
Article:
Wu, Z., Dong, Y.-N., Wei, H.-L. orcid.org/0000-0002-4704-7346 et al. (1 more author) (2020) Consistency measure based simultaneous feature selection and instance purification for multimedia traffic classification. Computer Networks, 173. 107190. ISSN 1389-1286

https://doi.org/10.1016/j.comnet.2020.107190

Manuscript accepted for publication by Computer Networks (2020-02-28)

Consistency measure based simultaneous feature selection and instance purification for multimedia traffic classification

Zheng Wu[a], Yu-ning Dong[a,*], Hua-Liang Wei[b], Wei Tian[a]

[a] College of Telecommunications & Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
[b] Department of Automatic Control & Systems Engineering, University of Sheffield, UK

Abstract

With the increase of multimedia traffic, the implementation of fast and accurate classification has become an important issue. Besides, a manually captured dataset contains certain noise and mislabeled instances, which influences the accuracy of classifier to some extent. Motivated by these observations, a novel feature selection and instance purification (FS&IP) method based on consistency measure is proposed. It utilizes a linear consistency-constrained algorithm for feature selection. In each round of iteration, it removes the instance with the minor labels in every pattern subset. Our method has three desirable properties: 1) it can simultaneously achieve feature selection and data purification. 2) when purifying instance, it doesn't need to annotate the noisy instance with learned labels; that is because it is an unsupervised method in terms of data purification. 3) through data purification, it is able to obtain a minimal feature subset on condition of maintaining accuracy. In addition, the proposed method can be used to discover a new discriminative feature based on linking behaviors called the flow fragment ($F-Frag$), which can reflect important information among the complex and multitudinous packet communication behaviors. The experimental results over six different datasets demonstrate the advantages of the proposed technique compared to six existing methods, and the discriminative power of the new flow fragment feature.

Keywords: Traffic classification, Feature selection, Instance purification, Flow fragment.

1. Introduction

One of the key components for supporting the QoS (Quality of Service)-enabled Internet is the provision and management of robust and automatic traffic classification (TC) [1, 2]. As a core part of QoS-enabled Internet, TC can be employed by the Internet service providers to identify various traffic categories to provide different QoSs for various types of traffic. However, with the increase of the volume and categories of multimedia traffic [3], it has become a challenging task to ensure the accuracy and effectiveness of TC. To guarantee the overall acceptability of an application or service perceived by the end-user, an effective and fine-grained multimedia classification system is necessary.

Due to the dynamic use of port numbers, the emergence of traffic encryption and encapsulation, and the concern of privacy protection in recent years, most of the port-based and deep packet inspection (DPI)-based TC techniques [4] becomes inapplicable anymore. Since machine learning (ML)-based methods are able to address the aforementioned issues effectively, they have become the signs of future success in TC [5]. Most ML-based methods construct the knowledge system through extracting the flow-level statistical information from transport layer, which is divided into supervised methods and unsupervised methods according to whether there is label information. However, due to the giant volume of traffic, the ML-based methods need much more time to train a model. Besides, the big data can easily cause the overfitting problem and degradation of time performance [6].

To address the problems of high time overhead and overfitting on high-dimensional dataset, model reduction is critical [7]. Therefore, some feature selection methods for traffic classification have been devised to improve the effectiveness of classification system. The feature selection algorithms are mainly split into the following three categories: filter, wrapper and embedded methods [8]. The feature subset selected based on wrapper and embedded methods is related to the subsequent predetermined classifier, hence, the features subset may not be suitable for other classifiers. Filter methods utilize the natural characteristics (such as correlation and information entropy) of data to rank and further select features [9]; therefore, the feature subset is not related to the predetermined classifier. At the same time, filter methods are usually more efficient than the wrapper and embedded methods on large-scale dataset.

1.1. Challenges and motivations

Most current feature selection methods determine the features in accordance with their relevance with the labels and the redundancy between features. For example, the correlation-based feature selection (CBF) is a widely used method, which considers both the feature-label and feature-feature correlations

*Corresponding author
Email addresses: wuzheng2310174030@163.com (Zheng Wu), dongyn@njupt.edu.cn (Yu-ning Dong), w.hualiang@sheffield.ac.uk (Hua-Liang Wei), tianw@njupt.edu.cn (Wei Tian)

during evaluation of features. However, this kind of method may fail to discover some useful feature combinations that may be significantly relevant and useful for a classification task, but each of the associated individual feature is of low relevance to some of the classes. One of the solutions is the consistency based methods [10], which relies on consistency measure to search for the best subset with high consistency score. However, the speed of most consistence-based method becomes slow when dealing with large volume of instances. Therefore, how to implement faster and more effective feature selection for large-scale and ceaseless traffic data is a challenging task.

Another key challenge is how to effectively purify traffic. When capturing traffic data, there exist complex factors which may influence the purity of the captured dataset, such as the variation of network conditions and the occurrence of some wild flows. These factors can cause some flows to be mislabeled. Even in the benign network conditions, typically, a non-negligible percentage of flow would be considered suspicious. These mislabeled instances are prone to misguiding the classifier to make an inaccurate decision margins in the training stage [11]. Therefore, data purification is important to the classification performance.

Motivated by the above considerations, this paper aims to develop new techniques to conduct feature selection and dataset purification simultaneously for more accurate multimedia traffic classification.

1.2. Key contributions

This paper makes four major contributions as follows.

- It designs a novel algorithm for joint feature selection and instance purification. The proposed algorithm has three advantages over other methods. First, it can achieve feature selection and data purification simultaneously. Second, as an unsupervised learning method (in terms of data purification), it does not need to know or learn the noisy instance labels. Finally, benefited from data purification, it can obtain a smaller feature subset while maintaining classification accuracy.

- A discriminative feature called flow fragment ($F-frag$) is discovered based on networks link behavior. Moreover, to measure the performance of the traffic features, a feature stability (FS) is proposed. The experimental results show that this new feature can significantly improve the accuracy of a classifier and have a good performance in terms of FS. This is an important finding, in that it provides a more insightful measure for classifier design.

- An integrated framework for multimedia traffic classification is developed, including data capturing, feature extraction, feature preprocessing, feature selection and classification. Among which, a traffic trace of 424.67GB size is captured, which comprises six multimedia categories, and 44 QoS-aware flow-level features are extracted to characterize multimedia flows.

- By using our own proposed algorithm and six other algorithms, extensive experiments are carried out over five UCI datasets and our multimedia traffic dataset. These methods are compared with the proposed method in terms of accuracy, running time and the subset size ratio. Moreover, the feature stability is proposed to evaluate the goodness of each feature. The experimental results suggest that the proposed method is better than other ones.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 defines the problem. Section 4 presents the overall framework of TC that comprises five modules. Next, Section 5 presents the novel algorithm for joint data purification and feature selection, and a new feature – flow fragment. The experiments and comparisons with existing methods are done in Section 6. Section 7 analyzes the experimental results. Finally, Section 8 concludes the paper.

2. Related work

Feature selection is a crucial technique for Internet traffic classification. There are much work on feature selection for traffic classification. In general, it includes three important categories [12], which are the wrapper, filter and embedded methods respectively. As mentioned earlier, filter methods generally behave more efficiently for large-scale data [13]. Therefore, this paper mainly focuses on the filter methods.

Many filter algorithms have been developed, including the information theory-based methods [14], similarity-based methods [15], sparse learning-based methods [16] and so on. Furthermore, great efforts have been made for feature selection in the TC field. For example, Ambusaidi et al. in [13] proposed a features selection algorithm based on mutual information and used the rank search method to select the best subset for building an intrusion detection system. Senliol et al. [17] presented a fast correlation-based filer (FCBF) method to separate and prioritize the sensor data from the multimedia traffic. Such an FCBF method is implemented using data segmentation, where the original data are split to an appropriate number of segments. The relevance of each feature for representing the target class is evaluated in each of the segments. The importance of each feature is then determined according to their overall performance. Note that, FCBF only considers the correlation relation between individual features, and does not consider the effects of the interaction of features.

Dong et al. [1] found some useful flow-level features and exploited the consistency-based method and information theory-based method jointly to select the best feature subset for achieving the fine-grained video traffic classification. It is known that the existence of noise in data may degrade the performance of a classifier especially when the trained classifier is applied to new data. To achieve more accurate selection of feature subsets, Adil et al. [18] put forward three evaluation criteria to assess feature selection from three aspects, and further developed an integrated feature selection technique by combining the results of five algorithms.

Although it can get more accurate results, it requires considerable computation power and time. Dong et al. [19] introduced a feature selection method based on a heuristic search algorithm (called RFPSO) to mitigate the problem of highly dimensional traffic classification.

Among the aforementioned methods, the consistency-based algorithms have many advantages over other methods [20], such as more efficiency of finding feature interaction and reduction of feature redundancy. Many feature selection methods have been developed based on consistency. Liu et al. [21] devised the INTERACT method, which uses the symmetric uncertainty to rank features and then evaluates individual features by the consistency contribution (CC). Although this feature selection method is fast, the resulting feature subset usually contains a relatively larger number of features. Following the INTERACT, Shin and Xu [22] introduced the linear consistency constrained (LCC) and complete consistency constrained (CCC) algorithms. Besides, the steepest-descent consistency-constrained algorithm (SDCC) was presented in [21], which exploits the monotony of consistency and applies the steepest descent method as search strategy. However, it needs more time to search the optimal subset. Shin et al. [23] presented an extended definition of consistency and theoretically proved that the binary measure has the best sensitivity among fifteen measures, based on which, the CWC algorithm developed based on a binary measure was presented. To accelerate the speed of CWC and LCC, the binary search is utilized in [24] to make the two algorithms suitable for dealing with big data.

In terms of instance purification, the editing neighbor nearest (ENN) [25] utilises the 'nearest neighbor rules' to remove the noisy instances in order to increase the classifier's generalization ability. However, this method only operates in continuous space and the selection of neighbors for nearest neighbor classifier needs additional consideration. In addition, ENN and its variants only work when a very small amount of noise is presented.

Different from other studies in the literature, by considering possible noisy samples and outliers, this paper proposes a novel approach that is able to realize the feature selection and data denoising simultaneously. For illustration, the main notations used in this article are summarized in Table 1.

Table 1: Main notations used in this article.

Symbol	Description
$a(S)$	The feature vector of instance a with regard to feature subset S .
C	The class label set, namely, $\{c_1, c_2, \cdots, c_n\}$.
D	The knowledge system, a finite discrete dataset.
F	A set of features, namely, $\{f_1, f_2, \cdots, f_l\}$.
ε	A pattern of D.
I	The instance set of D.
S, G	The feature subset of F.
I_ε	The pattern subset $\{a \in D \vert a(S) = \varepsilon\}$.
L_ε	The pattern count list with respect to ε.
$D_{S=\varepsilon, C=y}$	$\{a \in D \vert a(S) = \varepsilon \wedge a(C) = y\}$
$T[i]$	The ith element of the vector T.
σ	The consistency threshold.
$\vert A \vert$	The cardinality of set or vector A.

Table 2: An example of the problem of feature interaction and redundancy.

f_1	f_2	f_3	C
1	0	1	0
0	1	1	1
1	0	0	1
0	1	0	0
1	0	1	0
0	1	1	1
0.081	0.081	0	$I(f_i; C)$

3. Problem statement

For simplicity of description, the feature selection method based on information entropy (IE) is used as an example here. Let $I(F; C)$ represent the mutual information between feature set F and the corresponding label set C. Then, the sum of relevance (SR) with respect to $I(F; C)$ is defined as:

$$SR(F = \{f_1, f_2, \cdots, f_l\}) = \sum_{i=1}^{l} I(f_i; C), \tag{1}$$

where SR represents the sum of relevance of the feature set with labels. The classical examples of SR-based feature selection algorithm include the Relieff algorithm [26], FCBF [17] and so on.

Table 2 gives an example showing a knowledge system with feature set $\{f_1, f_2, f_3\}$ and the class labels $C = \{0, 1\}$.

This paper aims to tackle the following three key problems:

1. Feature interaction. In many situations, a single feature may be of low relevance or importance to the label, but when it interacts with other features, the interactions could play a significantly important role in indicating the label. For example, the class label in Table 2 can be completely determined based on the combination of f_1 and f_3 - say using the XOR relationship between them, but note that $SR(f_1, f_2) \geq SR(f_1, f_3)$ and $I(f_3, C) = 0$ holds. According to the SR rule, the feature set f_1, f_2 is better than f_1, f_3.

2. Feature redundancy. As shown in Table 2, f_1 and f_2 represent the same knowledge with respect to class label, because their values are the operation of pointwise inversion. Therefore, one of the two features can be removed without loss of any information. However, $SR(f_1, f_2)$ is bigger than either $SR(f_1)$ or $SR(f_2)$. The SR-based methods inevitably deem the subset of $\{f_1, f_2\}$ and are better than either f_1 or f_2 separately.

3. Noisy labels. Noisy labels are a significant problem in the real-world data collection in TC [27]. The most straightforward and accurate method to acquire labelled data is to only allow one category of traffic flow to pass through the capturing device. However, even in benign network conditions, there are typically a non-negligible percentage of flows which would be considered suspicious. Due to these influencing factors (e.g., network instability,

some mixed flows), some instance labels tend to drift to an adjacent category from the actual one. Therefore, some noisy instances in the decision boundaries may cause the performance degradation of classifiers. In the example shown in Figure 1, in the boundary area, there exist some noisy points denoted by red dots. During data learning, the classifiers will be easily misguided, unless necessary means are taken. Unfortunately, the mislabeled instances can not be distinguished from actual labeled ones in advance. Therefore, it is desirable that a method can remove the noisy instances without any pre-training.

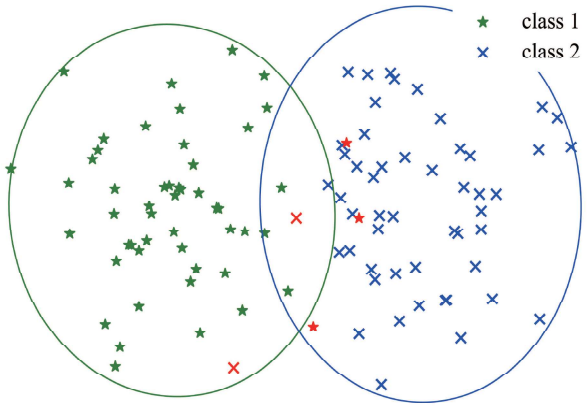


Figure 1: Some noises at the boundary between classes in a binary dataset.

4. The overall framework

The framework of the proposed method is shown in Figure 2, which gives the bird's view to the overall procedure of our work. The blocks with red border and slashes present the major works of this paper.

The first step is the data collection. The data flowing through the network interface card (NIC) was captured in the campus network of Nanjing University of Posts and Telecommunications from the beginning of September to the end of October of 2018. To guarantee the ground truth of dataset, the traffic flow traces were captured over three time periods, that is, morning, afternoon and evening. The captured data was eventually stored in the database in the five-meta format, and the data include the packet arriving time, source IP address, destination IP address, packets size, and protocol. Though two-month data capturing, 1636 flows were obtained comprising the following six categories: Internet live video (ILV), game, streaming video (SV), pear to pear video (P2PV), conversational video (CV) and web browsing (WB). The detailed information of the dataset is listed in Table 3.

Table 3: Description of our multimedia dataset.

Category	#Flow	Size (Gb)	Application
ILV	284	147.28	Sopcast[1], CNTV[2]
GAME	476	5.37	Dota[3], FWJ[4], NZ[5]
SV	360	182.90	Iqiyi[6], Youtube[7]
P2PV	206	82.97	Xunlei video[8]
CV	126	3.21	Skype video[9], WeChat video[10]
WB	184	2.94	Sina[11], Csdn[12]

The next step is the feature extraction module. To better utilize the statistical knowledge, 44 QoS-aware flow-level features were extracted simultaneously. Furthermore, these 44 features can be divided into the three groups of downlink, uplink and datalink features. The downlink features are the downstream statistical characteristics associated with local IP. Similarly, the uplink features are the upstream statistical characteristics associated with local IP. The datalink features are the statistical characteristics of bi-direction flows. All the 44 features are summarized at Table A1 in the Appendix.

Then, a feature preprocessing procedure is carried out which includes data normalization and data discretization. It is noted that some of the extracted feature have a large range of values (e.g. the downlink packet rate ranging from 0.179 to 48183 bytes/s), therefore, z-score method [28] is used for data normalization so that they have the same order of magnitude. Previous work [29, 30] has discovered that data discretization is conducive to boost the effectiveness of back-end classifier, and elevates the classification accuracy. Thus, discretization is carried out. It uses an equal-width-interval algorithm [31] to separate the varying range of each variable (feature) into N intervals (N=10 in the present study). After data preprocessing, it uses the proposed feature selection method to select the best feature subset and purify data. At the end, multiple classifiers are used to identify the multimedia traffic. The processed features are fed into classifiers to identify the traffic types. The classification results can be used in subsequent module, such as QoS mapping, resource allocation, and network surveillance.

5. Methodology

Thereinafter, a pattern is part of an instance without class label. It is a vector of real valued features in the feature subset [23]. Let D be a finite knowledge system consisting of a

[1] http://www.sopcast.com/
[2] http://tv.cntv.cn/
[3] https://www.dota2.com.cn/index.htm
[4] http://xyq.163.com/
[5] https://nz.qq.com/
[6] https://www.iqiyi.com/
[7] https://www.youtube.com/
[8] http://x.xunlei.com/
[9] http://skype.gmw.cn/
[10] https://weixin.qq.com/
[11] https://www.sina.com.cn/
[12] https://www.csdn.net/

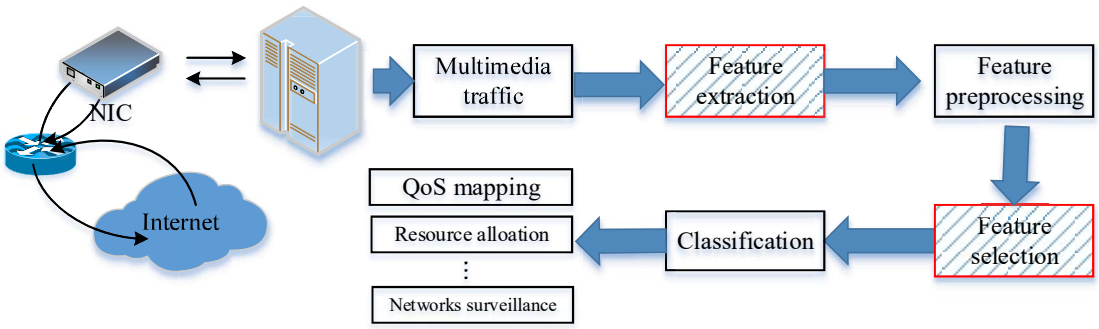


Figure 2: The proposed framework of video traffic classification.

feature set $F = \{f_1, f_2, \cdots, f_l\}$ and a class variable C. According to the pattern, the instance set can be divided into pattern subsets because some instances in dataset D share the same patterns. Specifically speaking, a pattern subset $I_\varepsilon (I_\varepsilon \subseteq D)$ is a combination of instances defined as follows: the value of an instance in the feature subset S is equal to ε. It is formally defined as:

$$I_\varepsilon = \{a \in D | a(S) = \varepsilon\}, \tag{2}$$

where $a(S)$ denotes the vector of feature values for an instance a from feature subset S.

The pattern subset can be further partitioned into smaller subsets I_ε^y by class labels as follows:

$$I_\varepsilon^y = \{a \in D | a(S) = \varepsilon \wedge a(C) = y)\}. \tag{3}$$

Thereby, the inconsistent count for pattern ε is defined as:

$$IC(\varepsilon) = |I_\varepsilon| - \max_y \left|I_\varepsilon^y\right|. \tag{4}$$

With regard to the feature subset S of dataset D, there is a finite countable set of patterns $\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_p$. The inconsistency rate equals to the ratio of inconsistent count of all the patterns to the number of all instances of D. Formally,

$$ICR(\varepsilon) = \frac{\sum_{1 \le i \le p} IC(\varepsilon_i)}{|I|}, \tag{5}$$

where I is the whole instance set of dataset D.

For simplicity, $ICR(S; C)$ is used to represent the process of calculating the inconsistency rate for feature subset S, where $S \in F$, and C is the corresponding class label set.

The inconsistency rate has the following properties [23]:

1. $ICR(S, C) = 0$, if and only if, F determines C.
2. If $S \subseteq G$, $ICR(S, C) \ge ICR(G, C)$ holds.
3. $ICR(S, C) \le \frac{n-1}{n}$, where n is the number of classes.

As mentioned above, the consistency-based methods have some advantages. First, the consistency rate can be used to discover the interacting feature sets while removing redundant features. Second, the inconsistent rate increases with the decrease of features according to Property 2. It is a monotonous function, which can be utilized to speed up the process of feature selection.

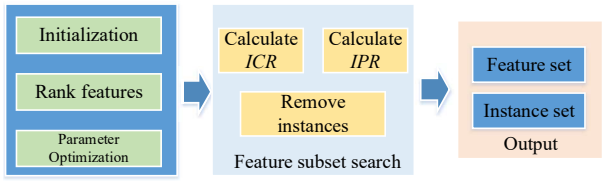


Figure 3: Flowchart of proposed algorithm.

5.1. Algorithm

This subsection presents a novel method based on inconsistent rate which can be used to select a feature subset while purifying the instance (referred as FS&IP). Figure 3 presents the overall procedure of the proposed method. The first step comprises initialization, feature ranking and parameter optimization. After that, the feature subset search is carried out, where the ICR and IPR are calculated and meanwhile noisy instances are removed at each iteration. Finally, the selected feature set and purified instance set are calculated as the output. In the following, the technical details of each step are presented.

Pattern count vector (PCV) is defined for pattern ε according to the labels as:

$$L_\varepsilon = [r_1, r_2, \cdots, r_n] \tag{6}$$

where r_i is the occurrence ratio ε for class i, which is calculated by $\left|I_\varepsilon^i\right| / |I_\varepsilon|$. We define the majority label as the class label with the max pattern count of L_ε. Formally,

$$\text{majority label} = \arg\max_{y \in C}(L_\varepsilon[y]). \tag{7}$$

Assuming that a pattern only attributes to one class, it should belong to the majority label, so its occurrences in other classes should be viewed as noise. Because of the data capturing error and mixing with noise, it is possible that the patterns belonging to one class migrate to other patterns belonging to other adjacent classes. As a result, it may cause the emergence of other minor labels.

To purify the dataset, the instances with the pattern whose occurrence ratio is lower than a threshold could be removed. Therefore, the impurity ratio (IPR) is defined as the threshold to make instance purification more flexible. Given a pattern ε, the removal set (RS) is given as:

$$RS_{\varepsilon} = \bigcup_{c_i \in C} D_{S=\varepsilon, C=c_i}, \ r_i < IPR, \tag{8}$$

where C is the label vector.

For example, given a pattern set ε which is partitioned by the labels, we can count the amount of instances with the same pattern according to labels. Assuming the obtained PCV, $L_{\varepsilon} = [60/65, 0, 5/65]$, if the IPR is set to be 0.1, apparently, the five instances for the third class need to be removed. The detailed purifying process is shown in Algorithm 1.

Algorithm 1 Data Purifying Process

Input: A pattern ε, IPR μ, class set C;
Output: The removal set R;

function REMOVE(ε, C, μ)
  $R \leftarrow \emptyset$;
  Calculate L_{ε} $[r_1, r_2, \cdots, r_n]$ according to classes;
  for $i = \{1, \cdots, T\}$ **do**
    if $r_i < \mu$ **and** $r_i \neq 0$ **then**
      $R \leftarrow R \cup D_{S=\varepsilon, C=c_i}$
  return R

Actually, the proposed purification method can be adopted in most consistency based feature selection methods to realize instance purification and feature selection simultaneously. In this paper, LCC is applied and embedded into the proposed method because of its effectiveness and concision among consistency-based methods.

The overall procedure of the proposed method is presented in Algorithm 2. It takes the dataset and uses two preset thresholds, (namely, the thresholds of inconsistency σ and IPR μ) as inputs and outputs of the minimal feature subset S and purified instance set P. Firstly, due to its good robustness and stability property, the minimum Redundancy Maximum Relevance (mRMR) algorithm [32] is used to rank the features. Then let the minimal feature set S and purification instance set G as the universal set of feature and instance respectively. In the following, at each iteration, a feature is eliminated from S and checks if the inconsistency rate reaches the inconsistency threshold σ, whilst purifying the dataset at the $9th$ and $11th$ line. The algorithm terminates when the inconsistent rate reaches the threshold σ.

In addition, to exploit the monotonicity of consistency, the binary search is carried out to boost the speed of search [24]. The search complexity is $O(\log(N_f))$, where N_f is the number of features.

Figure 4 illustrates the relationship among the inconsistency rate, instance purification and feature selection. Through data purification the inconsistency will decrease, while as the feature selection progresses, the ICR will increase until the inconsistent rate reaches σ. Therefore, with the instance purification, the

Algorithm 2 FS&IP

Input: A dataset D can be described by features $F = \{f_1, \cdots, f_l\}$ and instances $I = \{i_1, \cdots, i_m\}$ respectively, label set C, inconsistency threshold σ and IPR μ;
Output: A minimal feature subset S,
A purifying instance set P;

1: **function** FS&IP(D, σ, μ)
2: Sort F in the incremental order of mRMR;
3: Let $S \leftarrow F$, $P \leftarrow I$;
4: **for** $i = \{1, \cdots, l\}$ **do**
5: $R \leftarrow \emptyset$, $IC \leftarrow 0$
6: S can be divided into $\{\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{k_i}\}$ according to the patterns;
7: **for** $j = 1, 2, \cdots, k_i$ **do**
8: $IC \leftarrow IC(\varepsilon_{k_i}) + IC$
9: $RS \leftarrow Remove(\varepsilon_{k_i}, C, \mu) \cup RS$
10: $ICR \leftarrow IC/|P|$
11: $P \leftarrow P \backslash RS$
12: **if** $ICR < \sigma$ **then**
13: $S \leftarrow S \backslash f_i$
14: **else**
15: break
return S and P

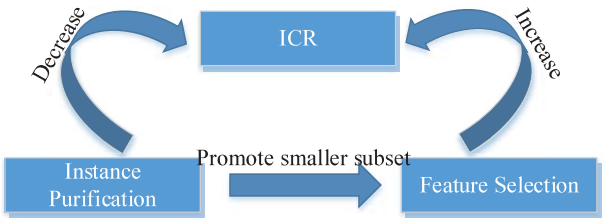


Figure 4: The relationship among feature selection, instance selection and ICR.

increasing trend will slow down, so that the proposed algorithm can obtain a smaller feature subset.

To give a clear illustration of the performance of the above process, feature selection is conducted by FS&IP on our dataset. Figure 5 shows the process of feature selection with three different thresholds of IPR. Because the linear search method (at each iteration, one feature would be eliminated from the feature set.) is adopted and the monotonicity of inconsistency, it is can be observed that the overall inconsistent rate is continuously increasing when $\mu = 0$ (this change also adapts to other datasets). However, when $\mu = 0.1$ *or* 0.2, the curves decline slightly in certain areas because of data purification, causing it to reach the threshold a bit late, and as a result, a smaller subset is obtained by proposed method than the original LCC algorithm.

5.2. Parameter optimization

There are two significant parameters in the proposed algorithm, which are the threshold of IPR and ICR, μ and σ. μ determines the amount of removed instances for a pattern subset and σ accounts for when the algorithm terminates.

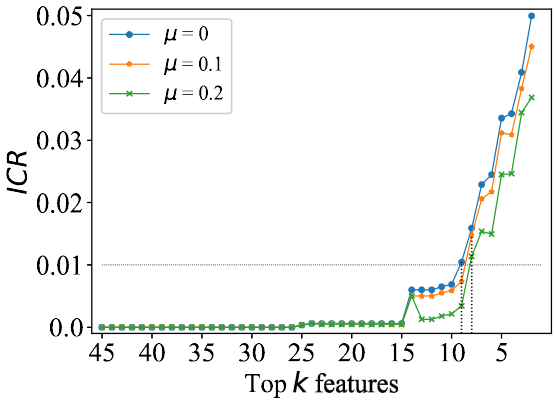


Figure 5: The relationship between *ICR* and feature subset under different *IPR*s.

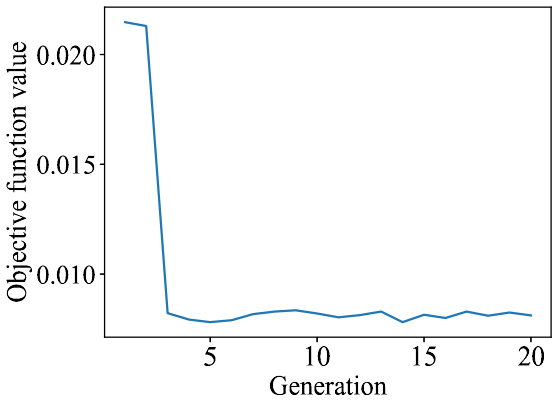


Figure 6: The parameters optimization for *ICR* and *IPR* by genetic algorithm (gotten by proposed traffic dataset).

To scale down the region of search, the value search ranges of two parameters are set in advance. Though *IPR* and *ICR* have the same value range [0,1], the values in predesigned search ranges need to be the most probable values for both thresholds. If *IPR* is too large, the algorithm will be disabled due to the fact that it would remove more instances even the instance with majority label. Therefore, *IPR* (impurity ratio) is selected from 0 to 0.3. If σ is too large, it would lead to no feature selected. In addition, with regard to the range value of *ICR*, we referred to literature [33, 34]. Consequently, *ICR* (inconsistent ratio) threshold is set from 0 to 0.1. However, there is still a problem that it is expensive to use the exhaustive search for the optimal solution. Therefore, the heuristic search method genetic algorithm is utilized [35] to expedite the process, which can use many mechanisms such as gene mutation, gene crossover to skip local optimum for getting global optimum.

Among dataset, there are a certain proportion of instances with noise which should be removed. On the other hand, instances collection requires some cost, especially for the medical or biology field; therefore, we hope algorithm can remove instances as few as possible but guarantee the accuracy. Meanwhile, it is desired that the output feature subset has less features. Consequently, the objective function is designed to search its minimum for the expectation to make a balance between accuracy and data reduction.

$$f = \rho \cdot (1 - A^t) + \frac{(1-\rho)}{2} \cdot (1 - \frac{S_i^t}{N_i} + \frac{S_f^t}{N_f}) \tag{9}$$

where A denotes the average accuracy of classifier with 10-fold cross validation, S_i means the amount of selected instances, and N_i represents the total amount of instances in dataset. S_f is the selected feature subset, N_f is the whole number of feature and t denotes the tth generation of genes. The first term on the right hand side of Eq. (9) accounts for the overall accuracy, and the second term for the data reduction. ρ is a weight coefficient ranging from 0 to 1 to adjust the important degree of two parts in the objective function. If ρ is more than 0.5, it suggests that the accuracy of classifier is more important, or else, data reduction takes more priority, where ρ is set to 0.8 in the paper.

In genetic algorithm, the maximum iteration number is set to 100 and the size of gene population is taken to 40, and the two parameters *ICR* and *IPR* are initialized with random values within the search ranges. In addition, to avoid wasting too much time after convergence, the additional stop condition where the average fitness value of last generation minus that of the present generation is less than 10^{-5} is added. The scores of objective function through multiple generations are plotted as Figure 6.

5.3. Complexity analysis

Given one dataset, the parameter optimization only needs to be carried out once. Thus, in addition to the process of the parameter optimization, the computing complexity of the proposed method mainly comprises two parts, i.e., the feature ranking and subset selection.

The first phase of the proposed method is feature ranking where different algorithms can be specified. Suppose the complexity of ranking algorithm is $O(R)$. The second is subset searching. When binary search is applied to boost the speed of search, the complexity degree of the search method is $O(\log_2 N_f)$, where N_f represents the number of features in the dataset. In addition, at each round of search, the *ICR* (inconsistent ratio) only needs to be calculated once. By employing a hashing mechanism where it actually would take certain time in constituting the hash table, the complexity degree of computing inconsistency is $O(N_i)$ [36], where N_i is the number of instances given the dataset. Besides, the *ICR* and *IPR* are almost calculated simultaneously, the calculation of *IPR* does not need additional time complexity. Therefore, the asymptotic time complexity of the searching algorithm is $O(N_i \log_2 N_f)$. Overall, the time complexity degree of proposed method is $O(max(N_i \log_2 N_f, R))$.

5.4. A novel feature-flow fragment

This subsection presents a set of discriminative features which has some unique properties and can significantly improve classification accuracy and performance. Firstly, we introduce the concept of flow in this paper. A flow is defined as a series of packets within a certain time which is serviced by a sequence of servers/routers along the path from the source to the destination in the network.

We only record the transport-layer session information of the packet header to avoid violating individual privacy. Hence, a packet is represented by a five-meta tuple:

$$x = \{time, srcIP, desIP, proto, packetsize\}, \tag{10}$$

where time is the timestamp to indicate when the packet is sent to the destination, $srcIP$ and $desIP$ are the source and destination IP addresses respectively, $proto$ is the transport layer protocol (e.g. TCP and UDP), and finally, $packetsize$ is the size of packet.

At the same time, we divide a datalink into the downstream link (DSL) and the upstream link (USL) by local IP address of the data capturing device. The DSL and USL are defined respectively as:

$$DSL = \{x_i | desIP(x_i) = localIP\}, \tag{11}$$

$$USL = \{x_i | srcIP(x_i) = localIP\}, \tag{12}$$

where x_i denote the ith packet arriving at NIC (network interface card).

Figure 7 shows the communication graph of DSL (downstream link) of one flow. Each session of downlink is represented by plotting a line. What we can see from communication graph is that each category indeed has its various behavioral representations, which are reflected by the linking condition, the protocol used for transferring and so forth. However, it is significantly complex, and this is the only downlink session. Therefore, it is necessary to extract sufficient discriminative features in order to identify these multimedia categories accurately.

From this complex communication behavior, we observe a group of unique linking behaviors for each category, called the flow fragment ($F-Frag$), which enables better distinction among different categories.

Definition 1. *A F-Frag in downstream link is defined as the continuous series of packets that have the same source IP address. Formally,*

$$\begin{aligned} &F-Frag = \{x_i | desIP(x_i) = localIP\} \\ &s.t. \quad i \text{ is consecutive} \end{aligned} \tag{13}$$

Our study shows that there are many such flow fragments in symmetric traffic. Therefore, the $F-Frag$ existing in DSL can be exploited to identify traffic generated by various applications.

To get the statistical characteristics, we count the number of $F-Frag$s, the average/entropy/variance of linking number of $F-Frag$s, and the total bytes of $F-Frag$s.

6. Experiments

This section describes the details related to the experiments, including the evaluation metrics, baseline methods, benchmark datasets and the experiment procedure.

6.1. Evaluation metrics

1. Overall accuracy (OA). Overall accuracy is the ratio of true samples to all the tested samples, which is expressed as:

$$OA = \frac{TP + TN}{TP + FP + TN + FN}, \tag{14}$$

where TP is the true positive instances; TN is the true negative instances; as such, FP and FN are the false positive and false negative instances, respectively. OA is used to assess the accuracy of feature subset in specific classifiers.

2. Subset size ratio (S_r). Besides the effectiveness of a feature subset, its size also needs to be considered. The subset size ratio is defined as:

$$S_r = \frac{S_f}{N_f}, \tag{15}$$

where S_f is the selected feature subset, and N_f is the whole feature set. A smaller value of S_r represents a smaller subset S.

3. Runtime of algorithms. To measure the speed of algorithm, the average runtime of each algorithm is calculated.

4. Data volume. To intuitively present the reduced volume of data, the data volume is defined as:

$$Data\ volume = \frac{S_f}{N_f} \cdot \frac{S_i}{N_i}, \tag{16}$$

where S_f is the amount of selected feature set, N_f is the amount of total feature set, S_i is the amount of purified instance set and N_i denotes the amount of the total instance set.

5. Feature stability (FS).

Definition 2. *The feature stability is defined to measure the quality of one feature as:*

$$FS(f_n) = \frac{1}{T} \cdot \sum_{i=1}^{T} \sum_{j=1}^{|S_i|} l(f_n = f_j), \tag{17}$$

where $FS(f_n)$ is the stability of feature f_n, and $l(t)$ is the indicator function; that is: $l(t)$ is 1, if the event t is true; it is 0, otherwise. T represents the number of participating algorithms and S_i is the selected subset of the ith algorithm. However, considering the ranks of features in one

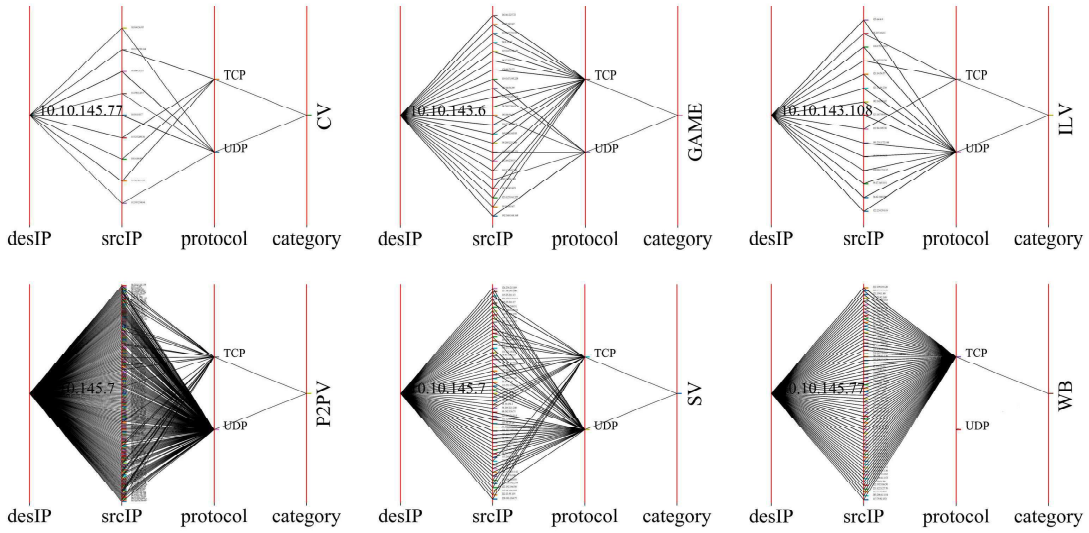


Figure 7: The communication graph of DSL of one flow for each category.

algorithm, the feature with higher rank should have the bigger weight. Therefore, we design the weighted item for each feature appearing in the subset. The final equation of FS is defined as:

$$FS(f_n) = \frac{1}{T} \cdot \sum_{i=1}^{T} \sum_{j=1}^{|S_i|} l(f_j = f_n) \cdot \frac{|S_i| - rank(f_n)}{|S_i| - 1}, \quad (18)$$

where $rank(\cdot)$ is the rank function which outputs the rank of features in descending order. The best feature in the subset multiplies by weight 1, while the feature with the lowest rank multiplies by $\frac{1}{|S_i|-1}$. A feature with lower rank is assigned with a lower weight. FS (feature stability) can not only represent the occurrence number of one feature, but also reflect the rank in one algorithm. A better feature has a higher FS score. Note that in the present study feature stability (FS) is defined as the robustness property or representative performance of a feature for some specific classification purposes, when evaluated using several or many different algorithms. FS employs the weighted frequency of each feature to evaluate feature performance on one dataset. It is known that different feature selection algorithms may generate different results, this is because different methods select features from different views. Therefore, given one feature, if it can achieve higher scores by different feature methods, it will then be treated as a good feature with stable discriminative ability.

6.2. The baseline methods and parameter setting

The proposed method is compared with the following six state-of-the-art algorithms: the improved correlated-based feature selection (ICFS) [37], FCBF [17], super-LCC (SLCC) [24], Steepest-Descent Consistency-Constrained (SDCC) [10], SCWC[23] and RFPSO [19] algorithms respectively; these methods directly output the optimal subset rather than the feature ranking. Among all seven methods, the consistency threshold for SLCC and SDCC is 0.01 (0.01 is the typical value from literature [24, 38]), which is a positive threshold obtained through multiple trials; the threshold value for the proposed method, however, is optimized by genetic algorithm. The number of particles in RFPSO is set at 50, and the number of iteration is 100; the fitness function is the same as in [39]. The threshold of FCBF and ICFS is set to 0.01. In this study, three classifiers, namely, decision tree (J48 version) (DT), SVM with RBF kernel (SVM-rbf) and Naive Bayes(NB), are used to verify the generalization and robustness performance of the selected feature subsets. Moreover, a grid search is implemented to search a better parameter combination of SVM through cross-validation since it is more sensitive to the hyper-parameter settings, where we evaluated the regularization parameter $C = [1, 10, 100, 1000]$ and kernel efficient $\gamma = [0.0001, 0.001]$.

6.3. UCI Datasets

In addition to our multimedia dataset, five UCI datasets are also used to evaluate the performance of each algorithm. Table 4 lists the size of instances and features, the classified categories, and the ratio of the number of instances to the number of features (In/F). From Table 4, we can see that these datasets incorporate both binary and multivariate data, and they are ranked by the ratio of In/F in order to compare the performances of algorithms in multi-class and binary classification tasks and also to investigate the effect of In/F on algorithms.

6.4. Experiment setup

An overview of experiment is shown in Figure 8, where the blocks with slashes represent the algorithm modules; particularly, the red one employs our proposed method. The data is divided into training and test sets for 10-fold cross

Table 4: Description of our multimedia dataset.

Dataset	#Instance	#Feature	Category	In/F	Reference
Har	7352	561	6	13.10	[40]
Traffic	360	40	6	9.00	[1]
Madelon	4000	500	2	8.00	[41]
Hiva	3845	1617	2	2.37	[42]
Gisette	6000	5000	2	1.20	[41]
Gene	801	20531	5	0.03	[43]
Arcene	100	10000	2	0.01	[41]

Figure 8: Experiment system overview.

validation (CV). In each CV, one fold is used as test set and remaining folds are employed as training set. The discretization algorithms are employed in training set to generate the discretized cut-points; then, the entire dataset is input into the feature selection algorithm to reconstruct dataset based on features and instances. The reconstructed dataset is used to train classifier and get the test score. Though discretization may lead to information loss, it can also remove some redundant information. As long as using discretized intervals to achieve adequate number, the accuracy of classifiers will be probably higher than that in the continuous space. We set the discretized intervals at the best value 10 through multiple tests.

To make the fair comparison, all experiments were conducted with Python on a computer running Windows 10, with the configuration of an Intel® Core™ 3 GHZ CPU and 16G of RAM. In addition, some data analysis in the early stage was done on WEKA[1], including the analysis of data distribution and data visualization.

7. Results and Discussions

7.1. Results on noisy multimedia traffic dataset

Because of the difficulty of visualizing the dataset with multiple features, we carry out feature reduction by principal components analysis (PCA) to observe the results, which consists of two principal components. The compared scatter plots of the proposed dataset are shown in Figure 9, where the horizon and vertical axes of two subgraphs are the first and second principal components respectively.

As shown in Figure 9, it is noteworthy that the closer it is to the overlapping area, the more points are removed. Moreover, these points which deviate from the area of its own self-category and enter other categories are removed by the algorithm. It can help improve the performance of classification.

[1]https://www.cs.waikato.ac.nz/ml/weka/

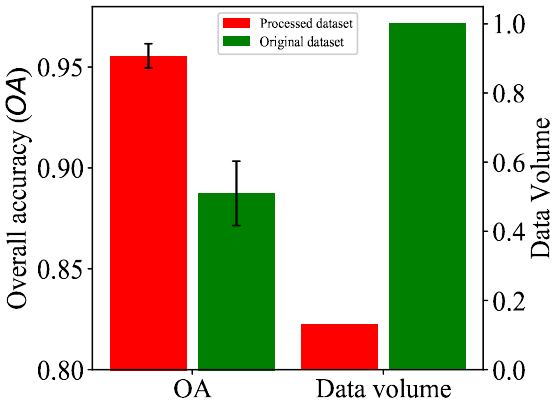


Figure 10: Comparison between the processed dataset and the original one, where the average *OA* is obtained by three mentioned classifiers.

7.2. Accuracy comparisons

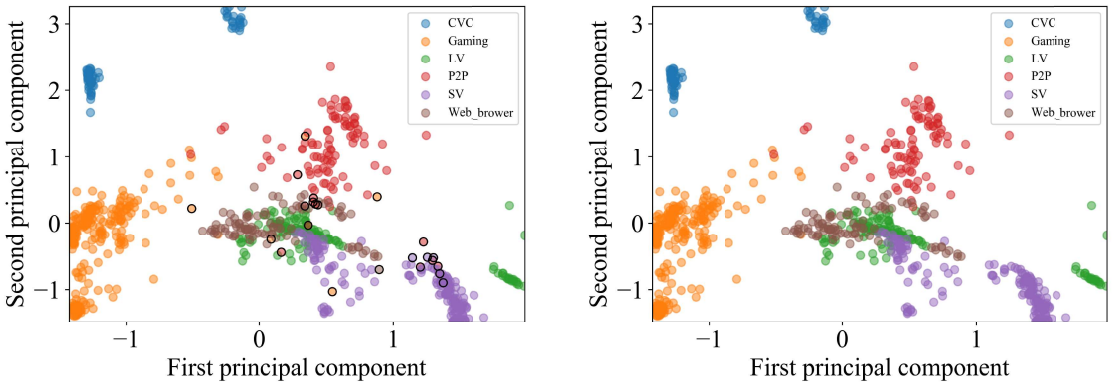
Figure 10 shows the comparison between the processed dataset and the original one. It can be observed that the *OA* (overall accuracy) is improved by about 6% and the fluctuation becomes smaller, suggesting that the proposed method is able to select a good feature subset, and reduce the performance variation of classifier. In terms of data volume, the proposed method is capable of reducing the volume by about 77% over the original dataset for both features and the number of instances. Large reduction of data volume can reduce the computational memory and increase effectiveness of back-end classifier. Therefore, it can be validated that proposed method is largely conducive to multimedia traffic classification.

Figures 11 and 12 shows the *OA* of DT (decision tree), SVM and NB (Naive Bayes) to reflect the accuracy of selected feature subset, and S_r score which measures the size of the selected feature subset. The proposed method FS&IP has a significantly better performance than ICFS, FCBF and SDCC. Compared with SLCC, our method behaves better when using the DT and NB classifiers; at the same time, it selects a smaller size of feature subset. It is because that our method employs data selection to make traffic data purer. The results indicate that our method is able to generate a better feature and instance subset to improve classification performance. In comparison with SCWC, our method achieves more accurate classification though it produces feature subset with bigger size. On accuracy, RFPSO has similar performance with our method. However, in terms of the dimension of feature, our method only uses about one seventh of the features selected by RFPSO to achieve the close accuracy.

The above comparisons demonstrate that the proposed method can find a set of more discriminative representatives (features and instances) for multimedia traffic classification.

7.3. Feature stability

Ten algorithms are compared to evaluate features stability, including Relieff [26], Chi2 [44], CMIM [14], six baseline methods, and the proposed method. Because mRMR, Chi2 and



(a) The original dataset (the points with black circles are the removed instances)

(b) The processed dataset by the proposed method

Figure 9: The scatter points of PCA over multimedia traffic dataset.

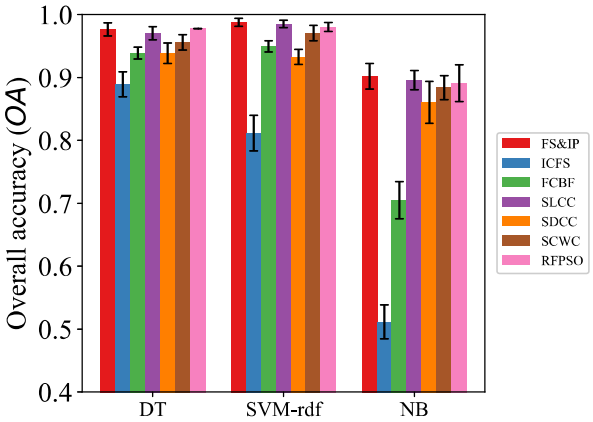


Figure 11: Performance comparison for seven algorithms (comprising the *OA* of DT, SVM, NB).

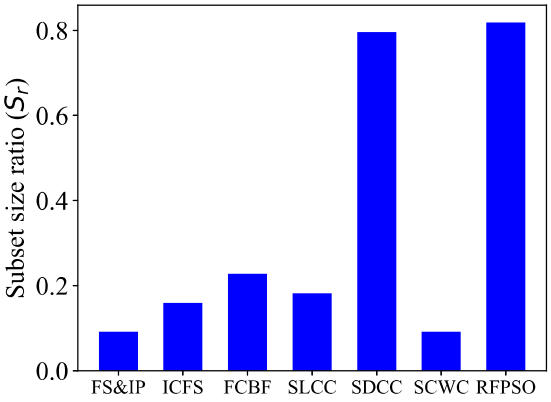


Figure 12: The comparison of feature subset size ratio for seven algorithms.

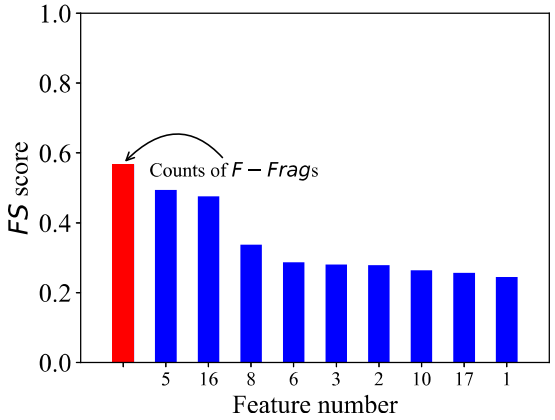


Figure 13: *FS* score for top 10 features.

CMIM only give the rank of features, we select the subset of top 10 features in the rank for evaluation. As shown in Figure 13, the *FS* score for each feature is calculated, and 10 features with top *FS* scores are presented.

In Figure 13, the abscissa axis shows the serial number of features which are presented in Appendix in the descending order of *FS* score. In particular, the red bar represents the performance of the counts of $F-Frag$s. It is observed that the proposed feature (the counts of flow fragments) behaves best in terms of *FS*, about 7% higher than the second best feature. This suggests that the probability of $F-Frag$ based features to be selected is higher and they rank high among these features. Different traffic types have different network link behaviors, while the set of feature based on $F-Frag$s is calculated on the basis of link behaviors, that is the reason behind the success of the proposed features.

7.4. Comparison on UCI datasets

In this subsection, we make comparison over seven public datasets listed in Table 5. Because SDCC uses the exhausted

Table 5: The performance comparison on public datasets.

Dataset	Method	Best_dis	Best_con	Mean_dis (Std)	Mean_con (Std)	S_r	Time (Sec.)
Har In/F: 13.10	Proposed	0.800	0.896	0.776 (0.033)	0.871 (0.012)	0.023	642.08
	ICFS	0.549	0.682	0.519 (0.042)	0.639 (0.054)	0.021	1142.276
	FCBF	0.499	0.628	0.496 (0.003)	0.589 (0.038)	**0.018**	**434.741**
	SLCC	0.753	0.826	0.706 (0.034)	0.756 (0.102)	0.023	624.541
	SCWC	0.922	0.951	**0.863 (0.064)**	0.901 (0.065)	0.082	625.796
	PSO	**0.962**	**0.975**	0.835 (0.153)	**0.909 (0.075)**	0.631	1947.541
	Whole	0.962	0.979	0.826 (0.167)	0.906 (0.073)	1.000	
Traffic In/F: 9.00	Proposed	**0.949**	0.945	**0.940 (0.008)**	**0.938 (0.010)**	0.250	8.54
	ICFS	0.714	0.821	0.706 (0.008)	0.720 (0.082)	**0.150**	**6.12**
	FCBF	0.926	0.938	0.915 (0.010)	0.913 (0.031)	0.250	7.51
	SLCC	0.873	0.936	0.853 (0.014)	0.855 (0.079)	0.325	8.96
	SCWC	0.945	**0.950**	0.931 (0.011)	0.908 (0.056)	0.550	8.04
	PSO	0.944	**0.950**	0.938 (0.008)	0.913 (0.045)	0.550	9.41
	Whole	0.938	0.942	0.867 (0.100)	0.907 (0.045)	1	
Madelon In/F: 8.00	Proposed	**0.826**	0.790	**0.751 (0.052)**	**0.748 (0.045)**	0.042	43.983
	ICFS	0.762	0.738	0.675 (0.070)	0.680 (0.047)	**0.012**	113.361
	FCBF	0.785	0.789	0.698 (0.051)	0.716 (0.049)	0.020	**14.551**
	SLCC	0.782	**0.807**	0.749 (0.062)	0.738 (0.051)	0.046	42.115
	SCWC	0.792	0.804	0.717 (0.087)	0.730 (0.051)	0.054	39.260
	PSO	0.736	0.724	0.689 (0.061)	0.635 (0.082)	0.624	248.994
	Whole	0.763	0.742	0.642 (0.065)	0.602 (0.012)	1.000	
Hiva In/F: 2.37	Proposed	**0.965**	—*	**0.965 (0.0003)**	—	0.021	580.64
	ICFS	0.945	—	0.924 (0.012)	—	0.032	1085.24
	FCBF	0.958	—	0.947 (0.000)	—	**0.006**	**45.65**
	SLCC	0.964	—	0.943 (0.016)	—	0.026	599.85
	SCWC	0.961	—	0.951 (0.001)	—	0.023	641.90
	PSO	0.964	—	0.959 (0.010)	—	0.625	526.70
	Whole	0.969	—	0.960 (0.010)	—	1	
Gisette In/F: 1.20	Proposed	**0.973**	0.952	**0.949 (0.041)**	**0.920 (0.012)**	0.004	1233.443
	ICFS	0.970	0.967	0.882 (0.084)	0.891 (0.082)	0.020	3699.329
	FCBF	0.856	0.844	0.834 (0.021)	0.795 (0.062)	**0.002**	**457.382**
	SLCC	0.957	0.926	0.935 (0.031)	0.908 (0.010)	0.004	909.943
	SCWC	0.948	0.972	0.924 (0.017)	0.903 (0.056)	0.031	939.656
	PSO	**0.973**	**0.973**	0.946 (0.019)	0.903 (0.056)	0.615	4450.637
	Whole	0.964	0.971	0.941 (0.023)	0.916 (0.061)	1.000	
Gene In/F: 0.03	Proposed	0.960	0.960	0.947 (0.011)	0.942 (0.002)	0.001	77.453
	ICFS	—**	—	—	—	—	—
	FCBF	0.755	0.748	0.698 (0.072)	0.732 (0.001)	**0.0005**	1085.611
	SLCC	0.958	0.955	0.944 (0.012)	0.944 (0.003)	0.001	**50.012**
	SCWC	0.974	**0.978**	0.964 (0.008)	**0.971 (0.001)**	0.001	51.974
	PSO	**0.976**	0.966	**0.965 (0.017)**	0.901 (0.032)	0.607	1873.561
	Whole	0.978	0.971	0.925 (0.043)	0.910 (0.023)	1.000	
Arcene In/F: 0.01	Proposed	**0.790**	**0.780**	**0.747 (0.033)**	**0.733(0.017)**	**0.001**	0.912
	ICFS	0.560	0.580	0.533 (0.019)	0.550 (0.036)	0.002	0.927
	FCBF	0.580	0.610	0.557 (0.001)	0.590 (0.014)	0.002	46.228
	SLCC	0.742	0.700	0.727 (0.025)	0.610 (0.021)	0.003	**0.559**
	SCWC	0.780	0.760	0.740 (0.036)	0.717 (0.031)	0.004	0.667
	PSO	0.761	0.730	0.653 (0.054)	0.670 (0.085)	0.611	209.889
	Whole	0.800	0.742	0.690 (0.086)	0.720 (0.012)	1.000	

* Hiva is a discretized dataset originally; thus it have no the results in the columns Best_con and Mean_con.
** ICFS did not complete the job probably because of lack of memory or takes much time.

search to select features, it will spend much time with large-scale dataset, which cannot meet the demand of effectiveness in TC, so SDCC is not considered in this comparison. Table 5 shows the overall performance of six feature selection algorithms using three classifiers on seven datasets. Among them, the 'Best_dis' and 'Best_con' columns are the best *OA* of three classifiers (including SVM, DT, NB) for each algorithm in discretized and continuous datasets respectively. The 'Mean_dis' and 'Mean_con' columns present the average *OA* of three classifiers in the discretized and continuous datasets. For each dataset, the 'Whole' represents that the classification is conducted with its all features. The values highlighted in bold are the top scores in each dataset with regard to the certain criteria. Since the ICFS algorithm

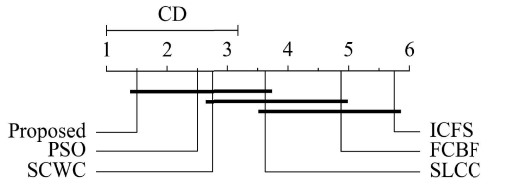
involves very high time overhead, it fails to finish running within 2h on Gene dataset.

From Table 5, we can observe that the proposed method outperforms other algorithms on three datasets in terms of accuracy, in the case of selecting smaller feature subset. Compared with SLCC, our method always obtains better results with smaller or equal size subset, which is due to the instance purification. RFPSO also gets the high accuracy score, but its feature subset as input of classifier is 30 times bigger than that of proposed method on average. Besides, it needs much runtime to select features. SCWC also works best on two datasets, which has the similar performance to the proposed method, but it tends to select bigger subset than the proposed method.

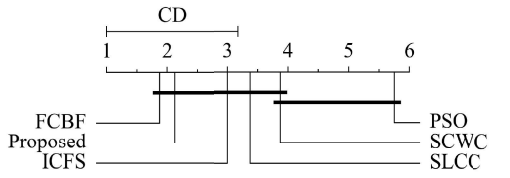
In terms of the dimension of selected features, FCBF and ICFS tend to select a relatively smaller number of features, but cannot obtain the accurate and stable results. RFPSO is able to achieve the high good accuracy, but it is inclined to select the feature subset with bigger size, and require long running time. Among the consistency-based methods, our method is able to select the smallest subset and still get similar or better results.

On runtime, RFPSO seems to need more time to complete classification task, and because of each run of heuristic search, its performance is not stable. ICFS is at the second place and it tends to consume a lot of memory. The consistency-based methods generally take the shorter runtime. Specially, for the dataset with the small size of instances and large-scale features like Gene and Arcene, they show the significant advantage over other methods. Therefore, the methods based on consistency are particularly suitable for these datasets with smaller In/F. Among three consistency based methods, the proposed method consumes more time than others, because it needs additional time to check and remove the noise in each iteration, while FCBF seems to take shorter time on the dataset with many instances. Regarding discretization, it is interesting that it indeed redounds to improve the performance of some classifiers on some datasets. But the degree of improvement depends on both classifiers and datasets.

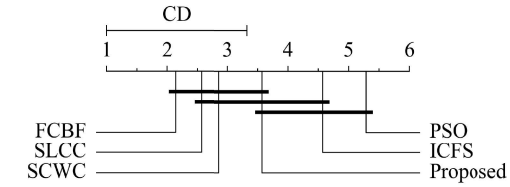
Now we look back to the three major problems as stated in Section 3. With regard to the first two problems, although some works have been done previously, such as the FCBF [17], SLCC [24] and SCWC [23], our work can solve these problems more effectively and accurately, as shown in Figure 11 and Table 5. This is mainly because our work is based on the consistency measure, while considering data purification at the same time. The consistency measure can spontaneously discover the feature interaction and reduce the redundant features. While the data purification enables the feature selection algorithm to choose a more compact feature set, as illustrated in Figure 5. As for the third problem, although the denoising approaches have been adopted by previous works [27, 45], our work can solve it more effectively. Because there is no need to pre-learn these mislabeled instances, instead, it utilizes the natural characteristics of the dataset, consistency measure, to purify data; specifically, it removes the instances with lower *IPR* (impurity ratio) in *PCV* (pattern count vector). More importantly, FS&IP can solve these problems concurrently, and the processes of feature selection and data purification are mutually advancing without the learning stage. Therefore, the proposed method can better solve the stated three problems and achieve better overall performance in comparison with the compared methods.



(a) The critical difference diagram of OA (overall accuracy).



(b) The critical difference diagram of S_r (subset size ratio)



(c) The critical difference diagram of time.

Figure 14: Comparison of all six methods with Nemenyi posthoc test; groups of methods that are not significantly different (at $\alpha = 0.1$) are connected with bold lines.

7.5. Statistical analysis

To determine if an algorithm significantly more sensitive to the heterogeneity of different datasets, two statistical hypothesis tests are performed: Friedman test followed by Nemenyi post-hoc test [5]. Friedman test compares the results of the proposed method and other methods on different datasets to determine whether there are statistically significant differences between them by ranking the results given by each method on different datasets. Particularly, the best method is ranked at the first place, the second best one is ranked at the second place, and so on. If null-hypothesis is rejected, we proceed to Nemenyi posthoc test in a pairwise manner. The performances of two methods are significantly different if their corresponding average ranks differ by at least the critical difference $CD = q_\alpha \sqrt{\frac{k(k+1)}{6G}}$, where k is the number of methods, G is the amount of datasets ($G = 8$ in the experiment), and q is based on the studentized range statistic divided by $\sqrt{2}$. When comparing all the algorithms against each other, the results of the post-hoc tests are visually represented with a simple diagram in Figure 14, where the groups of the algorithms, which are not significantly different, are connected.

The analysis reveals that FS&IP, FCBF and PSO are more likely to perform significantly better than ICFS and FCBF in

terms of OA. For S_r (subset size ratio), the results confirms that FCBF, ICFS and FS&IP have the better performance than that of PSO. With regard to running time, PSO has the significant worst performance and the group of methods FS&IP, FCBF, SCWC, SLCC has insignificant differences. Overall, the proposed method performs well when in terms of the accuracy, subset size and running time.

8. Conclusions and future work

How to efficiently select features to solve the aforementioned problems of interest, while cleaning up noisy data becomes a pivotal point in TC (traffic classification). This paper proposes a novel technique to address the above problems simultaneously, which not only makes use of the measure of the inconsistency rate (*ICR* defined in Eq. (5)) for selecting the most relevant features in each iteration, but also employs a newly proposed measure, called the impurity ratio (*IPR*), for removing noisy instances during each iteration. In addition, the experimental results suggest that the proposed method outperforms other methods on most datasets in terms of accuracy, running time and S_r score. Following that, a type of more discriminative features termed as $F - Frag$ based on the network connection behaviors is proposed and an evaluation criteria on the goodness of features FS is defined. The empirical results suggest that the proposed features are more valid than other features in terms of FS. The source of this work is available on Github at the url: https://github.com/wuzheng1994/FS-IP.git.

Still, this work can be extended in several ways. First, in FS&IP it uses the mRMR serves for feature ranking, and the employment of other feature ranking algorithms may lead FS&IP to produce different performances, and some algorithms may perform better than mRMR. In future work, other feature selection methods will be considered and integrated to the proposed framework if they work better. Besides, the consistency-based methods are proved efficient consisting of a large number of variables or features. While for datasets with large-scale instances, to reduce the runtime for feature selection will be one of the focuses in our future work. Finally, in this paper, the performance of seven algorithms are compared over six datasets. More extensive experiments can be carried out in order to statistically analyze the performance of different algorithms.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (No.61271233).

References

[1] Y. Dong, J. Zhao, J. Jin, Novel feature selection and classification of Internet video traffic based on a hierarchical scheme, Computer Networks 119 (2017) 102–111.

[2] T. T. Nguyen, G. J. Armitage, A survey of techniques for Internet traffic classification using machine learning, IEEE Communications Surveys and Tutorials 10 (1-4) (2008) 56–76.

[3] Cisco Visual Networking Index: Forecast and Trends, Website, `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html?dtid=osscdc000283` (2019).

[4] C. Xu, S. Chen, J. Su, S. M. Yiu, et al., A Survey on Regular Expression Matching for Deep Packet Inspection: Applications, Algorithms, and Hardware Platforms, IEEE Communications Surveys & Tutorials 18 (4) (2016) 1–1.

[5] J. Zhang, X. Chen, Y. Xiang, W. Zhou, et al., Robust Network Traffic Classification, IEEE/ACM Transactions on Networking 23 (4) (2015) 1257–1270.

[6] H. Shi, H. Li, Z. Dan, C. Cheng, W. Wei, Efficient and robust feature extraction and selection for traffic classification, Computer Networks 119 (C) (2017) 1–16.

[7] Z. Liu, R. Wang, N. Japkowicz, Y. Cai, et al., Mobile app traffic flow feature extraction and selection for improving classification robustness, Journal of Network and Computer Applications 125 (2019) 190–208.

[8] B. Xue, M. Zhang, W. N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, IEEE Transactions on Evolutionary Computation 20 (4) (2015) 606–626.

[9] A. Senawi, H.-L. Wei, S. A. Billings, A new maximum relevance-minimum multicollinearity (MRmMC) method for feature selection and ranking, Pattern Recognition 67 (2017) 47–61.

[10] A. P. Angulo, K. Shin, Fast and accurate steepest-descent consistency-constrained algorithms for feature selection, in: International Workshop on Machine Learning, Optimization and Big Data, Springer, 2015, pp. 293–305.

[11] D. Junhua, L. Xinchuan, K. Xiaojun, K. Xiaojun, A Case Study of the Augmentation and Evaluation of Training Data for Deep Learning, Journal of Data and Information Quality 11 (2019) 20–41.

[12] B. Tang, S. Kay, H. He, Toward optimal feature selection in naive bayes for text categorization, IEEE Transactions on Knowledge and Data Engineering 28 (9) (2016) 2508–2521.

[13] M. A. Ambusaidi, X. He, P. Nanda, Z. Tan, Building an intrusion detection system using a filter-based feature selection algorithm, IEEE Transactions on Computers 65 (10) (2016) 2986–2998.

[14] S. Jiang, K. S. Chin, G. Qu, K. L. Tsui, An Integrated Machine Learning Framework for Hospital Readmission Prediction, Knowledge-Based Systems 146 (2018) S0950705118300443.

[15] H. Shi, H. Li, D. Zhang, C. Cheng, X. Cao, An efficient feature generation approach based on deep learning and feature selection techniques for traffic classification, Computer Networks 132 (2018) 81–98.

[16] M. Liu, D. Zhang, Pairwise constraint-guided sparse learning for feature selection, IEEE Transactions on Cybernetics 46 (1) (2015) 298–310.

[17] B. Senliol, G. Gulgezen, L. Yu, Z. Cataltepe, Fast correlation based filter (fcbf) with a different search strategy, in: 2008 23rd International Symposium on Computer and Information Sciences, IEEE, 2008, pp. 1–4.

[18] A. Fahad, Z. Tari, I. Khalil, I. Habib, H. Alnuweiri, Toward an efficient and scalable feature selection approach for internet traffic classification, Computer Networks 57 (9) (2013) 2040–2057.

[19] Y. Dong, Q. Yue, M. Feng, An efficient feature selection method for network video traffic classification, in: 2017 IEEE 17th International Conference on Communication Technology (ICCT), IEEE, 2017, pp. 1608–1612.

[20] A. Onan, A fuzzy-rough nearest neighbor classifier combined with consistency-based subset evaluation and instance selection for automated diagnosis of breast cancer, Expert Systems with Applications 42 (20) (2015) 6844–6852.

[21] Z. Zhao, H. Liu, Searching for interacting features in subset selection, Intelligent Data Analysis 13 (2) (2009) 207–228.

[22] K. Shin, M. X. Xian, Consistency-Based Feature Selection, in: International Conference on Knowledge-based & Intelligent Information & Engineering Systems, 2009.

[23] K. Shin, S. Miyazaki, A Fast and Accurate Feature Selection Algorithm Based on Binary Consistency Measure, Computational Intelligence 32 (4) (2016) 646–667.

[24] K. Shin, T. Kuboyama, T. Hashimot, D. Shepard, Super-cwc and

[1] https://github.com/

super-lcc: Super fast feature selection algorithms, in: 2015 IEEE International Conference on Big Data (Big Data), IEEE, 2015, pp. 1–7.
[25] D. L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man, and Cybernetics (3) (1972) 408–421.
[26] J. Zhang, M. Chen, S. Zhao, S. Hu, Z. Shi, Y. Cao, ReliefF-based EEG sensor selection methods for emotion recognition, Sensors 16 (10) (2016) 1558.
[27] B. Anderson, D. McGrew, Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 1723–1732.
[28] H. Schwarzenbach, A. M. da Silva, G. Calin, K. Pantel, Data normalization strategies for microrna quantification, Clinical chemistry 61 (11) (2015) 1333–1342.
[29] Z. Wu, Y. Dong, L. Yang, P. Tang, A new structure for internet video traffic classification using machine learning, in: 2018 Sixth International Conference on Advanced Cloud and Big Data (CBD), IEEE, 2018, pp. 322–327.
[30] J. Zhang, C. Chen, Y. Xiang, W. Zhou, Y. Xiang, Internet traffic classification by aggregating correlated naive bayes predictions, IEEE Transactions on Information Forensics and Security 8 (1) (2012) 5–15.
[31] S. Garcia, J. Luengo, J. A. Sáez, V. Lopez, F. Herrera, A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning, IEEE Transactions on Knowledge and Data Engineering 25 (4) (2012) 734–750.
[32] O. A. Alomari, A. T. Khader, M. A. Al-Betar, M. A. Awadallah, A novel gene selection method using modified MRMR and hybrid bat-inspired algorithm with -hill climbing, Applied Intelligence 48 (5439) (2018) 1–19.
[33] A. P. Angulo, K. Shin, Fast and accurate steepest-descent consistency-constrained algorithms for feature selection, in: International Workshop on Machine Learning, Optimization and Big Data, Springer, 2015, pp. 293–305.
[34] K. Shin, D. Fernandes, S. Miyazaki, Consistency measures for feature selection: a formal definition, relative sensitivity comparison and a fast algorithm, in: Twenty-Second International Joint Conference on Artificial Intelligence, 2011.
[35] M. Zhao, C. Fu, L. Ji, K. Tang, M. Zhou, Feature selection and parameter optimization for support vector machines: A new approach based on genetic algorithm with feature chromosomes, Expert Systems with Applications 38 (5) (2011) 5197–5204.
[36] M. Dash, H. Liu, Consistency-based search in feature selection, Artificial Intelligence 151 (1-2) (2003) 155–176.
[37] M. Mursalin, Z. Yuan, Y. Chen, N. V. Chawla, Automated Epileptic Seizure Detection Using Improved Correlation-based Feature Selection with Random Forest Classifier, Neurocomputing 241 (C) (2017) 204–214.
[38] K. Shin, D. Fernandes, S. Miyazaki, Consistency measures for feature selection: a formal definition, relative sensitivity comparison and a fast algorithm, in: Twenty-Second International Joint Conference on Artificial Intelligence, 2011.
[39] S. M. Vieira, L. F. Mendona, G. J. Farinha, J. M. C. Sousa, Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients, Applied Soft Computing 13 (8) (2013) 3494–3504.
[40] D. Anguita, A. Ghio, L. Oneto, X. Parra, J. L. Reyes-Ortiz, Energy Efficient Smartphone-Based Activity Recognition using Fixed-Point Arithmetic, J. UCS 19 (9) (2013) 1295–1314.
[41] I. Guyon, S. Gunn, M. Nikravesh, L. A. Zadeh, Feature extraction: foundations and applications, Vol. 207, Springer, 2008.
[42] I. Guyon, A. R. S. A. Alamdari, G. Dror, J. M. Buhmann, Performance prediction challenge, in: The 2006 IEEE International Joint Conference on Neural Network Proceedings, IEEE, 2006, pp. 1649–1656.
[43] J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, et al., The cancer genome atlas pan-cancer analysis project, Nature Genetics 45 (10) (2013) 1113.
[44] H. Liu, R. Setiono, Feature selection via discretization, IEEE Transactions on knowledge and Data Engineering 9 (4) (1997) 642–645.
[45] F. Gargiulo, C. Sansone, Improving performance of network traffic classification systems by cleaning training data, in: 2010 20th International Conference on Pattern Recognition, IEEE, 2010, pp. 2768–2771.

Appendix. Collection of statistical features

Table A1 lists 44 statistical flow-level features used in this paper with their serial number used in Section 7.3.

Table A1: Collection of features used in this paper.

No.	Feature
1	Average downlink packetsize
2	Average uplink packetsize
3	Maximum downlink packetsize
4	Maximum uplink packetsize
5	Valid protocol ratio of downlink
6	Average uplink rate
7	Average downlink rate
8	Variance of uplink packetsize
9	Variance of downlink packetsize
10	Maximum downlink arrival interval
11	Maximum uplink arrival interval
12	Average downlink arrival interval
13	Average uplink forward interval
14	Variance of downlink arrival interval
15	Variance of uplink forward interval
16	Information entropy of downlink packetsize
17	Valid IP ratio of datalink
18	Rate ratio of downlink to uplink
19	Packet counts ratio of downlink to uplink
20	Bytes ratio of downlink to uplink
21	Entropy of downlink interval
22	Average datalink packetsize
23	Average datalink interval
24	Variance of datalink packetsize
25	Maximum datalink packetsize
26	Variance of datalink interval
27	Maximun datalink interval
28	Datalink flow fragment counts
29	Average links of datalink flow fragments
30	Variance of links of datalink flow fragments
31	Information entropy of datalink flow fragments
32	Average bytes of datalink flow fragments
33	Coefficient variance of downlink arrival rate
34	Variance of downlink packetsize
35	Coefficient variation of downlink arrival interval
36	Coefficient variation of uplink packetsize
37	Variance of uplink packetsize
38	Coefficient variation of uplink forward interval
39	Variance of uplink interval
40	Coefficient variation of datalink packetsize
41	Coefficient variation of datalink interval
42	Coefficient variation of datalink rate
43	Average datalink rate
44	Variance of datalink rate