# Intrusion Detection using Ensemble Learning on Combined Features

Michael Milliken, Yaxin Bi, Leo Galway, Glenn Hawe
*School of Computing and Mathematics, Ulster University, Belfast, United Kingdom*

## Abstract

*Network intrusions may illicitly retrieve data/information, or prevent legitimate access. Reliable detection of network intrusions is an important problem, misclassification of an intrusion is an issue by the resultant overall reduction of accuracy of detection. A variety of potential methods exist to develop an improved system to perform classification more accurately. Feature selection is one area that may be utilized to successfully improve performance by initially identifying sets and subsets of features that are relevant and non-redundant. Within this paper explicit pairings of features have been investigated in order to determine if the presence of pairings has a positive effect on classification, potentially increasing the accuracy of detecting intrusions correctly. In particular, classification using the ensemble algorithm, StackingC, with F-Measure performance and derived Information Gain Ratio, as well as their subsequent correlation as a combined measure, are presented.*

## 1. Introduction

An ongoing challenge in modern networks are intrusions illicitly retrieving data/information, or preventing legitimate access [1]. These intrusions are otherwise known as attacks. In response to these attacks, Network Intrusion Detection Systems (NIDS) are developed to provide some method of detection. Detection may be performed during or after the occurrence of an attack through analyzing all features of packets or a feature set forming a summary of packets incoming in real time or retained historically.

An important aspect of NIDS is identification of features relevant to specific types of intrusion and intrusions in general. Justification for this is threefold: firstly, it improves understanding/interpretation. Secondly, it can lead to reductions in the training times of models [2], [3]. Thirdly, it improves accuracy by reducing over-fitting [2], [3].

NIDS are typically evaluated using measures such as accuracy/detection rate and false alarm rate. A particular measure of the accuracy is F-Measure (FM) which takes into account False Positives (FP) and False Negatives (FN), all of which are described later in *5. Results and Discussion*. One particular problem of intrusions and detection is the volume of FP and FN results obtained based on a confusion matrix of actual and predicted values. Actual values are those that are provided with and accurately describe instances. In this work, an instance is comprised of multiple features and associated values derived from a packet, along with the class label. Predicted values are those produced by the classification model or a classifier. Comparisons of actual and predicted values allow evaluation of detection.

An NIDS may typically experience balance or tradeoff between the number of FP and number of FN values obtained [3], [4], [5], thus a primary challenge is reduction of false values, independent of high volumes of traffic, or presence of unknown attacks. A specific issue known to be a limiting factor of NIDS is need for a reduction in rate of occurring FPs, i.e. instances incorrectly predicted by NIDS to be intrusions that were in actuality *Normal* [6], [7], [8].

A popular approach over the last decade has been use of multi-layered/tiered approaches [9], [10], [11], incorporating multiple methods/algorithms within a hierarchy. The focus of this paper concerns function of algorithms and results of earlier processing steps, analysis and classification of instances, specifically explicit feature pairings and combinations and results of such pairings when implemented with an ensemble learning algorithm.

The hypothesis is that combination and pairing of features should improve algorithmic performance, such as reduced FP rate or improved FM. However it is recognized as possible that such a combination could potentially reduce the algorithmic performance in some instances. To test the hypothesis we measure performance of supervised Machine Learning (ML) algorithms on sets of experimental data containing paired and merged features. In addition, we also use entropies of instance data for each pairing to calculate its Information Gain Ratio (IGR) and correlation to FM.

As correlation describes the relationship between the two values of FM and IGR, it potentially provides means of selecting a feature pairing based on the correlation coefficient between the values. Consequently, positive magnitude is preferable, as it indicates that IGR and FM/performance increase together, potentially leading to more accurate classification.

The rest of this paper describes some background of Feature Selection and Extraction as well as ML algorithms and existing systems. It also describes the methodology employed for the results of the experiments as well as the experiment setting, followed by discussion of results and subsequent conclusion.

## 2. Background

To determine a more useful feature set for classification of a problem, feature selection and feature combination/feature extraction needs be performed. Feature selection typically chooses the features to include within a subset based on some criteria while feature combination transforms or combines existing features into new features; in short, it seeks to reduce the dimensionality of a problem and potentially provide more discriminative features. Reduction of dimensionality is considered important where a large set of unnecessary features can increase the computational complexity in performing classification.

There are three categories of algorithms for feature selection; filter methods, wrapper methods and embedded methods. Filter methods perform selection of a subset according to a criteria. Wrapper methods perform selection of a subset according to algorithmic performance. Embedded methods perform feature selection as part of training.

For selection of features some filter methods include Information Gain, Information Gain Ratio, Gini Index, Chi-square [8], minimal redundancy maximal relevance (mRMR), and Normalized Mutual Information Feature Selection (NMIFS) [12]. Some wrapper methods include Sequential Forward Selection (SFS); adding features to improve performance, Sequential Backward Selection (SBS); removing features to improve performance and Genetic Algorithms (GA); selecting feature sets without exhaustive search. An example of embedded methods is Support Vector Machine Recursive Feature Elimination (SVM-RFE).

Relevance of features is notably important when making selections of subsets, a feature selection method that functions incrementally, considering one feature at a time, may only consider features individually while ignoring inherent combinations that may or may not be explicitly known or stated [12]; the absence of one or more of these features may have a negative effect on performance, by decreasing discrimination ability of one or more other features. Discrimination being a measure of a features ability to correctly classify instances, the better a feature set is at discriminating the more accurate and successful classification may be.

Where a feature subset is used it is hoped that removal of irrelevant and redundant features provides better performance; irrelevant meaning not supporting a particular problem and redundant meaning providing information already provided by another feature.

Some feature combination methods are Principal Component Analysis (PCA) [2], Independent Component Analysis (ICA) [12], linear discriminant analysis, and projection pursuit. PCA aims to reduce the dimensionality of the problem by linearly transforming a number of observations. Kernel PCA, related to PCA, performs a nonlinear transformation.

ML algorithms are commonly used to detect intrusions by utilizing input from a dataset containing multiple features. Methods of feature selection are often used with ML algorithms to determine optimal feature sets to form a reduced/changed dataset. The reduced/changed dataset is generated with hope that it will improve the results beyond those where all features were used without applying some selection of features.

Typically, supervised or unsupervised approaches are used, employing labelled or unlabeled data respectively to form a model that may be subsequently used for classification of future network packets. ML algorithms may also be *offline* or *online*, referring to analysis of historical or incoming real-time data respectively. This paper focuses on *offline*, supervised ML algorithms.

Examples of differing classification approaches follow. In a binary class approach, instance classification may be *Normal* or *Attack*. In a multi-class approach, classes may be more than binary or reduced to binary. A multi-class approach with multiple classes may focus on *Normal*, *Probe* and *Denial Of Service* (*DOS*), if reduced to a binary approach the classes may be reduced to *Normal* and *Other* (non-normal). Classifications from multiple base algorithms may be utilized in a single classification output, this approach is an *ensemble*. Typically, outputs from multiple base algorithms are used as input into a meta learner to formulate an ensemble and provide a single aggregated output. Consequently, the final output from the meta learner may provide more accurate results than individual base algorithms due to diversity of individual classifiers or associated feature spaces [13]. Popular ensemble methods include Bagging, Boosting and Stacking.

Bagging replicates datasets with multiple instances of a base algorithm. Differences introduced in replicated datasets introduce variation to base algorithm thus improving average classification result [14].

Boosting uses sequential base algorithms with the same set of data. Output of each base algorithm is passed to subsequent algorithms. Results improve at each subsequent stage by changes in weighting of misclassified instances to improve classification [2].

Stacking involves the use of multiple different base algorithms with the same set of data [15]. The

output, actual class and predicted probabilities of classes for an instance, of each base algorithm is used as input to a regression model per class to provide final classification of an instance [16]. Use of class probabilities rather than class predictions improved performance of Stacking. StackingC further improved efficiency of Stacking. This was achieved using only the probability of a specific class for each linear model for that class rather than considering other class probabilities as well [17].

Octopus-IIDS [10], an Intelligent Intrusion Detection System (Intelligent IDS), uses an ensemble of Kohonen and SVM networks, splitting data into attack classes using a Kohonen network then reclassifying instances into *Attack* or *Normal* using SVMs.

Hidden Markov Models with Payl (HMMPayl) [18] is an improvement over the Payl detection method that looks to the contents and distribution of a payload for indication of attacks. HMMPayl still performs analysis on payloads but also uses Hidden Markov Models (HMMs) as the initial step to form multiple classifications, using the results to form a final classification. Additions of multiple HMMs to the method improves results; using diverse results while using the same type of classifier.

Work described in [19] uses an ensemble of K-Means Clustering (KMC) with Naïve Bayes (NB) classification; forming similarly behaving clusters with KMC before using NB to correct misclassifications from KMC, effectively grouping data, making classification efficient and more accurate.

Ensemble based methods are popular and work well for intrusion classification. As such, StackingC has been chosen for use in testing the hypothesis. This work differs from the research found within the literature in the early manipulation of input data prior to use with algorithms to determine if the method of pairing can have a positive effect, at least on small scale datasets.

## 3. Methodology

A pair of features is formed through explicit selection of existing features. An example pairing is Destination IP (indicating relevant host, may be focus of data traffic) with Source Port (may indicate traffic instances). It is anticipated that feature pairings may potentially produce a feature more able to be used to classify instances while reducing feature set dimensionality.

Features pairings are formed by simple process of appending specific features as individual strings to each other into a larger combined string, containing information from both features within a single feature, thereby potentially reducing enumerations on more than one set of feature values for the same information.

The above method is restricted by necessity to maintain an adequate number of features where extreme pairing of appending all features together would negate the ability of applying ML approaches. Particularly where a model is generated from the presence and interactions of multiple features. Unless the approach was aware of and able to differentiate specific portions of large pairings as individual features. However this would require further adjustments to any ML approach. A further reason would be where pairing features into one feature of a specific number of characters may increase complexity of using the larger feature for analysis, producing a negative effect.

To determine and measure potential benefit of pairing features it is important that effects of pairings and combinations are investigated. One method used is measuring effects through classification performance of each pairing and combination with respective outputs from various specified ML algorithms. Multiple related measures of performance in comparisons between differing feature pairings to determine the best performing for each measure and, if possible, overall.

A second method used, calculating IGR of each pairing and respective feature set to compare results, the IGR calculation is given in (1).

$$IGR = \frac{InfoGain}{SplitInfo} \tag{1}$$

Calculation of IGR differs only in entropy calculation with two alternating assumptions made as to dependence or independence of features.

Overall entropy of all features was calculated, thus feature sets were considered as one feature and, similarly, value sets were considered as one value. This was performed as entropy would remain the same where features were not different in some way.

Also, ranking of features involved in the pairing may not be easily comparable when including different numbers of features, for example Pairing 1 (P1) containing four features with other pairings containing only two features, and all pairings containing different features.

A combination of both methods is correlation of FM and IGR using both dependence and independence assumptions.

Use of an included flow file was made, automatically labelling individual packets based on selected features used to identify which, if any, labelled flow packets belonged to. The aforementioned file is a labelled set of data instances, included with the dataset used herein, representative of packet flows between sources and destinations; each instance represents the aggregate of network traffic including payloads and number of packets and bytes of packets between the source and destination, both identified with respective IP addresses and

Ports. A set of specific features, Source Port and Destination Port of P1 were reformatted to nominal values to provide closer comparisons to the other pairings. The dataset from which the flow file originated is described in Section 4.

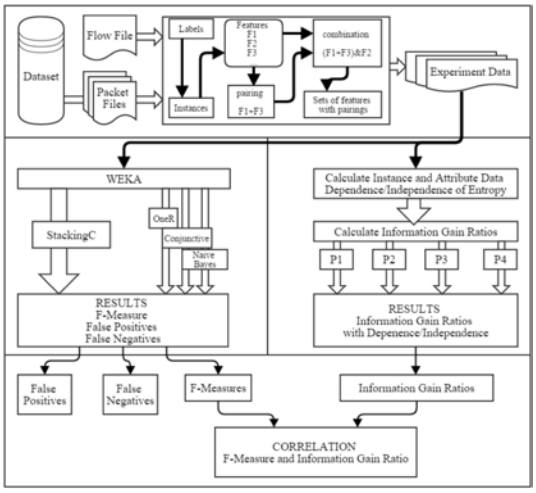



Figure 1. Methodology depicting pairings of features, classification, calculations of information gain ratio and subsequent correlation

Figure 1 depicts the methodology of the experiment and generation of training data set files. The methodology is split into 3 stages. Stage 1, top of Figure 1, labels instances of the packet capture format (pcap) files by using the flow file, then selecting and pairing features. The creation of pairings is performed by first selecting all features from the pcap file that are to be included in each experiment instance. Following this, a number of features are selectively paired together and merged, forming feature pairings. Then pairings and other features from the previously selected set form combinations of the total feature sets.

Combinations of features and differing pairings form feature sets uniquely identified by included feature pairing, subsequently referred to as experiment data files. Subsets are produced from each experiment data file to follow a pattern of additions of features; each subset with the original set files are used at Stages 2A and 2B, as illustrated respectively in middle-left and middle-right sections of Figure 1.

In Stage 2A, Weka was used to classify instances using a number of supervised ML algorithms. Four algorithms were used in total; three base algorithms (OneR, Conjunctive and Naïve Bayes) and one meta algorithm (StackingC). The base algorithms being chosen for low complexity, similar to the meta learner of StackingC, as well as potential diversity. All generated training file instances were classified using OneR, Conjunctive, Naïve Bayes and StackingC to produce a series of performance results, found in Section 5.

In Stage 2B, IGR was calculated using (1). The IGR is calculated for each subset from and including the Base feature set with additions of features, numbered 6, 7, 10, 11 and, with each addition alternating the presence of, 8 as listed in Table 1, to the full feature set. The Base differs based on the pairing, as described in Table 2, and includes features numbered 5 and 9 with presence of 8 from Table 1. Calculation of IGR is performed as normal with exception that for each subset all features are merged into one large feature; following this each instance value is merged together into a single value of the larger feature.

Two approaches are then taken, considering values dependent or independent of each other, in considering instance values to form the instance into a singular value. In these cases calculation of the probability of each instance differs. Where dependent, probability of each instance is based on co-occurrences of each instance per class. Where independent, probability of each instance is based on the product of probability of each value per class prior to the formation of the features to a larger feature and values into instances. Resultant entropies from both assumptions are used to calculate the IGR as expressed in (1).

Stage 3, bottom of Figure 1, correlates the FM and IGR of each pairing and feature sets from Stages 2A and 2B respectively. Pearson's correlation coefficient is used to calculate the correlation between the FM and IGR of each pairing as the features are added.

## 4. Experiment Setting

The dataset used for the experiment presented was the ISCX2012 dataset [20]. The ISCX2012 dataset consists of seven days of network traffic. Each day of network traffic contains a specific attack scenario with each scenario adding further complexity when considered with the previous day. The attack scenario specifically selected for the experiment was that of a "Distributed Denial of Service using IRC botnet". Relatively low volumes of network traffic were selected, producing three sets of experiment data. The three sets of experiment data consisted of 10,000 (10K), 20,000 (20K) and 30,000 (30K) individual packets taken from the day of traffic, with each subsequent set including the previous set. The 20K set was 10K with an additional 10,000 packets, and the 30K set was 20K with an additional 10,000 packets. No restrictions were made as to the inclusion of the data, i.e. no particular host was chosen relating to incoming or outgoing packets. The experiment sets consisted of a total of 11 features excluding the class, as listed in Table 1.

Training data is collectively four training data files. Each file is a subset of the full feature experiment set based on the included pairing. Each

file has a feature removed subsequently to produce a specific subset after each removal. The experiment results follow an order, reverse from the removals, from the initial Base set to the full feature sets, as previously described.

Table 1. Feature list

| Number | Name |
|---|---|
| 1 | Source IP |
| 2 | Source Port |
| 3 | Destination IP |
| 4 | Destination Port |
| 5 | Payload |
| 6 | App Name |
| 7 | Source TCP Flags |
| 8 | Bytes to Source |
| 9 | Bytes to Destination |
| 10 | Packets to Source |
| 11 | Packets to Destination |

The features were identified by appearance in the included flow file and selected from the individual packets contained in the included pcap file. Each individual packet was analyzed with particular selections up to the deepest packet made, selecting the required data from packets to form instances for classification.

The experiments detail the use, and potential benefit, of explicit specific feature pairings. The specific pairings made were each formed from appending heterogeneous features. Three such pairings, Pairing 2 (P2), Pairing 3 (P3) and Pairing 4 (P4) were made, with the lack of paired features being considered as its own pairing, (P1), for the sake of comparison, as presented in Table 2.

Table 2. Partial feature pairings base set

| Pairing | Partial Feature Base Set ('&' denotes paired features) |
|---|---|
| P1 | Source IP, Destination IP, Source Port, Destination Port |
| P2 | Source IP&Destination Port, Destination IP&Source Port |
| P3 | Source IP&Destination IP&Destination Port, Source Port |
| P4 | Source IP&Destination IP&Source Port, Destination Port |

For the experiment, additions of features were made to determine results achievable with their inclusion or exclusion, thus the Base feature sets were identified as an initial starting point. Base sets consisted of multiple features. These features were paired features of respective pairings, as presented in Table 2, as well as Bytes to Source. After the Base set the remaining features were added in a stochastic specified order using forward selection method. Inclusion and subsequent exclusion of Bytes to Source alternate with the addition of each feature.

## 5. Results and Discussion

Results of the experiment are discussed and described with differing measurements. Measurements are used to analyze performance of the four pairings, producing overall analysis of performance and comparison between pairings. The goal of this analysis is determining performance benefits, decreases and similarities which may exist dependent on a particular pairing.

First measurement used for analysis is FM, the measure of a result's accuracy being the harmonic mean of Precision (Pr) and Recall (Re). Both Pr and Re may be expressed by respective equations through the use of classification results from experiment results, these base values being True Positives (TP), FPs and FNs. TPs measure the amount of correctly classified positive case instances. FPs and FNs measure the amount of incorrectly classified positive and negative case instances, those that are *Normal* but deemed *Attack* and those that are *Attack* but deemed *Normal* respectively. Second and third measurements for analysis are FPs and False Negatives (FN). Thus Pr, Re and FM are expressed in (2), (3) and (4) respectively.

$$\mathrm{Pr} = \frac{TP}{TP + FP} \tag{2}$$

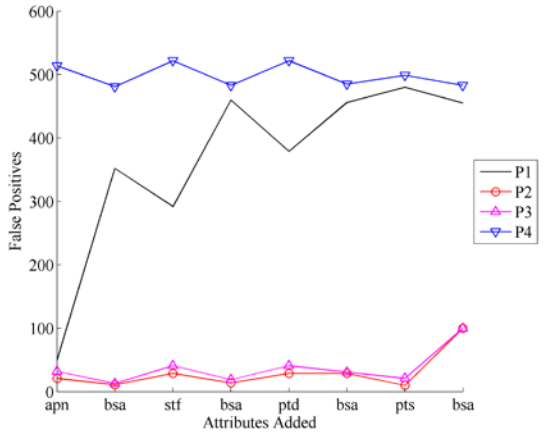
$$Re = \frac{TP}{TP + FN} \tag{3}$$

$$FM = 2\left(\frac{\mathrm{Pr} * Re}{\mathrm{Pr} + Re}\right) \tag{4}$$

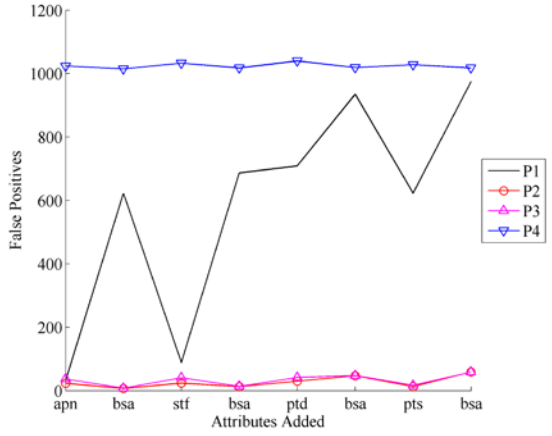
### 5.1. Base algorithms

Figure 2(a), Figure 2(b) and Figure 2(c) show results obtained for FP from differing pairings using Naïve Bayes for each dataset, 10K-30K represented by subfigures 2(a)-2(c).

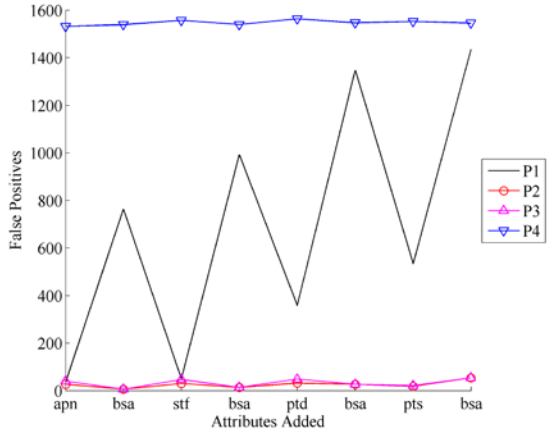
FM results for OneR and Conjunctive for each pairing do not differ from each other as features are added with each dataset. The FP and FN results are the same in this regard, essentially each pairing result is the same with the results of the individual performance measures only differing with each dataset size. An exception to this is for Conjunctive and only the first two instances where the values of FM differ. Generally as dataset size is increased the FM results are consistent with differences limited to a maximum of 0.005 for OneR and a maximum of 0.36 for Conjunctive for the aforementioned exception, while the FP and FN results increase.




2(a) 10K Dataset




2(b) 20K Dataset




2(c) 30K Dataset

Figure 2. False positives from naïve bayes results (apn = app name, bsa = bytes to source add, stf = source tcp flags, ptd = packets to destination, pts = packets to source)

Compared to OneR and Conjunctive, differences between pairings are apparent with Naïve Bayes. Results of Naïve Bayes as the dataset size is increased also presents differences between pairings.

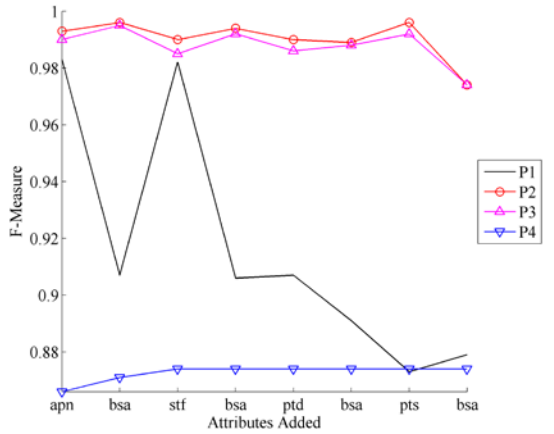
FM results for Naïve Bayes for each pairing contain differing performances, with exceptions where P2 produces better results with minor differences to P3 at a maximum of 0.005, this occurs for each dataset from 10K to 30K. P2 and P3 perform best with higher results than P1 and P4 across each dataset. P2 and P3 improve with increasing dataset size. As features are added, P1 FM results generally decrease, although not as low as P4. Decreasing results of additional features is lessened as dataset size increases. FM results of P4 produced improving performance with increasing dataset size. Additionally with increasing dataset size the difference in results as features are added decreases. Although as mentioned in relation to P1 the results of P4 does not match those of other pairings, remaining below 0.85 for each dataset size.

FP instance results for Naïve Bayes suggest a positive correlation between increasing dataset size and number of FP instances P1 and P4, but not others. For P1 and P4 there is an increase as the datasets sizes increase, albeit they follow differing patterns based on the additional features. FP instances with P1 increases with each added feature, in particular the addition of Bytes to Source as shown in Figure 2 most notably Figure 2(c), while with P4 the instances vary in a small range however at a consistently high number of FP instances.

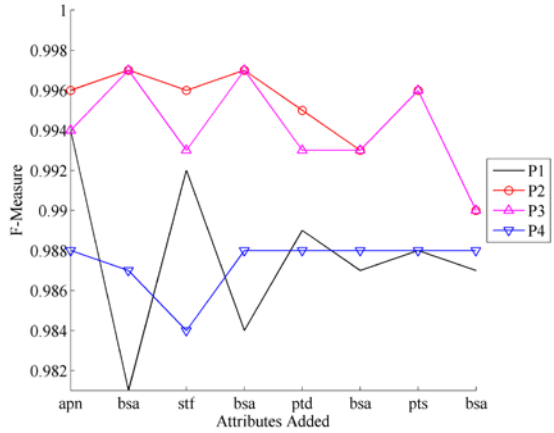
In contrast to P1 and P4, both P2 and P3 produce results, while similar to P4 in pattern, greatly differ in instances. P2 and P3 generally decrease in FP instances as dataset size increases with their largest number of instances decreasing with the increase in dataset size.

FN instance results for Naïve Bayes contain a similar pattern between P2, P3 and P4 with distinct differences to P1. P1 performs best out of all pairings where, taking into account all feature additions, instances remain low with slight increases as features are added. Other pairings, also producing low numbers of instances, do not perform as well with a lessening similarity as dataset sizes increase. P2 and P3 perform the same with no difference in FN instances. In comparison to P4, the instances from the results of P4 in some cases are the same there are more instances overall where P4 does not perform as well as the other pairings.

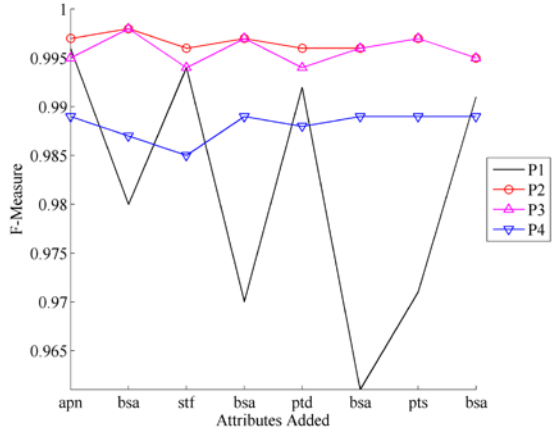
Summarizing the above performance measures, it can be observed that while Naïve Bayes produces differing results based on the pairings, the increasing dataset size and the addition of features, the same is not true for the other base algorithms, OneR and Conjunctive. Thus summarizing only Naïve Bayes it may be generally said that while P2 and P3 perform well for FM and FP it is P1 that performs well for FN. The pairings, all those except P1, perform more consistently relating to FM and FP with increasing dataset size, where their results even as features are added do not differ greatly. While true for P4 it is




3(a) 10K Dataset




3(b) 20K Dataset




3(c) 30K Dataset

Figure 3. F-measure from stackingc results (apn = app name, bsa = bytes to source add, stf = source tcp flags, ptd = packets to destination, pts = packets to source)

less positive where those results are the lowest or highest for FM and FP respectively. The performances may indicate that P2 and P3 are better able to determine *Normal* instances while P1 is better able to determine *Attack* instances, it is important to note that the scales of performance between both P2 and P3, and P1 for these respective performances are very different. Differences between pairings for FN are approximately less than 10, while for FP they increase greatly to over 200 or more, even increasing with dataset size.

## 5.2. Ensemble algorithm

Figure 3, Figure 4 and Figure 5 show results obtained for the performance measures from the differing pairings using StackingC for each dataset size, 10K-30K represented respectively by subfigures (a)-(c).

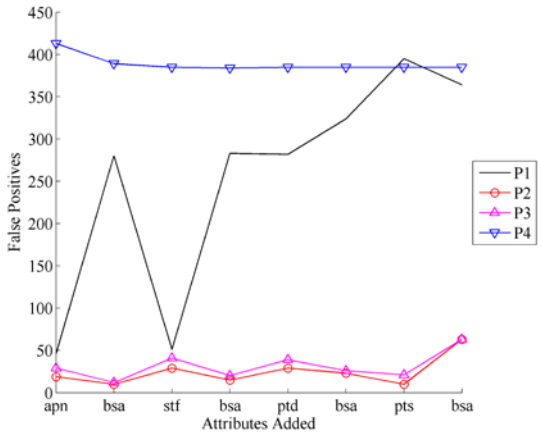
In Figure 3(a), Figure 3(b) and Figure 3(c), FM results for StackingC show P1 and P4 ultimately perform similarly, improving with increasing dataset size, P2 and P3 however consistently outperform P1 and P4 as features are added to the set.

Evidently there is a trend of P4 outperforming an increasing majority of P1 instances as dataset size increases, indicating a pairing may maintain an increasing FM performance as dataset size increases while also outperforming P1. This is most evident in Figure 3(c), where dataset of 30K shows P1 experiencing increasing difference to P2 and P3.

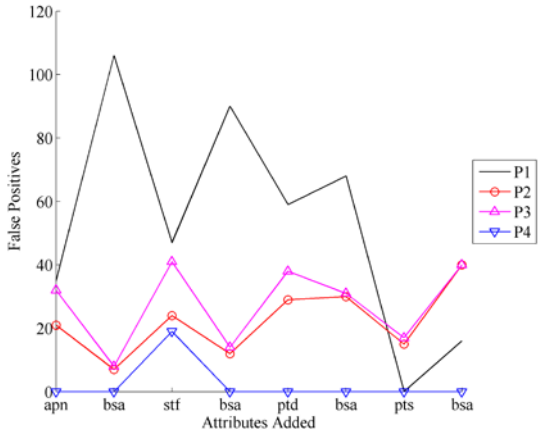
FP instance results for StackingC show somewhat of a negatively correlated performance between all pairings, that is based on the full feature set and the increasing dataset size the best performing pairing appears to change. However differences in performance are present between pairings when comparing them. After increase from 10K, where P4 performs worst, to 20K, shown by Figure 4(b), there remains a ranked performance of P4 outperforming other pairings with a value of 0 for dataset sizes 20K and 30K. P2 and P3 outperform the majority of P1 instances while P2 and P3 experience a decreasing FP instance value of 63 to 40 then 22 with increasing 10K to 30K, shown by the last instance of Figure 4(a), Figure 4(b) and Figure 4(c).

There is a difference in best performing pairing with 20K where P1 performs better but only for the final two instances. As evidenced by Figure 4(a), Figure 4(b) and Figure 4(c) the majority of instances across the three datasets indicate that the use of pairings improve the results. The Figure 4(a), Figure 4(b) and Figure 4(c) show P2 and P3 performing better than P1 where P1 experiences fluctuating values, decreasing between Figure 4(a) and Figure 4(b) then increasing between Figure 4(b) and Figure 4(c) while P2 and P3 experience a more consistent decrease as dataset size increases.

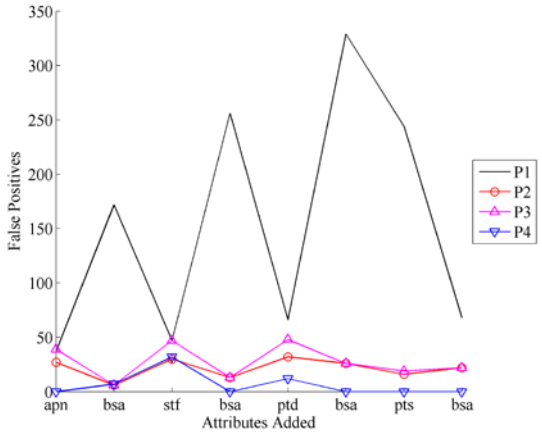
FN instance results for StackingC show a reversal in pattern of FP results, most evident by increases for P2 and P3 as dataset size increases. However there is still a difference of performance between the pairings when comparing them. For the majority of instances P1 performs the best while P4 performs the worst for each dataset size.




4(a) 10K Dataset




4(b) 20K Dataset




4(c) 30K Dataset

Figure 4. False positives from stackingc results (apn = app name, bsa = bytes to source add, stf = source tcp flags, ptd = packets to destination, pts = packets to source)

Specifically for dataset size 10K at the total feature set, P1 to P4 all produce similar if not the same results as each other. Specifically, P1 to P3 achieve 10 and P4 achieves 8 FN instances respectively. However these are results for the total feature set while for majority of subsets prior, P1 performs best followed by P2 and P3.

A similar pattern is present for dataset size 20K where P1 performs best for the majority of instances followed by P2 and P3, the majority of instances for P1 to P3 being less than 10 and P1 being 5 or less.

This pattern evidently repeats for dataset size 30K with a more distinct difference in performance between P1, P2 and P3, particularly with the full feature set where P1 achieves, 8, less than half of those instances, 19, achieved by P2 and P3. Even greater than the instances achieved by P2 and P3 are those instances achieved by P4 for the same feature set.

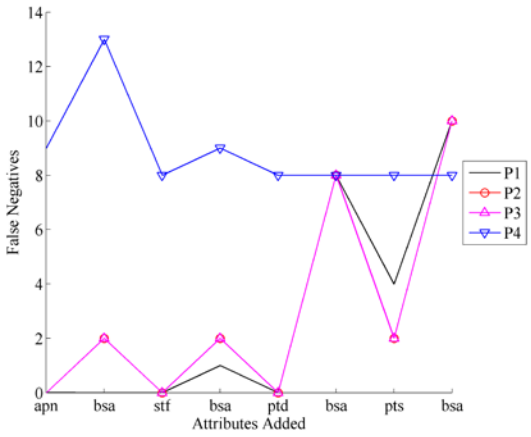
From dataset sizes 20K to 30K number of instances increase from 66 to 88, the latter is the FNs lower bound. Indicating the degree to which P4 performs the least well for the dataset size 30K, this is also the case for 10K and 20K where, while not as high in number of instances, P4 is outperformed by the other pairings.

Summarizing the above three measures it can be observed that there is no best performing pairing for all three. Here the criteria for best performing is, a pairing must achieve best performance for each measure uniquely for some majority. While a pairing may perform best for two of the measures, if an existing pairing shares best performance of the third there is no best performing pairing.

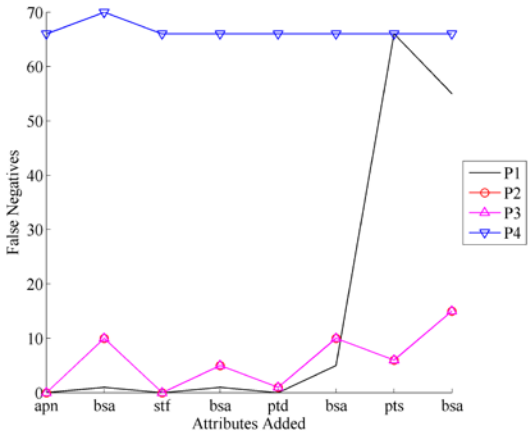
Instead of a single best pairing, each performs best at respective performance measures. For FM, P2 and P3 perform best while P4 performance improves with increasing dataset size, as shown in Figure 3(c).

For FP instances, P4 performs best outside of dataset size 10K, most notably in dataset size 30K, as shown in Figure 4(c), most notably against P1, consistently high, where P4 performs well. In a reversal to FP instances, the best performing pairing for FN instances is P1. Consequently, usage of pairings evidently affect analysis performed by base algorithms, in some cases increasing and others decreasing performance in a way as to change apparent diversity of the base algorithms, in such a way as to improve distinctions the meta classifier is capable of making. FM and FP instance values improve with pairing, the feature pairings being a potentially beneficially distinctive feature for detection of *Normal* instances. Correlation between FM and IGR vary for each dataset size and each pairing.

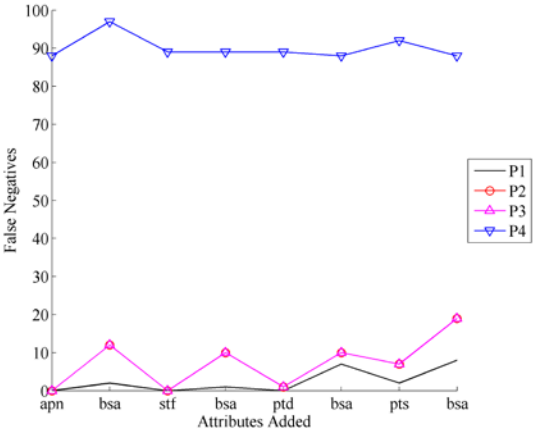
From Table 3 a strong positive correlation can be observed for P1 when probability in entropy was considered to be dependent; this positive correlation is shown to occur for all other pairings. This however changes as the dataset sizes increases to 20K where, while still all positive, P1 to P3 experience a weaker correlation. This differs in results observed for P4 which experiences a stronger correlation but not as strong as P1 had previously.




5(a) 10K Dataset




5(b) 20K Dataset




5(c) 30K Dataset

Figure 5. False negatives from stackingc results (apn = app name, bsa = bytes to source add, stf = source tcp flags, ptd = packets to destination, pts = packets to source)

This continues with dataset size increase to 30K where P1 to P3 vary further, P1 and P3 becoming negatively correlated and P2 remaining positive but decreasing in strength.

With increase to 30K, as with 20K, P4 differs from other pairings, in this case correlation remains positive as before but decreases in strength, although not to the same degree that P2 does. An interesting observation is that of P4, while not the highest possible value, for which correlation does not differ greatly and instead is the only pairing which maintains a consistently positive correlation remaining similar in value, experiencing a difference between its lowest and highest values of only 0.03, approximately.

From Table 4 it can be observed that for all but 10K with P1 correlation is negative, the strength of which reaches a maximum of -0.59 for 10K with P4 from a minimum of -0.47 for 10K with P3, after which correlation of each pairing has a small difference to the others as datasets size increases from 10K to 20K to 30K. With such a consistent and small difference in the correlations of each pairing it may be that correlation with independent entropy provides no clear distinction between pairings.

### 5.3. Bytes to Source

Additions of each feature were followed by addition of Bytes to Source for the previous feature set with the feature set that followed including the addition of a different feature without Bytes to Source. An example is the first addition of Bytes to Source where that feature set would include App Name and all previous features, the feature set that follows the first addition includes Source TCP Flags but not Bytes to Source. Thus the alternating presence/performance of Bytes to Source could be analyzed.

The reasoning was determining importance of that feature where number of bytes sent to a destination over time may be of some importance for analysis. For OneR and Conjunctive, as there is no difference in change for any pairing there is no apparent importance in those cases.

For Naïve Bayes, there is a difference in change that may be explored. Briefly covering FM, FP and FN it is shown that inclusion of Bytes to Source decreases performance primarily with P1 while other pairings improve in some cases. Increasing dataset size seemingly increases differences between those feature sets with and those without Bytes to Source, although it is also shown that those feature sets with Bytes to Source still improve, as apparent in comparisons of performance between FM with 10K and 30K dataset size.

For StackingC, the most apparent changes from addition of Bytes to Source are those for P1, decreasing in performance. Other pairings benefit from the addition in some cases of FM and FP.

## 5.4. Correlation

Correlation of FM and IGR where both measure a feature set in a differing way is meant as a combined measure of both individual measures, taking into account both to provide a resultant measure. FM determines the classification performance, in this case StackingC, on feature sets. IGR determines the 'information gained' or relevance of feature sets.

The IGR values, as previously described here, indicate differences between considering and calculating entropies to be independent or dependent in relation to differing pairings. Results of the calculation of IGR show that differences between the IGR values for each pairing differ for some early feature additions, however after these additions the IGR becomes very similar with only negligible differences. The noticeable differences occur for longer where IGR is calculated as independent.

Both FM and IGR are taken as independent of each other also while representing some differing aspects of the feature sets. From this perspective a correlation, if any, of these two values may provide a measure or part of a measure. This measure may provide some selection as to a beneficial feature set by taking into account the differing measures of performance and relevance as provided by FM and IGR respectively.

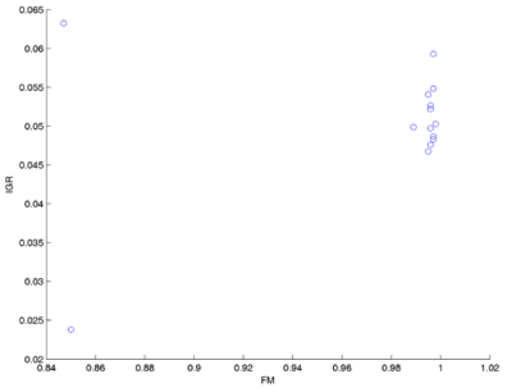
The correlation may define, if any, a potential 'agreement' between the two values. Any 'agreement' may provide some decision as to selection of a feature set among the others. This selection may differ across differing pairings as a result of differing values for FM, and their resultant correlation, for each pairing. The correlation between FM and IGR follows the same analysis of the correlation of any two values.

As presented in Table 3 and Table 4, the calculated values of IGR showed that between independent and dependent, the latter entropy provides a clearer distinction between each of the pairings. The distinction between pairings when IGR is calculated as independent is small in the majority of cases.

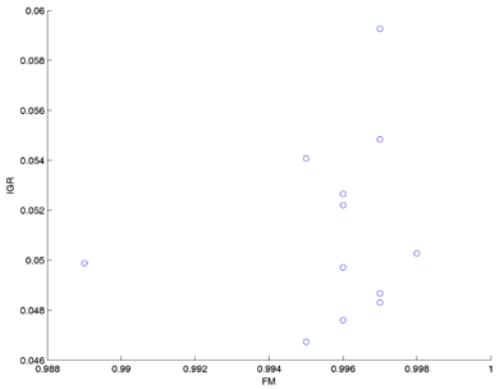
When IGR is calculated as dependent, two pairings remain positive as the dataset size increases, these pairings are P2 and P4. Although P2 decreases by 0.1 from 0.5 while P4 differs on a range of 0.03 between 0.55 and 0.58 thus P4 may remain close to those values or increase as the dataset increases.

The correlations over all feature sets shows some positive and some negative results as well as a range of weak to moderate to strong results. However a number of comparisons made between the correlation coefficient, as well as its expected pattern, and data points as represented in a scatter plot, indicate that while seeming to show patterns of correlation for each pairing the correlation coefficients do not fully fit to their respective data points.

Individual data points of a scatter plot represent feature sets, the majority of which grouped together, at least in relation to pairings P2, P3 and P4. For the aforementioned pairings there exist outliers from the groupings, each time these outliers are the two smallest feature sets and those with the lowest FM value. Removal of these outliers changes the correlations of their respective pairings and further defines the shape of the grouping as being primarily vertical where differences between data points in relation to their FM value are small. An example is shown in Figure 6 where removal of outliers decreases correlation from 0.30 to 0.16 and further evidences a less correlated pattern.




6(a) P2 30K with outliers




6(b) P2 30K without outliers

Figure 6 (a), (b). P2 scatter plots showing data point patterns (with/without outliers)

As the remainder of data points, after removal of the outliers, are grouped together in a more vertical pattern it can be determined that a linear correlation does not fit to the data points where the slope would follow the more vertical direction. Taking into consideration the earliest and latest few data points shows that typically while there is some underlying

pattern in some data points where, typically, low values of FM follow low values of IGR and, typically, high values of FM follow high values of IGR, although not always the case. However this is certainly only underlying and not strictly found by matching the scatter plot with calculation of the correlation. It is more a general pattern that exists, especially, where the highest FM value is not associated with the highest IGR value. Thus while correlation in these instances may well describe a few of the data points, it is not an accurate description which is followed by a suitable majority of the data points. Further to where correlation of FM and IGR in its presented method may not directly describe the data the pairings presented herein produce a number of data points that fall on a vertical line of a single FM where IGR varies with each feature set.

It is thus considered that the use of linear correlation or a linear correlation of FM and IGR, as presented, may not be adequate to select best feature sets from the subset. A method of ranking each value of FM and IGR and correlating as such has no direct benefit. Thus an alternative should be explored for a more representative relationship between FM and IGR or a selection of some or all differing measures.

Table 3. F-measure and information gain ratio correlation (dependent)

| Pairing | Dataset 10K | Dataset 20K | Dataset 30K |
|---|---|---|---|
| P1 | 0.92 | 0.09 | -0.20 |
| P2 | 0.50 | 0.40 | 0.30 |
| P3 | 0.71 | 0.10 | -0.24 |
| P4 | 0.55 | 0.58 | 0.57 |

Table 4. F-measure and information gain ratio correlation (independent)

| Pairing | Dataset 10K | Dataset 20K | Dataset 30K |
|---|---|---|---|
| P1 | 0.13 | -0.54 | -0.53 |
| P2 | -0.52 | -0.54 | -0.58 |
| P3 | -0.47 | -0.51 | -0.56 |
| P4 | -0.59 | -0.53 | -0.56 |

## 6. Conclusion

The hypotheses proposed that combination and pairing of features should improve algorithmic performance, such as reduced FP rate or improved FM. Results of FM and FP show where P2, P3 and P4 achieve better results, and the hypothesis may be supported. This indicates the use of those particular pairings increase accuracy of classification. P2 and P3 performing best followed by P4. Also the ability of the ensemble to differentiate those instances that are actually *Normal*, this may be evident by P4, followed by P2 and P3, reducing FPs greatly compared to P1.

However, FN results show that P2, P3 and P4 did not perform as well as P1, indicating that while able to better identify *Normal* instances, they are less able to identify *Attack* instances, this being particularly true for P4. For FN values, P1 was observed to perform the best, followed by P2 and P3 then P4 in that order.

Comparing Naïve Bayes and StackingC FP results for P4 seemingly presents the benefit of the ensemble used. In Figure 2 and Figure 4 P4 performs least well and best respectively. Potentially indicating some diversity between base algorithms provided a benefit, however taking into account discussed similarities between pairings for OneR and Conjunctive it is noted that change of one or both of those algorithms may benefit the ensemble, as such it is a potential future focus.

Additionally where proposed that correlation of FM and IGR may provide some selection as to beneficial feature sets, it has been shown that the current correlation of those measures yields no clear overall pattern for use as selection of well performing feature sets. While the correlation may provide some indication as to a pattern between the measures, comparisons against a scatter plot show that a linear correlation does not fully and accurately describe existing patterns between measures. Further work relating to this method of correlation for selection may be possible with alternating usage or form of measures and the way they are calculated or differing the measure of correlation in some way, potentially using a nonlinear approach.

Overall, pairings of heterogeneous IPs and Ports as well as certain feature subsets above a number of features seemingly reinforce and improve StackingC's ability to detect *Normal* events.

A future approach may be utilizing pairings with base algorithms. Potentially achieving diversity by use of differing pairings for each base algorithm, similar to how they may receive differing dataset instances. Thus, they may be able to provide more accurate results and further reductions of FPs and FNs by differing results from each base algorithm where they may each perform best, given specific pairings. As no particular pairing was found to perform best for all measures, this may provide motivation towards a multi-objective approach for further work. Further analysis of feature subsets, including present and number of features, is a potential future focus.

## 7. References

[1] A. Kumar and E. Fernandez, “Security Patterns for Intrusion Detection Systems,” … Softw. Archit. Patterns (LACCEI …, 2012.

[2] A. Balon-perin and B. Gamback, “Ensembles of Decision Trees for Network Intrusion Detection Systems,” Int. J. Adv. Secur., vol. 6, no. 1, pp. 62–77, 2013.

[3] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, “Network Anomaly Detection: Methods, Systems and Tools,” IEEE Commun. Surv. Tutorials, vol. 16, no. 99, pp. 1–34, 2013.

[4] K. A. García, R. Monroy, L. A. Trejo, C. Mex-Perera, and E. Aguirre, “Analyzing log files for postmortem intrusion detection,” IEEE Trans. Syst. Man Cybern. Part C Appl. Rev., vol. 42, no. 6, pp. 1690–1704, 2012.

[5] R. Harang, “Bridging the Semantic Gap: Human Factors in Anomaly-Based Intrusion Detection Systems,” Netw. Sci. Cybersecurity, vol. 55, pp. 15–38, 2014.

[6] S. Axelsson, “The Base-Rate Fallacy and its Implications for the Difficulty of Intrusion Detection,” Proc. 6th ACM Conf. Comput. Commun. Secur. - CCS ’99, pp. 1–7, 1999.

[7] J. M. Estevez-Tapiador, P. Garcia-Teodoro, and J. E. Diaz-Verdejo, “Anomaly detection methods in wired networks: A survey and taxonomy,” Comput. Commun., vol. 27, no. 16, pp. 1569–1584, Oct. 2004.

[8] J. Han, M. Kamber, and J. Pei, Data Mining, Third. Morgan Kaufman, 2012.

[9] D. Bolzoni, S. Etalle, P. Hartel, and E. Zambon, “POSEIDON: A 2-tier anomaly-based network intrusion detection system,” in Proceedings - Fourth IEEE International Workshop on Information Assurance, IWIA 2006, 2006, vol. 2006, pp. 144–156.

[10] P. M. Mafra, V. Moll, J. Da Silva Fraga, and A. O. Santin, “Octopus-IIDS: An anomaly based intelligent intrusion detection system,” in Proceedings - IEEE Symposium on Computers and Communications, 2010, pp. 405–410.

[11] E. Menahem, L. Rokach, and Y. Elovici, “Troika - An improved stacking schema for classification tasks,” Inf. Sci. (Ny)., vol. 179, no. 24, pp. 4097–4122, Dec. 2009.

[12] P. a. Estévez, M. Tesmer, C. a. Perez, and J. M. Zurada, “Normalized mutual information feature selection,” IEEE Trans. Neural Networks, vol. 20, no. 2, pp. 189–201, 2009.

[13] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, “McPAD: A multiple classifier system for accurate payload-based anomaly detection,” Comput. Networks, vol. 53, no. 6, pp. 864–881, 2009.

[14] L. Breiman, “Bagging predictors,” Machine Learning, vol. 24, no. 2. pp. 123–140, Aug-1996.

[15] D. H. Wolpert, “Stacked generalization,” Neural Networks, vol. 5, no. 2. pp. 241–259, 1992.

[16] K. M. Ting and I. H. Witten, “Issues in stacked generalization,” J. Artif. Intell. Res., vol. 10, pp. 271–289, 1999.

[17] A. Seewald, “How to Make Stacking Better and Faster while also Taking Care of an Unknown Weakness,” in Proceedings of the Nineteenth International Conference of Machine Learning, 2002, pp. 554–561.

[18] D. Ariu, R. Tronci, and G. Giacinto, “hmmpayl: An intrusion detection system based on Hidden Markov Models,” Computers & Security, vol. 30, no. 4. pp. 221–241, 2011.

[19] W. Yassin, N. I. Udzir, and Z. Muda, “anomaly-based intrusion detection through k- means clustering and naives bayes classification,” in Proceedings of the 4th International Conference on Computing and Informatics, ICOCI 2013, 2013, no. 049, pp. 298–303.

[20] A. Shiravi, H. Shiravi, M. Tavallaee, and A. a. Ghorbani, “Toward developing a systematic approach to generate benchmark datasets for intrusion detection,” Comput. Secur., vol. 31, no. 3, pp. 357–374, May 2012.