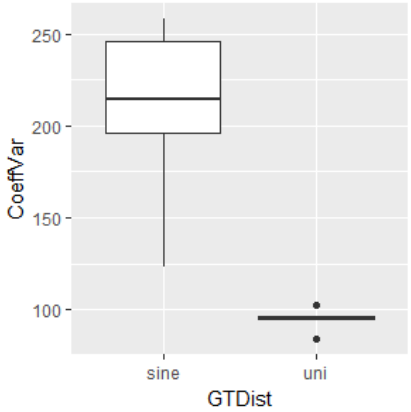

Figure 19 - GT_{Dist} Coefficient of Variation Correlation

The strongest correlation was found between GT_{Dist} and coefficient of variation. As shown in Figure 19 the difference in the value of the coefficient of variation for the 2 values of GT_{Dist} tested had no overlap. Coefficient of variation was therefore expected to provide an effective measure for determining the distribution of events within a *GT* set without having to calculate a fitted distribution.

5.3.2. D_{Dist} Landmark

Determining a landmarking feature which effectively captures the base distribution used to generate *D* relied on capturing what could be called the noisiness of the data set. A value stream

generated about a uniform distribution will have a much larger portion of data points lying >1 standard deviation away from the mean. The initial assumption was that *spikiness*, calculated using equation 16, may be effective for the task as a direct calculation of the number of data points outside of each sensitivity level, weighted towards the larger outliers. The correlation coefficients of D_{Dist} with each potential landmark are shown in Table 21.

Table 21 - DDist Landmark Correlation Testing

	Coeff. Var.	***Spikiness***	***NRange***
$\boldsymbol{D_{Dist}}$	-0.57	-0.47	-0.92

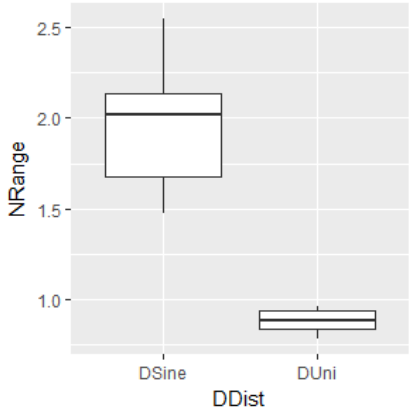

Figure 20 - D_{Dist} NRange Correlation

The correlation detected between the value of D_{Dist} and *NRange* is shown in Figure 20. With a correlation coefficient of -0.92, the two variables had a strong negative correlation and *NRange* could easily be used to accurately estimate the base distribution of *D*. Both *spikiness* and coefficient of variance also had a negative correlation with D_{Dist}, but with a much lower correlation coefficient leaving more chance of values at each level of D_{Dist} to overlap and have a less significant difference.

5.3.3. D_{Amp} Landmark

The final landmarking feature developed had to track the amplitude of the patterns in a provided data stream. The preliminary assumption was again that spikiness may capture D_{Amp} accurately as the amplitude of a pattern applied directly affects the number of points falling outside different s levels. The results of the correlation testing of D_{Amp} against the potential landmarking features are shown in Table 22.

Table 22 - D_{Amp} Landmark Correlation Testing

	Coeff. Var.	***Spikiness***	***NRange***
D_{Amp}	0.66	-0.38	0.27

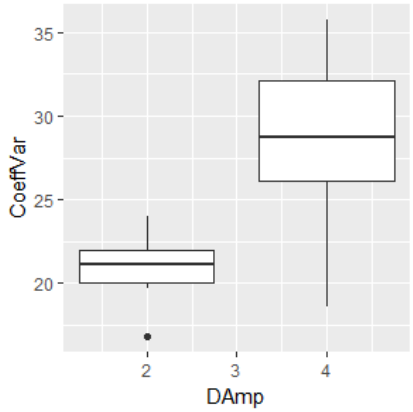

Figure 21 - D_{Amp} Coefficient of Variation Correlation

The strongest landmarking feature correlation was once again with coefficient of variation. Although not as distinct of grouping as the previous two generation parameter/landmark feature pairs, Figure 21 still shows a very clear separation between the values at each level of D_{Amp}. Although some minor overlap exists between the values of the coefficient of variation of D, the separation is statistically significant enough that the landmarking feature may be used effectively.

5.3.4. Final Landmarking Feature Selection

The features to be used in the training and testing of the landmarking system are listed in Table 23. Each of these features are either a direct calculation of a generation parameter or have shown to be strongly correlated with a generation parameter. The calculation of each landmarking feature was conducted prior to the application of any transformations in the finished system.

Table 23 - Final Landmarking Features

Gen. Parameter	Landmarking Feature	Description	Correlation Coeff.
GT_{Tail}	GT_{Tail}		1.00
GT_{Prob}	GT_{Prob}		1.00
GT_{Dist}	GT_{CoVar}	Coefficient of variation of event spacings within GT *(Equation 15)*	-0.87
D_{Dist}	D_{NRange}	Range of the max and min values of D_l divided by mean of D_l *(Equation 17)*	-0.92
D_{Amp}	D_{CoVar}	Coefficient of variation of D *(Equation 15)*	0.66

5.4. Limit Experiment

Prior to conducting a large-scale experiment to test the proposed landmarking system and features, further testing was conducted to determine the min/max values of the generation parameters which were being landmarked. The subset of experiments was conducted through varying a generation parameter over a more extensive range of values than had previously been tested in order to better understand the parameter's effect on the F-Score. With a clearer understanding of each of these variable's effects, more informed upper and lower testing limits were established where the rate of change in F-Score levels off or drops to 0. Through variation of generation parameters such as GT_{Dist}, D_{Dur}, D_{Dist}, $D_{Pattern}$, and $D_{PatternType}$ 8 different replications of generation values were tested for each of the levels of the particular variable being investigated. For example, in an experiment

testing the performance of the process on a data set generated using D_{Amp} equal to 3, 8 unique generated *D/GT* pairs would be created using that D_{Amp} value.

The data generation values used for level testing are shown in Table 24 and were used for generation of all data sets for the experiment unless otherwise specified. The transformation values which remained fixed across all limit testing iterations are shown in Table 25. Maintaining equal values of N and D_{LT}, the lead time of the pattern applied, ensured that the transformation method always correctly captured the window of time over which predictive patterns occurred. The same reasoning applied to the values of T_L and D_{LT}, as maintaining $T_L \geq D_{LT}$ guaranteed that the full pattern was captured. A remains varied over a set of feasible values based on the fixed T_L. GT_{FP} and D_{Rand} were maintained at a value of 0 as they had no substantial 2-factor interaction with any of the features being tested in the experiment. In the case of any 2-factor interactions, both levels of the factor which had a significant interaction with the generation parameter being tested were run. Running both levels of any significant interactions ensured that the results of the experiment were representative of the effect of the generation parameter independent of other generation or transformation parameters.

Table 24 - Limit Experiment Generation Parameters

Parameter	Fixed Values
GT_{TL}	3000
GT_{Dist}	Uni, Sine
GT_{Prob}	.1
GT_{FP}	0
D_{Amp}	4
D_{Dur}	5, 20
D_{LT}	5
D_{Rand}	0
D_{Dist}	Uni, Sine
$D_{Pattern}$	Down, V
$D_{PatternType}$	Mag, Prob

Table 25 - Limit Experiment Transformation Parameters

Parameter	Fixed Values
TailLength T_L	20
BinSize A	1, 5, 10, 20
LeadTime N	5

5.4.1. GT_{TL}

In order to better examine the relationship between the value of GT_{TL} and the performance of the overall system, 32 test iterations were performed for each value of GT_{TL} shown in Table 26. For the experiment, the levels of *s* and *classifier type* were altered to reduce the necessary run time. Because sensitivity value had no significant interaction with GT_{TL}, the values tested were reduced to the min (0) and max (3) *s* values tested prior. *Classifier type* also showed no significant interaction with GT_{TL}, so the Bayes Net was kept as the only tested classifier in order to yield the least negative possible interaction with an increase in GT_{TL}.

Table 26 - Generation & Transformation Parameter Values Used for GTTL Limit Test

Parameter	Experiment Values
GT_{TL}	500, 2000, 3500, 5000, 6500, 8000, 9500
Sensitivity s	0 (No Spike Transform), 3.0
Classifier Type	Bayes

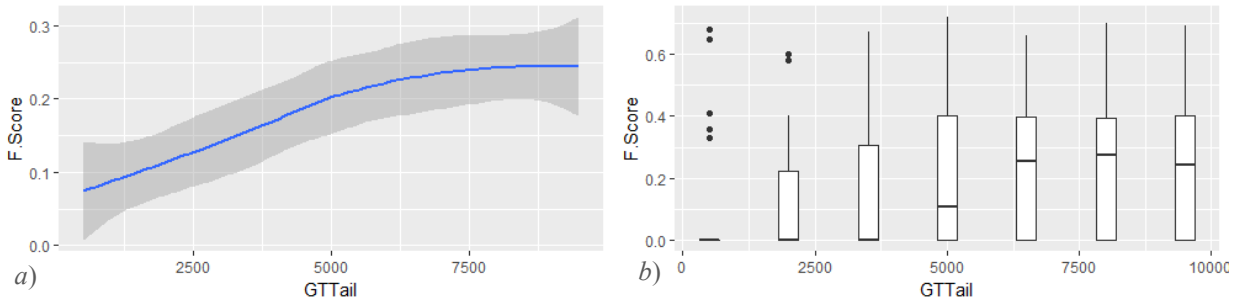

Figure 22a & b - GT_{Tail} Limit Testing Results

The experiment to determine a valid range of testing values for GT_{TL} was initially conducted using the screening experiment values and extended to larger and smaller values until a consistent slope was shown in the change in F-Score. The results of the testing are shown in Figure 22a & Figure 22b, where the average F-Score is plotted at different values of GT_{Tail}. At the rate of decrease that the F-Score experiences, the response appears to linearly approach 0 as the value of GT_{TL} approaches 0. The linear increase in average F-Score as tested seems to start tapering around 5000 and reach a slope of 0 at 7500. Following these results, the maximum limit for further testing has been set at 7500 to capture the peak of the F-Score improvement and the minimum has been at 1200 to capture a low value for GT_{Tail} prior to F-Score dropping to 0.

5.4.2. GT_{Prob}

The range of values used for limit testing on GT_{Prob} was extended to capture the multiple changes in slope that the average F-Score encountered as the parameter was varied. The final set of test values used for the experiment is shown in Table 27.

Table 27 - Generation & Transformation Parameter Values Used for GT_{Prob} Limit Test

Parameter	Experiment Values
GT_{Prob}	.005, .01, .02, .05, .1, .15, .2, .25, .3, .35
Sensitivity s	0 (No Spike Transform), 1.0, 2.0, 3.0
Classifier Type	Tree, SVM, Bayes, KNN

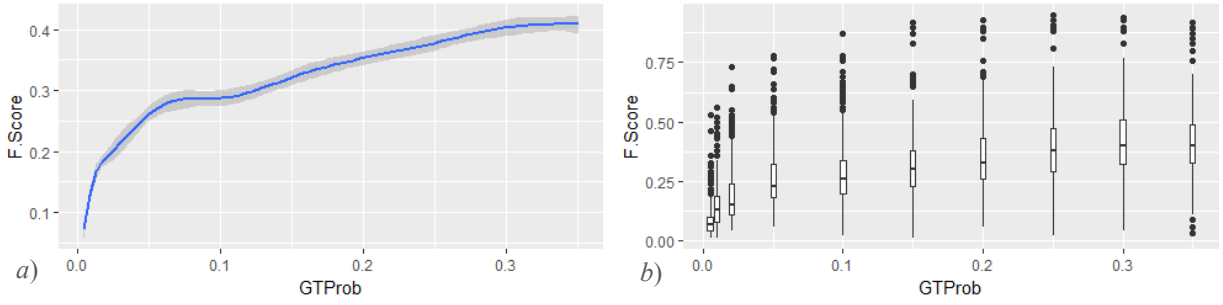

Figure 23a & b - GT_{Prob} Limit Testing Results

256 test iterations were conducted for each value of GT_{Prob} shown in Table 27. As shown in Figure 23a & Figure 23b the transformation process managed to capture events down to an average probability of less than half of a percent, at which point the likelihood of an event occurring drops to close enough to zero that the classifier begins to predict all negative occurrences. The method experiences rapid performance improvement from 1%-5%, a lack of improvement from 5%-10%, and then a linear increase to the limit of where testing ended at 30%. By 30%, the F-Score is leveling off at which point predicting all positive for events would yield roughly equivalent performance to what the process is producing. In order to best capture performance across the values of GT_{Prob} in later experiments, 3 testing values were selected for GT_{Prob}. The lowest value was 1%, the minimum value at which the produced classifiers still predict some events. The highest value tested was 7%, after which the results showed a linear increase until GT_{Prob} reaches a value of 30%, at which point there was severe overlap of predictive patterns between events. A middle value of 2% was also tested, as the results showed a linear increase in performance from GT_{Prob} values of 1% to 7% and capturing a point along the line provides better feedback as to how the process performs when provided with scarce events.

5.4.3. D_{Amp}

Range testing on D_{Amp} concluded with the narrowest range of experimental values out of the 3 generation parameters being tested. The F-Score being produced increased linearly from 0 to ~.3 rapidly but leveled off almost immediately. The leveling off observed may be due in part to the fact that once amplitude has exceeded 3 the pattern spikes being generated are all easily captured given the range of s values being tested. The D_{Amp} values tested are shown in Table 28.

Table 28 - Generation & Transformation Parameter Values Used for D_{Amp} Limit Test

Parameter	Experiment Values
D_{Amp}	1, 2, 3, 4, 5, 6
Sensitivity s	0 (No Spike Transform), 1.0, 2.0, 3.0
Classifier Type	Tree, SVM, Bayes, KNN

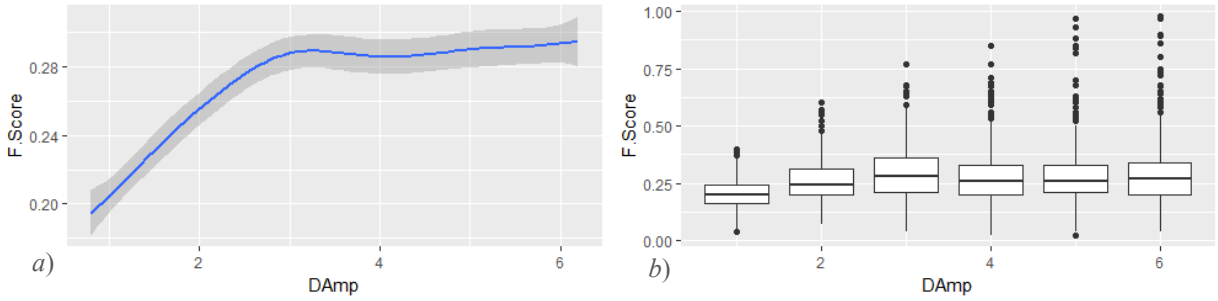

Figure 24a & b - D_{Amp} Limit Testing Results

256 test iterations were conducted for each value of D_{Amp} shown in Table 28. As shown in Figure 24a & Figure 24b after a value of 3 has been reached increasing D_{Amp} does little to increase the performance of the system. From 3 to 1 the performance of the system followed a consistent linear decrease in F-Score which is expected to drop to 0 rapidly as D_{Amp} approaches 0, as the pattern approaches the point of having no effect on the data set. Although the upper bound of s is limited which may affect the test, the assumption was that once the value of D_{Amp} surpasses an upper bound the classifier can detect the pattern equally as well without the spike transform, potentially even better. The spike transform is meant to assist to separate a pattern from noise levels of similar amplitude. Because of the limited range over which D_{Amp} affects F-Score, a maximum limit of 3 and a minimum limit of 1 were set for further testing.

5.4.4. *Sensitivity (s)*

Testing was also conducted to determine the effective upper limit of s. As proposed in 6.3.3, the current assumption was that after a certain level applying the spike transformation yields no improved performance over simply training on the raw values.

For each value of D_{Amp} tested, a subset consisting of the top 10% highest F-Scores achieved by all generation and transformation parameter sets tested was created. The F-Scores of the top performing subset for each value of D_{Amp} are shown in Figure 25. In the figure, each of the F-Scores achieved are shown as individual points, with the data points which used the spike transformation ($s > 0$) differentiated from the data points which were created using the raw data values ($s = 0$). The average and standard deviation at each value of D_{Amp} is also plotted as a solid ($s > 0$) or dotted ($s = 0$) line. The analysis was meant to examine at what point the spike transformation stops being used in the top performing transformation parameter sets tested.

Starting from the lowest value of D_{Amp} tested, the average of the top results created using the spike transformation never surpasses the average F-Score of the top results created with $s = 0$. However, at $D_{Amp} = 1$ and 2, the highest performing result does utilize the spike transformation. After $D_{Amp} = 2$ the spike transformation is no longer included in the highest performing transformation.

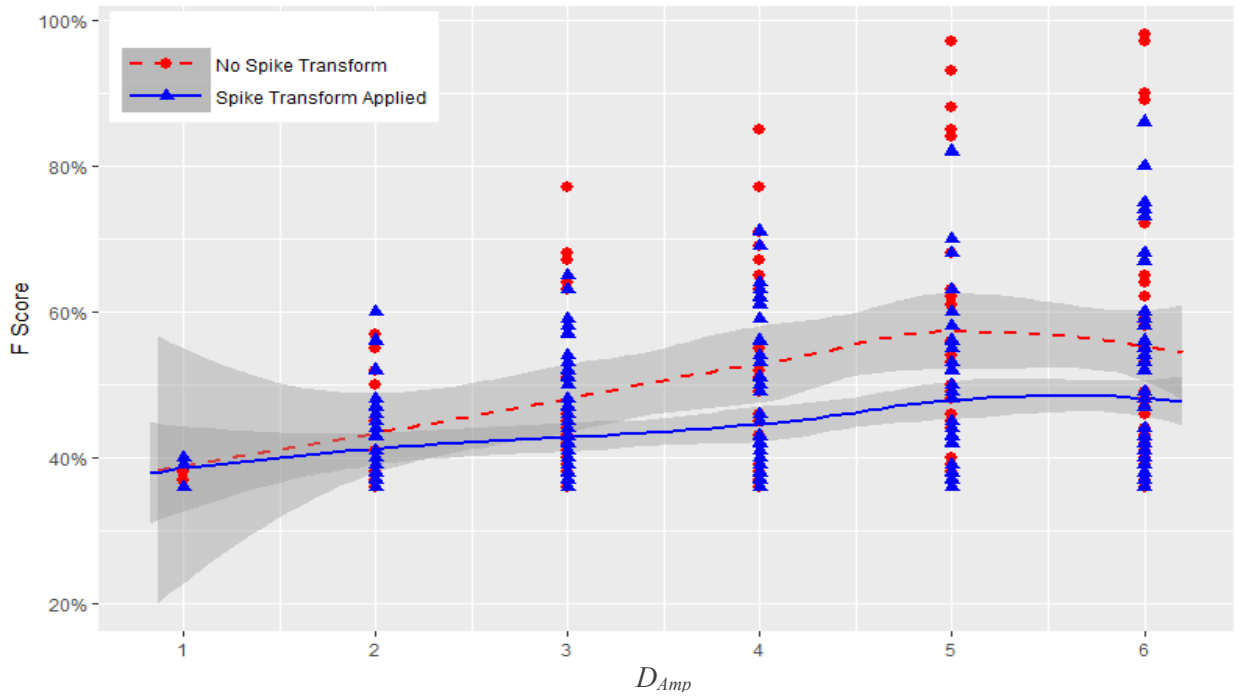

Figure 25 - Sensitivity Limit Testing Results

At $D_{Amp} = 5$ the spike transformation no longer lies in the top 5 transformations. These findings led the max values of the spike transform to be limited to 3, as even at sensitivity = 3 the spike transform underperformed capturing patterns of $D_{Amp} = 3$. The results of the test indicate that the most potential performance benefit from the transformation method was found using data streams with lower pattern amplitude.

5.4.5. Landmark/Transformation Parameter Correlation

After completion of limit testing on all potential landmarking features, the correlation between the landmarking values and the highest performing set of transformation values was tested. The test was meant to ensure that not only do the landmarking features have a significant impact on system performance, but that they directly affect the best performing level of at least one transformation value. The statistical significance of these correlations is shown in Table 29a. Each landmarking

parameter was found to have a significant effect on at least one transformation parameter, while the value of each transformation parameter was found to be affected by at least one landmarking value. Due to these findings, all landmarking parameters were utilized in the final system. The significance of these interaction can be found in Table 29b.

Table 29a & b - Landmark/Transformation P-Value (a) & Correlation Coefficients (b)

Landmark	Classifier Type	Tail T_L	BinSize A	LeadTime L_T	Sensitivity s
GT Tail	0.410	0.001**	0.146	0.134	0.000**
GT Prob	0.001**	0.251	0.021*	0.942	0.000**
GT CoVar	0.823	0.446	0.083	0.615	0.000**
D NRange	0.000**	0.005**	0.000**	0.270	0.000**
D CoVar	0.000**	0.001**	0.000**	0.002**	0.013*

a) * >95% confidence of interaction ** > 99% confidence of interaction

Landmark	ClassifierType	Tail T_L	BinSize A	LeadTime L_T	Sensitivity s
GT Tail	0.04	**0.14**	0.06	0.06	**-0.63**
GT Prob	**0.14**	-0.05	**-0.10**	0.00	**-0.37**
GT CoVar	-0.01	-0.03	0.07	0.02	**-0.31**
D NRange	**0.22**	**0.12**	**-0.19**	0.05	**-0.43**
D CoVar	**0.30**	**0.15**	**-0.19**	**0.05**	**-0.21**

b) **bold: > *95% confidence of interaction***

The correlation coefficients shown in Table 29b also clarify the effects of the landmarking feature values, and therefore the characteristics of the *D/GT* pair, on the most effective transformation parameter vales. The only transformation value whose relationship with the landmarking features cannot be assessed from these results is Classifier Type, as the variable is discrete and had to be codified for the purpose of the experiment. The only conclusion that could be drawn is that GT_{Prob}, D_{NRange}, and D_{CoVar} have a statistically significant relationship with the top performing classifier type.

The length of the provided data stream D is shown to have a positive effect on the tail length T_L used for training set generation. Additionally, a noisier data set containing a higher frequency of spikes, measured by D_{NRange}, as well as a greater pattern amplitude, measured by D_{CoVar}, result in a longer training set as well. The same combination of noisiness of the data and amplitude of the pattern to be detected also have a negative correlation with the best performing bin size A for the provided data stream. These relationships suggest that for longer, noisier data sets the best transformation set is more likely to utilize a smaller bin size, resulting in more features for training.

The best lead time L_T to use for prediction is only influenced by D_{CoVar}. The correlation coefficient between these two factors is also the smallest significant correlation detected. Therefore, the higher amplitude of a pattern may relate to a longer best lead time, but if so the relationship is insignificant.

The final transformation parameter examined was sensitivity s, which bears a negative correlation with every landmarking feature examined. The negative impact on s may suggest that on a less noisy data set with fewer events the spike transformation is more effective for detecting patterns than raw values would be, as the use of raw values is associated with $s = 0$. In addition, the strongest negative correlation in the results is between s and GT_{Tail}. The observed relationship would suggest that the spike transformation performs best when the system must create predictions using a smaller D on which to train the classifier. Having both a negative correlation with GT_{Tail} as well as GT_{Prob} strongly suggests that the advantage of the spike transform exists when the number of training instances is limited.

5.5. Landmarking System Testing

After determining which parameters affect the performance of the landmarking system and the limits at which to test these parameters, the data with which the system is tested was generated. Using the data generated according to the specified test limits the effect of each parameter of the landmarking system may be tested. The parameters of the landmarking system are shown in Table 30.

Table 30 - Landmarking Parameters Test Values

Parameter	Test Values
L_N	*1, 2, 3*
L_{Max}	*1, 2, 3*
L_{Split}	*.05, .1, .25, .50, .75, .90, .95*

L_N determines the total number of landmarks to be used by the landmarking system in testing. L_{Max} controls the number of unique transformations tested for each landmark selected. The percentage of the generated data to be randomly separated off from the test data and provided as a knowledge base is referred to as the L_{Split}. The value of L_{Split} directly controls the amount of prior results from which the landmarking system can reference and was therefore expected to have a strong effect on system performance. The amount of data provided to the knowledge base as well as the amount of data from the knowledge base utilized by the landmarking system had their effects on the performance of the system measured. Final conclusions were then drawn regarding the specifics of each of the parameter's effects.

The test values chosen were determined to represent a reasonable range of values over which each transformation may be applied. The test values of L_{Split} represented the maximum range over which the parameter may be varied while still providing a reasonable number of training or testing

instances. The testing values of L_N and L_{Max} were meant to allow a large enough range for each parameter's effect on the landmarking system to be represented in the results.

5.5.1. Test Data Generation

The *D/GT* pairs generated for the purpose of training and testing the landmarking system were meant to provide variance of all landmarkable features as well as features with which they have a significant interaction. Parameters which have no significant interaction with landmarking features were held constant for the test to reduce time required for data generation. Each landmarkable feature was tested at the limits determined through the prior limit testing. The generation parameters used to generate the landmark testing data are shown in Table 31. All valid combinations of the transformation parameters listed in Table 32 were tested on each *D/GT* pair to provide performance metrics for the knowledge base of the landmarking system.

Table 31 - Landmark Testing Data Generation Parameters

Parameter	Values		
GT_{TL}	1200	7500	
GT_{Dist}	Sinusoidal	Uniform	
GT_{Prob}	1%	2%	7%
GT_{FP}	0%		
D_{Amp}	1	3	
D_{Dur}	5	20	
D_{LT}	1	5	
D_{Rand}	0%		
D_{Dist}	Sinusoidal	Uniform	
$D_{Pattern}$	Down		
$D_{PatternType}$	Magnitude		

Table 32 - Landmark Testing Transformation Parameters

Parameter	Values
TailLength T_L	5, 10, 20, 30
BinSize A	1, 2, 4, 5, 10, 15, 20, 30
LeadTime N	1, 5
Sensitivity s	0.0, 1.0, 2.0, 3.0

5.5.2. Effect of L_{Split} on Landmarking Performance

The first landmarking parameter tested for effect against the performance of the system was the portion of the generated data sets which were provided to the landmarking system as the knowledge base. The findings of the L_{Split} testing are shown in Figure 26. In the figure boxplots are provided depicting the F-Score percentile achieved by all *D/GT* pairs when tested at the given L_{Split} value over 256 replications. The upper and lower edges of each box represent the 75th and 25th percentile response from each level of L_{Split}. The impact of L_{Split} value on the overall performance of the system was found to be minimal once a low threshold of .05 had been passed.

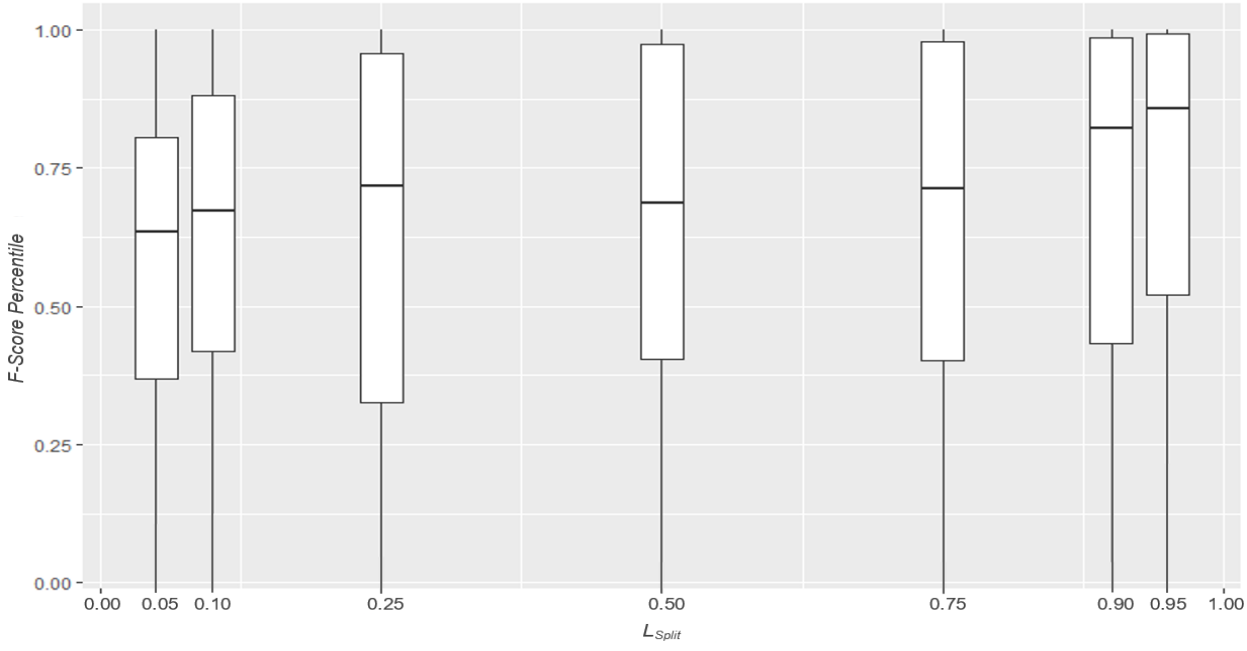

Figure 26 - Effect of LSplit on F-Score Percentile

Performance then began to increase sharply once again once L_{Split} surpassed .90. Once the system had any information to train, even only 5% of the generated data which equated to 8 unique runs, the system could set a reference point and achieve a consistent performance level up until L_{Split} = .9. Performance was then stable until nearly all instances had been added to the knowledge base, L_{Split} > .9, after which the system was capable of detecting landmarks which are nearly identical to the test instance.

5.5.3. Effects of L_N, L_{Max} on Landmarking Performance

L_N, the total number of landmarks being used, and L_{Max}, the number of unique transformations tested for each landmark, affected the way in which the system utilized the knowledge base. The effects of increasing values of each of these parameters are shown in Figure 27.

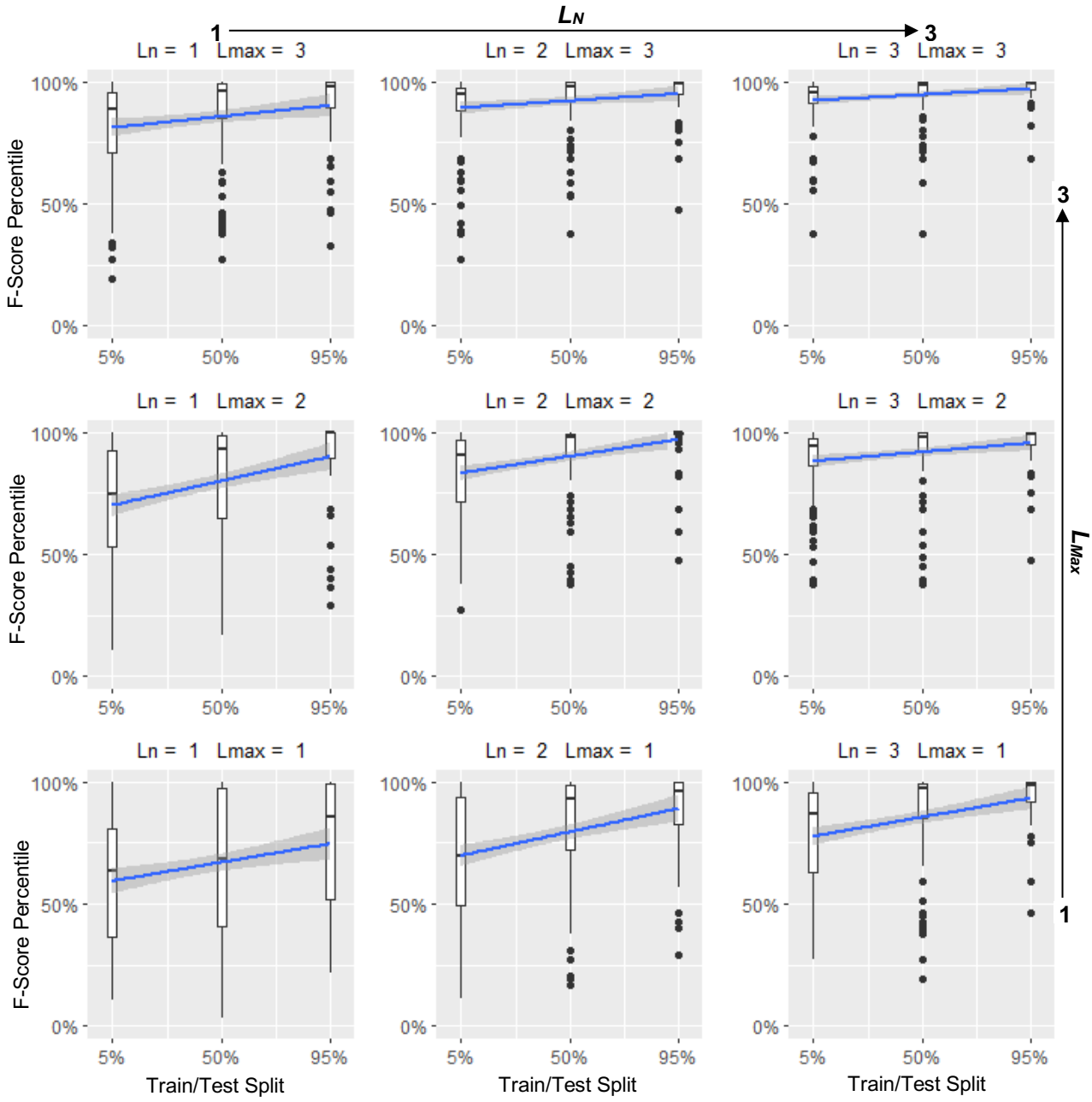

Figure 27 - Interaction Plot LN & LMax

The most notable effect shown in the results of the experiment are the positive impact of increasing the value of L_N on the F-Score of the system. At each value of L_{Max} tested, increasing the value of L_N both increased the average F-Score Percentile as well as reduced the standard deviation of the results plotted. The positive effect of L_N was consistent across all values of L_{Split} as well. At the $L_{Max} = 1$, the effect of increasing L_N over the entire range of testing values was most clearly demonstrated. As L_N was increased from 1 to 3, the standard deviation of the test results at all L_{Split} values decreased dramatically as the mean increases. Specifically, the mean of the testing results at $L_{Split} = .5$ are lower at $L_N = 3$ than the results of $L_{Split} = .95$ at $L_N = 2$. The conclusion drawn from these observations is that L_N has a more significant positive impact on F-Score percentile than any other landmarking system parameter.

Although the effect of L_N was more substantial than that of L_{Split}, the positive effect of L_{Split} was still consistent across all values of L_N and L_{Max}. At every combination of landmarking system parameter values tested, an increase in the value of L_{Split} both increased the mean of the F-Score Percentile as well as reduced the standard deviation of performance. The positive effect of L_{Split} was more pronounced at lower values of L_N when the system was more restricted on the number of transformation combinations utilized.

As with the other two landmarking system parameters, L_{Max} shows a positive correlation with the F-Score percentile of the system. At lower values of L_{Split} the effect of increasing L_{Max} was more noticeable in a reduced standard deviation and increased mean of the response. At $L_{Split} = .95$, especially at the highest value of L_N tested, L_{Max} yielded a much less significant improvement.

5.5.4. Final Landmark Conclusions

In terms of the effects of landmarking system parameters, the larger the knowledge base that the system is provided with, the greater the performance of the selected transformation set. By the time a significant knowledge base has been established, shown by $L_{Split} > .90$, the expected output of the system performs consistently in the top 25% of all possible transforms, even with the lowest L_N value tested. Increasing L_N yields a greater result, but by a value of $L_N = 3$ the average transformation set selected is in the top 20% regardless of the amount of prior knowledge provided.

At lower levels of prior knowledge, testing at a higher L_{Max} values allows the system to yield better results through maximizing the use of the landmarks with the most similar landmarking feature values. At a higher value of L_{Split}, with a larger knowledge base provided, the system's performance is less affected by L_{Max} as the greater number of landmarks likely result in more close matches to the test *D/GT*. In application, maximizing the knowledge base and understanding the L_{Max}/L_N tradeoff for a provided test instance could allow for consistent generation of effective transformation sets without the need for multiple full system iterations.

6. Validation Experiment

A validation experiment was conducted as a means of legitimizing the proposed transformation techniques. Although summation across values is a method commonly used for feature reduction, the method's performance in the specific system remained untested. An experiment was established using a baseline of raw values as well as a limited total enumeration run. The validation run contains only a measure of benefit provided by the transformation process.

6.1. Results

Table 33 displays the testing results of both the baseline transformations using raw values as well as the top performing Tree and SVM classifiers at $s = .5$ & $s = 1$. Regardless of lead time, the baseline decision tree classifier failed to detect any attacks. The failure to predict is likely due to excessive noise generated by a feature set containing 1 unique value and leads to the creation of branches which favor a heavy negative bias. After performing both the spike conversion at $s = .5$ or $s = 1$ as well as the binning process the decision tree was able to successfully perform at a similar or superior level to the SVM. Although the SVM was generating some accurate predictions prior to data transformation, the F-Score of the classifier was increased at every value of N through the transformation process. The SVM classifiers showed an increase in performance with either spike conversion, however all 3 N values produced better results through SVMs with $s = 1$. On the contrary, tree classifiers yielded mixed results between the two sensitivity levels depending on the value of N being tested. The primary conclusion which is upheld by these results is that performance is improved across the board by the data transformation process proposed.

Table 33 - Validation Testing F-Score Results

N-Value	***Sensitivity***	***Classifier Type***	***F-Score***
1 - Day	Baseline	Tree	0.0%
1 - Day	Baseline	SVM	8.0%
1 - Day	.5	Tree	11.6%
1 - Day	.5	SVM	13.0%
1 - Day	1	Tree	16.7%
1 - Day	1	SVM	21.3%
3 - Day	Baseline	Tree	0.0%
3 - Day	Baseline	SVM	9.5%
3 - Day	.5	Tree	20.7%
3 - Day	.5	SVM	10.1%
3 - Day	1	Tree	14.3%
3 - Day	1	SVM	17.1%
5 - Day	Baseline	Tree	0.0%
5 - Day	Baseline	SVM	7.5%
5 - Day	.5	Tree	23.1%
5 - Day	.5	SVM	12.4%
5 - Day	1	Tree	12.8%
5 - Day	1	SVM	17.1%

The detailed results of both the baseline transformations and the transformations which yielded the highest F-Score at each test value for *N* are shown in Table 34. Within the *N = 1* range of classifiers the transformation process yielded an increase in recall, with classifiers managing 5/5 correctly predicted attacks at both values of *s* compared to the baseline which yielded only 1 correct prediction. The performance advantage carried over to *N = 3* as well, where both classifiers trained on the transformed data set yielded higher recall, precision, and F-Scores than the baseline. With *N = 5*, the gap between recall of each classifier diminished. All 3 classifiers, *s = .5, s = 1,* and baseline, only predicted 3 of the 5 attacks. However, the precision of the classifiers built using

transformed data was much higher. The higher precision and F-Score show that the classifiers built on transformed data captured the same number of attacks, but with far fewer false positives.

Table 34 - Top Performing Classifiers vs. Baseline

N	*s*	T_L	*A*	*Type*	*F-Score*	*Precision*	*Recall*
	.5	10	2	SVM	13.0%	6.9%	5/5
1	1	30	10	SVM	21.3%	11.9%	5/5
		Baseline		SVM	8.0%	5.0%	1/5
	.5	20	10	Tree	20.7%	12.5%	3/5
3	1	10	5	SVM	17.1%	10.0%	3/5
		Baseline		SVM	9.5%	6.3%	1/5
	.5	10	10	Tree	23.1%	14.3%	3/5
5	1	20	5	SVM	17.1%	10.0%	3/5
		Baseline		SVM	7.5%	4.0%	3/5

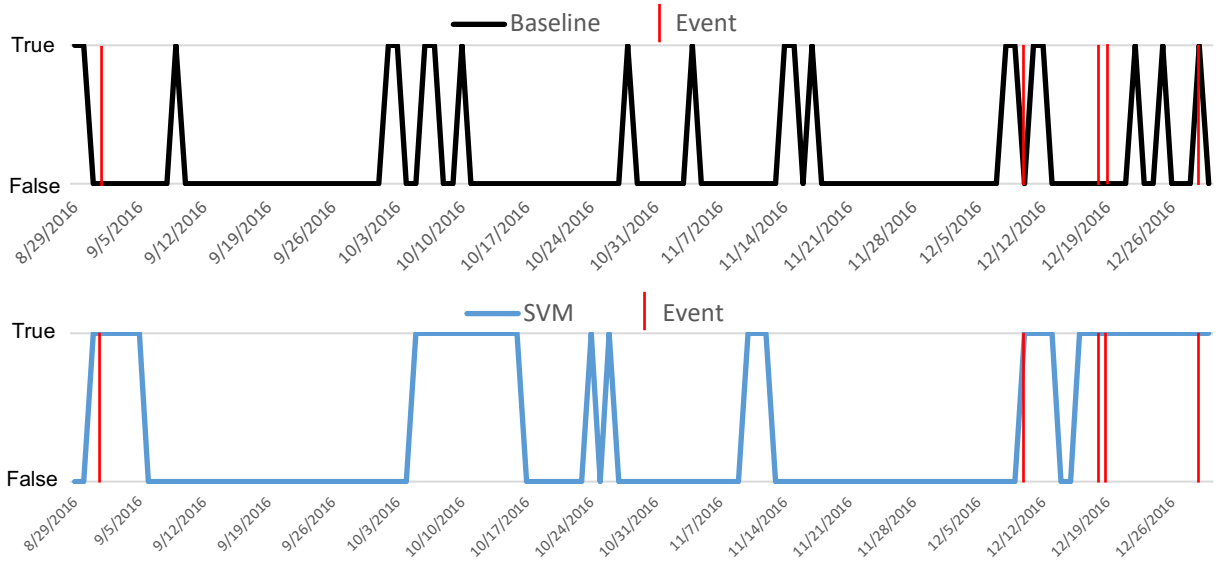

Figure 28 - Prediction Timeline of Baseline vs Transformation Method at N=1

Figure 28 illustrates the predictions made by both the highest performing SVM classifier in terms of F-Score as well the baseline predictor. Both classifiers were attempting to generate predictions with a lead time $N = 1$. As can be observed at both the beginning and end of the timeline, the

baseline failed to capture 4 of the 5 events, although produced fairly close predictions. After transforming the data, the trained SVM was able to capture all recorded attacks, creating a window of positive attack prediction between 12/19 and the end of the data set. The transformed feature set training resulted in some false positives but captured all 3 of the attacks over the examined time span. There is a similar observable pattern to the false positives generated by both classifiers between 10/3 – 11/21. The period of high predictive activity was likely either caused by abnormally high tweet counts with no resulting attacks or one or more attacks occurring without record. With regards to the prior, false positives are to be expected when generating predictors using a leading indicator because not every spike in activity around an entity will result in a cyber-attack.

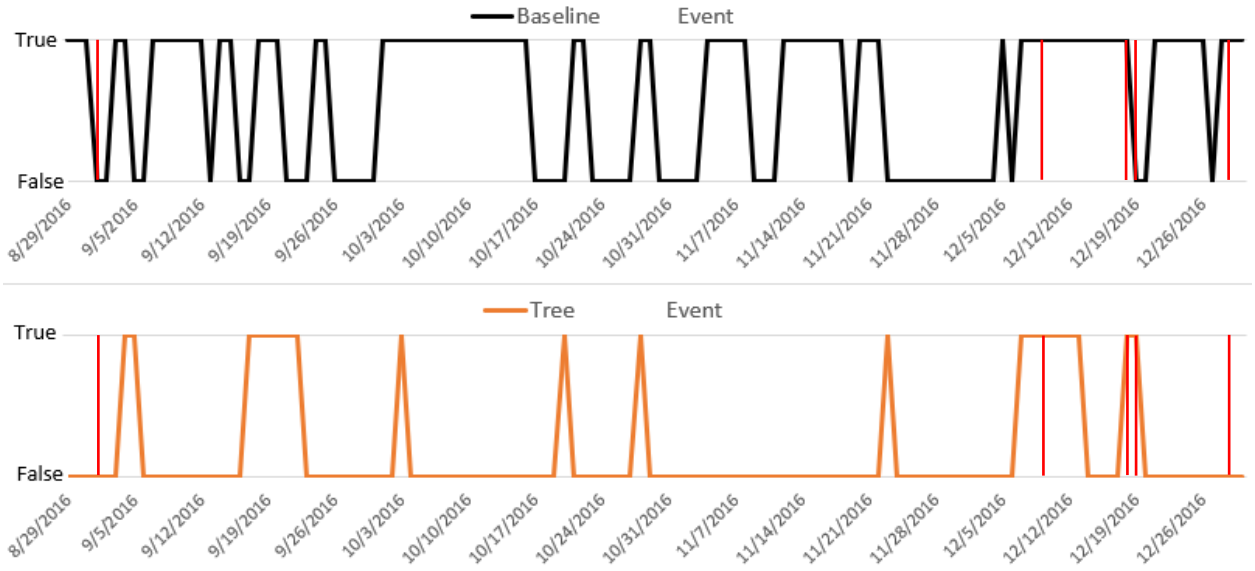

Figure 29 - Prediction Timeline of Baseline vs Transformation Method at N=5

Figure 29 displays a much larger discrepancy between the performance of the two classifiers being examined than was shown in Figure 28. While the decision tree classifier does not manage to capture any more events than the baseline, predicting only 3 of the 5, the results were achieved while creating far fewer false positives. Between the first recorded event on 9/1 and the event on

12/10 the baseline predictor generated approximately the same number of positive attack predictions as negative. Over the same time span the decision tree generated only 14 false positives, resulting in a final precision of 14.3% compared to the baseline's precision of 4.0%. Predictions began to lose their value if they occurred frequently with far too many false positives. The transformation process managed to greatly reduce the false positive problem, while maintaining the same recall rate and yielding more accurate predictions.

6.2. Validation Experiment Conclusions

The validation experiment showed a promising improvement in the predictive capability of both classifier types. As displayed in Figure 28 and Figure 29, applying a spike conversion as well as a binning transformation yielded a higher performing classifier than the baseline of using no transformations. Even using a limited range of values for each transformation yielded an average F-Score increase of >10% across the 6 unique N and s levels. Extending the granularity of the value set used for testing is hoped to result in a higher performing final classifier in terms of maximizing F-Score.

Increased granularity is possible with a minimal increase in processing time through the use of the proposed landmarking system. Implementing the landmarking process through training on similar data sets would render the system capable of matching a classifier type and parameter combo to the provided data set based on meta-feature values prior to performing any test iterations.

7. Conclusions and Future Work

The results of correlation testing conducted suggest that the transformation system developed outperforms the use of raw values when smaller, less noisy data sets are used. Because the system was originally intended to address an issue training and making useful predictions on limited data sets where event counts may be as low as <50, the results produced show promise for the application of the system in the intended scenario of cyber-attack prediction.

Although unable to be tested on real-world data, the capability of the landmarking system for determining a best fit transformation set has been assessed. The system has shown to be capable of selecting a transformation set within the top 25th percentile of all tested transformations when provided with three or more landmarks. The performance results hold true for any combination of L_N and L_{Max} as long as the total number of landmarks tested is >3. The knowledge base required for the landmarking system to perform effectively has been determined to be smaller than originally expected. The landmarking system consistently produced results in the top 50th percentile of all tested transformations with only 5% of the generated data being used as a knowledge base. When provided with a larger knowledge base, the system was able to consistently produce transformation sets in the top 10th percentile while testing <10 transformations. Being able to select a top performing result by testing only 10 transformations rather than having to try all ~2,300 potential transformations included in the experiment provides a huge reduction in run time and processing power required.

When tested through the validation experiment the transformation system showed a consistent performance improvement over using the raw data values provided. The baseline raw data values

were outperformed by the transformed training sets in terms of F-Score at every lead time tested. The event predictions that the best transformation set produced were less sporadic and more centered about the real event times, with a lower number of individual false positive event predictions occurring over the course of the prediction period.

While the system has been found itself useful in specific applications in the current state, potential improvements are proposed to be tested in the future. The primary improvement would be to use principal component analysis to reduce the number of low value bins included in the bin profile produced by the system. The number of features required to train a classifier would be reduced even further with minimal negative impact on performance.

Another proposed improvement involves adapting the system to be capable of training and testing on multiple transformations at once, whether they be separate transformations applied to the same data stream or multiple data streams simultaneously. The capability to handle multiple training and testing sets at once would allow the system to utilize any interactions or correlations between the multiple training features.

While there is room to improve upon the system developed, current results show promise for application in the prediction of real-world events when using limited training data.

8. Works Cited

Antunes, C. M., & Oliveira, A. L. (2001). *Temporal data mining: An overview*. Paper presented at the KDD workshop on temporal data mining.

Ashford, W. (2012). Powerful cyber attack tools widely available, say researchers. Retrieved Sep 16, 2017, from http://www.computerweekly.com/news/2240162578/Powerful-cyber-attack-tools-widely-available-say-researchers

Bernhard, P., Hilan, B., & Christophe, G.-C. (2000). *Meta-Learning by Landmarking Various Learning Algorithms.* Paper presented at the Proceedings of the Seventeenth International Conference on Machine Learning.

Brierley, P., Vogel, D., & Axelrod, R. (2011). Heritage Provider Network Health Prize Round 1 Milestone Prize: How we did it–Team 'Market Makers': ed.

Bronk, C., & Tikk-Ringas, E. (2013). The cyber attack on Saudi Aramco. *Survival, 55*(2), 81-96.

Brownlee, J. (2016). Supervised and Unsupervised Machine Learning Algorithms. *Machine Learning Mastery.* Retrieved Oct 12, 2017, from machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

Chae, B. K. (2015). Insights from hashtag# supplychain and Twitter Analytics: Considering Twitter and Twitter data for supply chain practice and research. *International Journal of Production Economics, 165*, 247-259.

Culnan, M. J., McHugh, P. J., & Zubillaga, J. I. (2010). How large US companies can use Twitter and other social media to gain business value. *MIS Quarterly Executive, 9*(4).

Doan, T., & Kalita, J. (2016). Predicting run time of classification algorithms using meta-learning. *International Journal of Machine Learning and Cybernetics* 8.6. 2017

Frank, E., Hall, M., Holmes, G., Kirkby, R., Pfahringer, B., Witten, I. H., & Trigg, L. (2009). Weka-a machine learning workbench for data mining *Data mining and knowledge discovery handbook* (pp. 1269-1277): Springer

Friedman, J. H. (1997). On bias, variance, 0/1—loss, and the curse-of-dimensionality. *Data mining and knowledge discovery, 1*(1), 55-77.

Heritage Health Prize. (2012). Retrieved Oct 12, 2017 from https://www.kaggle.com/c/hhp

InternetLiveStats. (2017). Twitter Usage Statistics. Retrieved Oct 8, 2017 from www.internetlivestats.com/

Jordan, T. (2001). Hacktivism: Direct Action on the Electronic Flows of Information Societies *Challenges to Democracy* (pp. 118-130): Springer

Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017). DDoS in the IoT: Mirai and other botnets. *Computer, 50*(7), 80-84.

Kubat, M., & Matwin, S. (1997). *Addressing the curse of imbalanced training sets: one-sided selection.* Paper presented at the ICML.

Lee, R. M., Assante, M. J., & Conway, T. (2014). German steel mill cyber attack. *Industrial Control Systems, 30*.

Lerman, K., & Ghosh, R. (2010). Information contagion: An empirical study of the spread of news on Digg and Twitter social networks. *Icwsm, 10*, 90-97.

Liu, Y., Zhang, J., Sarabi, A., Liu, M., Karir, M., & Bailey, M. (2015). *Predicting cyber security incidents using feature-based characterization of network-level malicious activities.* Paper presented at the Proceedings of the 2015 ACM International Workshop on International Workshop on Security and Privacy Analytics.

Maynard, T., & Ng, G. (2017). Counting the Cost: Cyber Exposure Decoded. Emerging Risks Report, https://www.lloyds.com

MCoM. (2015). *Traffic Violations - Comma Separated Values File*. from catalog.data.gov/dataset/traffic-violations-56dda/resource/450018e7-f6c0-43fd-b5c9-a83de293b206

Meina, M., Janusz, A., Rykaczewski, K., Slezak, D., Celmer, B., & Krasuski, A. (2015, 2015). *Tagging Firefighter Activities at the emergency scene: Summary of AAIA'15 data mining competition at knowledge pit.* Paper presented at the *2015 Federated Conference on Computer Science and Information Systems*.

Munkhdorj, B., & Yuji, S. (2017). Cyber attack prediction using social data analysis. *Journal of High Speed Networks, 23*(2), 109-135.

NOAA. (2017). Climate Data Online: Dataset Discovery. from www.ncdc.noaa.gov/cdo-web/datasets

Okutan, A., Yang, S. J., & McConky, K. (2017). *Predicting cyber attacks with bayesian networks using unconventional signals.* Paper presented at the Proceedings of the 12th Annual Conference on Cyber and Information Security Research.

Pontes, E., Guelfi, A. E., Kofuji, S. T., & Silva, A. A. (2011). *Applying multi-correlation for improving forecasting in cyber security.* Paper presented at the Digital Information Management (ICDIM), 2011 Sixth International Conference on.

Reif, M., Shafait, F., & Dengel, A. (2011). *Prediction of Classifier Training Time Including Parameter Optimization.* Paper presented at the KI.

Roman, R., Zhou, J., & Lopez, J. (2013). On the features and challenges of security and privacy in distributed internet of things. *Computer Networks, 57*(10), 2266-2279.

Syarif, I., Prugel-Bennett, A., & Wills, G. (2016). SVM Parameter Optimization using Grid Search and Genetic Algorithm to Improve Classification Performance. *TELKOMNIKA (Telecommunication Computing Electronics and Control), 14*(4), 1502-1509.

Werner, G., Yang, S., & McConky, K. (2017). *Time series forecasting of cyber attack intensity.* Paper presented at the Proceedings of the 12th Annual Conference on cyber and information security research.

Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation, 8*(7), 1341-1390.

Yu, H.-F., Lo, H.-Y., Hsieh, H.-P., Lou, J.-K., McKenzie, T. G., Chou, J.-W., Wei, Y.-H. (2010). *Feature engineering and classifier ensemble for KDD cup 2010.* Paper presented at the KDD Cup.

Zdravevski, E., Lameski, P., Mingov, R., Kulakov, A., & Gjorgjevikj, D. (2015). *Robust histogram-based feature engineering of time series data.* Paper presented at the 2015 Federated Conference on Computer Science and Information Systems (FedCSIS).