SVM-based loss differentiation algorithm for wired-cum-wireless networks

DENG Qian-hua (✉), CAI An-ni

School of Telecommunication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract

In a hybrid wired-cum-wireless network environment, packet loss may happen because of congestion or wireless link errors. Therefore, differentiating the cause is important for helping transport protocols take actions to control congestion only when the loss is caused by congestion. In this article, an end-to-end loss differentiation mechanism is proposed to improve the transmission performance of transmission control protocol (TCP)-friendly rate control (TFRC) protocol. Its key design is the introduction of the outstanding machine learning algorithm – the support vector machine (SVM) into the network domain to perform multi-metric joint loss differentiation. The SVM is characterized by using end-to-end indicators for input, such as the relative one-way trip time and the inter-arrival time of packets fore-and-aft the loss, while requiring no support from intermediate network apparatus. Simulations are carried out to evaluate the loss differentiation algorithm with various network configurations, such as with different competing flows, wireless loss rate and queue size. The results show that the proposed classifier is effective under most scenarios, and that its performance is superior to the ZigZag, mBiaz and spike (ZBS) scheme.

Keywords machine learning, SVM, TCP-friendly rate control (TFRC), congestion control

1 Introduction

In recent years, as wireless links have become more common on the Internet, transport protocol behavior in hybrid networks becomes important. Research on improving the performance of TCP and other TCP-friendly transport protocols in wired-cum-wireless networks has been focused on distinguishing losses caused by congestion from losses caused by wireless channel errors. Loss differentiation on TCP can usually be done based on TCP state variables [1], that is, congestion window (CWND), slow start threshold (Ssthresh), and acknowledgement (ACK), which, however, do not often exist in a best-effort transport protocol, such as TFRC [2]. In this article, an end-to-end classifier is proposed for differentiating the cause of packet loss for TFRC flows in networks with either last-hop or backbone wireless links. The classifier is based on a trained learning machine named SVM with multiple features. Its feature selection is mainly inspired by the following observations:

congestion–induced loss often occurs around a spike [3] of the relative one-way trip time (ROTT) and lasts for a period of time, and the inter-arrival times (IAT) of packets after two classes of the loss have different characteristics.

In this article, a packet loss event caused by wireless link error is denoted as a wireless loss, and a packet loss event caused by congestion is denoted as a congestion loss. The remainder of this article is organized as follows. Related works are described in Sect. 2, and the novel loss differentiation algorithm is proposed in Sect. 3. The simulations and evaluations are carried out in Sect. 4. Finally, conclusions are drawn in Sect. 5.

2 Related works

User datagram protocol (UDP) has been widely used in video streaming applications on account of its low delay. However, UDP lacks a mechanism for congestion control; therefore, it poses heavy threat to network stability. TFRC is an equation-based congestion control algorithm for UDP. It estimates the recent loss event rate of a video stream at the receiver, and informs this loss event rate back to the sender.

Received date: 02-06-2008
Corresponding author: DENG Qian-hua, E-mail: deng.qianhua@gmail.com
DOI: 10.1016/S1005-8885(08)60256-3
To behave in a TCP-friendly manner, the sender adapts its rate according to an equation that models the TCP’s response in a steady state, but does so with significantly less fluctuation in the sending rate than the standard TCP congestion control algorithm. The core throughput equation is:

\[ X = \frac{2ap}{3} R \left( \frac{3ap}{8} (p(1 + 32p^3)) \right) \]  
\[ \text{where } X \text{ is the transmission rate, } s \text{ is the packet size, } R \text{ is the round trip time (RTT), } p \text{ is the loss event rate, } t_{RTO} \text{ is the TCP re-transmission timeout value, and } a \text{ indicates the number of packets acknowledged by a single TCP acknowledgment.} \]

From Eq. (1), it can be seen that if the wireless loss is misclassified as congestion loss, the throughput \( X \) will unnecessarily decrease with \( p \). Therefore, loss differentiation is necessary for efficient link utilization in wired-cum-wireless networks. Research has been carried out to discriminate congestion loss and wireless loss to improve the performance of transport protocols. The known approaches can be mainly divided into two categories. One relies on the support of network inter-apparatus to perform the differentiation, and typical examples are the splitting connection protocols [4] and the explicit congestion notification (ECN) based scheme [5]. The other is called the end-to-end approach, which has the advantage of requiring no support from the network, and can be implemented by endowing one of the terminal devices with a classification function. Two kinds of end-to-end approaches have been suggested, one based on network measurements and the other based on model-based learning organon. Methods of the former type rely on information such as RTTs, ROTTs, and IATs measured on end systems to establish a decision-making scheme for loss discrimination. For example, Biaz [6], mBiaz [7], and the statistical packet loss discrimination (SPLD) [8] use packet inter-arrival times, the Zigzag [7] and the Spike [3] use relative one-way trip times, and the Vegas predictor [9] uses round trip times. The limitation of these methods is that the decision-making thresholds are difficult to be determined and are sensitive to application circumstances. Cen [7] compared the above methods and proposed a hybrid rule that selected one among the above loss differentiation algorithms (LDAs) depending upon observed network conditions, and obtained relatively good discrimination. A model-based learning organon was first proposed in Ref. [10]. A hidden Markov model (HMM) was used to develop a statistical model, which describes the distribution of RTTs of loss pairs.

Based on the derived model, congestion loss and wireless loss can be differentiated. Other studies [11–12] have shown that using only one feature may not provide good accuracy, and multiple indicators can have complementary functions. However, when combining multiple indicators, the relationship between the indicators and their thresholds are difficult to be determined.

### 3 Proposed loss differentiation algorithm

#### 3.1 The rationale of classification

In the proposed LDA, SVM [13] is employed to classify the loss causes. With the SVM, multiple indicators can be easily handled with a feature vector and no thresholds need to be empirically determined for each indicator. Furthermore, it has been proved that SVM can handle the classification problem successfully in small sample set and is simple to implement.

A classification task usually employs training and testing data sets that consist of several data instances. Each instance in the training set contains one ‘target value’ (class labels) and several ‘attributes’ (features). The goal of SVM is to produce a model that predicts the target values of the data instances in the testing set that are given the attributes only. Given a training set of feature-label pairs \( (x_i, y_i), i = 1,2,...,l \), where \( x_i \in \mathbb{R}^n \) and \( y \in \{ -1,1 \} \), the SVM requires the solution to the following optimization problem:

\[
\min_{\alpha,b,z} \left\{ \langle w, w \rangle + C \sum_{i=1}^{l} \xi_{i} \right\} \quad \text{s.t. } y_{i} [w^T \phi(x_i) + b] \geq 1 - \xi_{i}; \quad \xi_{i} \geq 0, i = 1,2,...,l
\]

This conditional optimization is achieved by Lagrange’s method of indeterminate coefficient \( \alpha_i \). Its dual is derived as follows:

\[
\max_{\alpha} \left\{ \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \right\}
\]

\[
\text{s.t. } y^T \alpha = 0; \quad 0 \leq \alpha_i \leq C, \quad i = 1,2,...,l
\]

where \( w \) is a weight coefficient vector, \( b \) is a bias term, \( \xi_i \) is a slack variable that indicates tolerances of misclassification, \( e \) is the vector of all ones, \( C > 0 \) is the penalty parameter of the error term, \( Q \) is an \( l \) by \( l \) positive semidefinite matrix, and \( Q_{ij} = y_i y_j K(x_i, x_j) \). Here, training vectors \( x_i \) are mapped into a higher (maybe infinite) dimensional space by function \( \phi \). The kernel function \( K(x, x^{'}) \) is introduced to reflect the relationship between the distance defined in the transformed space and the distance in the original space. Then,
all the calculation can be done using the kernel function and the actual \( \phi \), and the transformed space need not be known. The kernel function satisfies the following:
\[
K(x, x') = \phi(x)\phi(x')
\]

There are considerable typical kernels that can be used in SVM models. The radial basis kernel function (RBF) is by far the most popular choice of kernel types, taking the form as follows:
\[
K(x, x') = \exp\left(-\gamma \|x - x'\|^2\right), \quad \gamma > 0
\]

In this article, the RBF kernel is used in the experiments. Consequently, the optimization in Eq. (3) is reduced to a quadratic programming problem. The difficulty of solving the quadratic problem is the density of \( Q \), because \( Q_j \) is, in general, not zero. By using a library for support vector machines (LIBSVM) [http://www.csie.ntu.edu.tw/~cjlin/libsvm], the problem is solved by the sequence minimal optimization (SMO) method. After the solution \( \alpha \) is obtained, the decision function is:
\[
f(x) = \text{sgn}\left[\sum_{i=1}^{l} \alpha_i y_i K(x_i, x) + b\right]
\]

Here, a congestion loss is denoted as \( y = 1 \) and a wireless loss \( y = -1 \).

3.2 The choice of the inputs

Instead of using any single metric vulnerable to noise, the authors devise multi-metrics such that the complementary feature and noise independence of them can be effectively exploited. Metric ROTT denotes the one way delay of a packet. It is calculated using Eq. (7) at the receiver side whenever a packet arrives.

\[
P_k^{\text{ROTT}} = A^i - S^i
\]

where \( A^i \) is the arrival time of packet \( k \), and \( S^i \) is its sending time from the sender. Metric IAT denotes the arrival time difference between consecutive received packets, which is calculated as:

\[
P_k^{\text{IAT}} = A^i - A^{i-1}
\]

At the receiver, the instant when the packet is received is known and the moment when the packet was sent is the timestamp in the packet header. To compute the inputs, the authors define two observation windows that contain 5 packets before a loss event and 3 packets after the loss, respectively. The sequence of the concerned packets is shown in Fig. 1, and the notions used in the following texts are listed in Table 1. Here, it is assumed that the queue size of the bottleneck link is of 10 packets. Observations on the 5 samples (half of the queue size) should be able to show the varying trends of ROTTs and IATs before a loss event. Selecting the observation window of the 3 packets after a loss is in consideration that in TFRC, the loss of a packet is detected by the arrival of at least three packets with higher sequence numbers than that of the lost packet. To extract the

\[
\begin{align*}
&\text{BF} \quad \text{AF} \\
&P_{1,1} \quad P_{1,2} \quad P_{1,3} \quad P_{1,4} \quad P_{1,5} \quad \ldots \quad P_{2,1} \quad P_{2,2} \quad P_{2,3} \\
&\text{Legend} \quad \square \text{Received packet} \quad \square \text{Lost packet}
\end{align*}
\]

**Fig. 1** Packet sequence

**Table 1** Notations

<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF{}</td>
<td>Set of five packets before the loss event starts.</td>
</tr>
<tr>
<td>AF{}</td>
<td>Set of three packets after the loss event ends.</td>
</tr>
<tr>
<td>( P_{i,1} )</td>
<td>The ith packet in set BF{} \cdot (i \in {1,2,3,4,5})</td>
</tr>
<tr>
<td>( P_{i,2} )</td>
<td>The second packet in set BF{} \cdot (i \in {1,2,3})</td>
</tr>
<tr>
<td>( P_{i,3} )</td>
<td>The ROTT of the kth packet.</td>
</tr>
<tr>
<td>( P_{i,4} )</td>
<td>The inter-arrival time of the ith packet.</td>
</tr>
<tr>
<td>( P_{i,5} )</td>
<td>The average ROTT of the packets received since the end of the last loss event to the beginning of the current loss event</td>
</tr>
<tr>
<td>avg(IAT)</td>
<td>The average inter-arrival time of the second packets received since the end of the last loss event to the first packet after the end of the current loss event</td>
</tr>
<tr>
<td>( \text{min}(IAT) )</td>
<td>The minimal inter-arrival time observed so far</td>
</tr>
<tr>
<td>( \text{min}(\text{ROTT}) )</td>
<td>The minimal ROTT observed so far</td>
</tr>
</tbody>
</table>

feature vectors, the minimal, maximal, and average values of ROTTs and IATs in different time intervals are computed. Using ratios instead of absolute values in the features mentioned above enables the proposed scheme to be robust against variations of network conditions. The extracted 12 features and their detailed descriptions are listed in Table 2. Furthermore, the authors explain the rationale of feature selection. Since distributions of relative one-way trip time of packets fore-and-aft the loss event caused by congestion are different from those of wireless loss in some aspects, when ROTT is plotted against time at the receiver, one can observe spikes during congestion [3]. The loss during the spike periods can be classified as congestion loss. Though it is not easy to precisely determine where the spike area is, one can select indicators that can reflect the trend of ROTTs as the features. The increase of ROTTs with time indicates that the selection. Since distributions of relative one-way trip time of packets fore-and-aft the loss event caused by congestion are different from those of wireless loss in some aspects, when ROTT is plotted against time at the receiver, one can observe spikes during congestion [3]. The loss during the spike periods can be classified as congestion loss. Though it is not easy to precisely determine where the spike area is, one can select indicators that can reflect the trend of ROTTs as the features. The increase of ROTTs with time indicates that the...
ROTTs. Therefore, if the current loss is located at the falling part or around a peak of a ROTT curve, and the ROTTs within the observation windows are far larger than the minimum of ROTT (as measured by $\max(P_{\text{ROTT}})/\min(\text{ROTT})$ and $\max(P_{\text{ROTT}})/\min(\text{ROTT})$ in Table 2), the loss is often classified as a congestion loss. On the other hand, if the loss happens randomly in the flat part or in the rising part of a ROTT curve, and meanwhile, the average of ROTTs within the observation windows maintains a comparable value to the minimal ROTT (as measured by $\max(P_{\text{ROTT}})/\min(\text{ROTT})$), the loss is most likely a wireless loss. In addition, the IAT is a good complement to the ROTT. If the packet, immediately after the lost packets, arrives exactly around the time when it should arrive, it can be assumed that the lost packets are properly transmitted and lost because of wireless errors. Otherwise, it is attributed to congestion [7]. For example, $IAT_{\text{avg(IAT)}}/\max(IAT)$ may have a comparable value to the burst loss length for a wireless loss. In a competing environment with multiple flows, the statistical values of IATs around a loss can also be used to characterize the loss [8]. For example, it is a good indicator of a wireless loss if the ratio of $IAT_{\text{avg(IAT)}}/\max(IAT)$ is larger than 1.

### Table 2

<table>
<thead>
<tr>
<th>Features</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BurstLossLength</td>
<td>The number of lost packets in one burst loss event</td>
</tr>
<tr>
<td>$\text{ROTT}/\min(\text{ROTT})$</td>
<td>$\text{ROTT}$ divided by the minimal ROTT observed so far</td>
</tr>
<tr>
<td>$\text{avg}(P_{\text{ROTT}})/\text{ROTT}$</td>
<td>The average ROTT of the packets in $\text{BF}{}$ divided by $\text{ROTT}$</td>
</tr>
<tr>
<td>$\text{max}(P_{\text{ROTT}})/\min(\text{ROTT})$</td>
<td>The maximal ROTT of the packets in $\text{BF}{}$ divided by the minimal ROTT observed so far</td>
</tr>
<tr>
<td>$\text{avg}(P_{\text{ROTT}})/\text{ROTT}$</td>
<td>The maximal ROTT of the packets in $\text{AF}{}$ divided by the minimal ROTT observed so far</td>
</tr>
<tr>
<td>${\text{avg}(P_{\text{ROTT}}) - \min(\text{ROTT})}/\text{avg}(P_{\text{ROTT}})$</td>
<td>The difference between the average ROTT in $\text{BF}{}$ and the minimal ROTT divided by the average ROTT in $\text{AF}{}$</td>
</tr>
<tr>
<td>$\text{IAT}/\text{avg(IAT)}$</td>
<td>$\text{IAT}$ divided by the average LAT observed so far</td>
</tr>
<tr>
<td>$\text{max}(P_{\text{LAT}})/\text{avg}(P_{\text{LAT}})$</td>
<td>The maximal LAT of packets in $\text{AF}{}$ divided by the average LAT of packets in $\text{BF}{}$</td>
</tr>
<tr>
<td>$\text{LAT}<em>{\text{first}}/\text{max}(P</em>{\text{LAT}})$</td>
<td>The LAT of the first packet in $\text{BF}{}$ divided by the maximal LAT of packets in $\text{AF}{}$</td>
</tr>
<tr>
<td>$\text{LAT}_{\text{first}}/\text{min(IAT)}$</td>
<td>The LAT of the first packet in $\text{AF}{}$ divided by the minimal LAT observed so far</td>
</tr>
<tr>
<td>$\text{avg}(P_{\text{LAT}})/\text{min(IAT)}$</td>
<td>The average LAT of packets in $\text{BF}{}$ and $\text{AF}{}$ divided by the minimal IAT observed so far</td>
</tr>
</tbody>
</table>

### 3.3 Training process

To generate the losses, simulations are carried out with the network simulator ns-2 [http://www.isi.edu/nsnam/ns]. Two typical network topologies as shown in Fig. 2 are simulated. In Fig. 2(a), the last hop to the receiver is a wireless link, which is called wireless last hop (WLH), and in Fig. 2(b), the backbone between two lans is a wireless link, which is called wireless backbone (WB). The network parameters are listed under network configuration 1 in Table 3 with several values in the flow number and the wireless loss rate. The error model of wireless link is the Gilbert model and the simulated average loss rates are 1%, 3%, and 7%, respectively [6]. A file transfer protocol (FTP) application in TCP is established from the sender to the receiver as the background traffic to compete with TFRC flows, and $N (N=2,4,6,8,10)$ TFRC flows join the simulations according to the schedule of events illustrated in Fig. 3 to produce different levels of congestion. One simulation lasts 50 s and this procedure is repeated several times with three different wireless loss rates in two network topologies respectively. Thus, about 2 000 losses are collected in the above procedure and the feature vectors are extracted from the TFRC flows’ observations according to Table 2. The first 1 000 samples are selected as the training set and the remaining 1 000 samples constitute the testing set.
Table 3 Network settings in the simulations

<table>
<thead>
<tr>
<th>Network configuration 1</th>
<th>Network configuration 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network topology</td>
<td>WLH</td>
</tr>
<tr>
<td>Access link bandwidth/ (Mbit·s(^{-1}))</td>
<td>10</td>
</tr>
<tr>
<td>Access link latency/ ms</td>
<td>1</td>
</tr>
<tr>
<td>Backbone bandwidth/ (Mbit·s(^{-1}))</td>
<td>4</td>
</tr>
<tr>
<td>Backbone latency/msec</td>
<td>10</td>
</tr>
<tr>
<td>Last hop bandwidth/ (Mbit·s(^{-1}))</td>
<td>11</td>
</tr>
<tr>
<td>Last hop latency/ ms</td>
<td>5</td>
</tr>
<tr>
<td>CBR bitrate/ (kbit·s(^{-1}))</td>
<td>500</td>
</tr>
<tr>
<td>Packet size/B</td>
<td>1000</td>
</tr>
<tr>
<td>Queueing policy</td>
<td>DropTail</td>
</tr>
<tr>
<td>Buffer size</td>
<td>10 packets</td>
</tr>
<tr>
<td>Flow number</td>
<td>TCP + N(TFRC), (N=2,4,6,8,10)</td>
</tr>
<tr>
<td>Wireless loss rate</td>
<td>1%, 3%, 7%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network configuration 2</th>
<th>WLH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network topology</td>
<td>WLH</td>
</tr>
<tr>
<td>Access link bandwidth/ (Mbit·s(^{-1}))</td>
<td>10</td>
</tr>
<tr>
<td>Access link latency/ ms</td>
<td>1</td>
</tr>
<tr>
<td>Backbone bandwidth/ (kbit·s(^{-1}))</td>
<td>(2−N)*130</td>
</tr>
<tr>
<td>Backbone latency/msec</td>
<td>20</td>
</tr>
<tr>
<td>Last hop bandwidth/ (kbit·s(^{-1}))</td>
<td>150</td>
</tr>
<tr>
<td>Last hop latency/msec</td>
<td>10</td>
</tr>
<tr>
<td>Flow number</td>
<td>N(TFRC)</td>
</tr>
<tr>
<td>Flow bit rate/ (kbit·s(^{-1}))</td>
<td>150</td>
</tr>
<tr>
<td>Packet size/B</td>
<td>762</td>
</tr>
<tr>
<td>Queueing policy</td>
<td>DropTail</td>
</tr>
<tr>
<td>Buffer size</td>
<td>Max (bandwidth/60, 6) packets</td>
</tr>
<tr>
<td>Wireless loss rate/ %</td>
<td>7</td>
</tr>
</tbody>
</table>

Fig. 3 The simulation schedule in collecting losses

In the training process of SVM, the authors conduct a five-fold cross validation process to determine the optimal parameter \(C\) and \(\gamma\). They first divide the training set into five subsets of equal size. Sequentially, one subset is tested using the classifier trained on the remaining four subsets. Thus, each instance of the whole training set is predicted once. The cross-validation accuracy is the percentage of correctly classified data. The optimal parameters obtained using the above process are \(C = 32,768\) and \(\gamma = 0.03125\). These parameters are used to conduct a more efficient training and to accurately predict unknown data (i.e., testing data) in the next section.

3.4 Model evaluations

In this subsection, a criterion is defined to evaluate the accuracy of the proposed loss differentiation algorithm and an eclectic factor is introduced to the decision function of SVM to get a tradeoff in the classification accuracy of two types of losses.

An evaluation metric similar to that in Ref. [10] is used to measure the quality of the proposed loss differentiation algorithm. Let \(T = \{C, W\}\) \(C\) for 'congestion loss' and \(W\) for 'wireless loss') be the set of loss types. \(P(t|t')\) is defined as the probability of a loss being classified as loss type \(t\), while it actually belongs to loss type \(t'\), for \(t, t' \in T\). Therefore, \(P(C|C)\) is the correct classification rate of congestion loss, and \(P(W|W)\) is that for wireless loss. If \(P(C|C)\) is low, it indicates that some congestion losses are treated as wireless loss, and the sender will not adjust the sending rate properly in response to the congestion. On the other hand, if \(P(W|W)\) is low, the sender will probably reduce the sending rate unnecessarily. This can induce inefficient utilization of bandwidth but it may not adversely affect the performance of other network traffic. Generally speaking, different application environments may require emphasizing accuracy of one type and reducing demand for the other type. For the Internet to keep TCP-friendliness, the authors emphasize that the misclassification of congestion loss should be restricted to few. Thus, the authors add an eclectic factor \(\beta\) to the decision function of the trained SVM, and the decision function is extended as:

\[
 f(x) = \text{sgn} \left\{ \sum_{i=1}^{n} y_i K(x_i, x) + b \right\} + \beta
\]

(9)

By varying \(\beta\), the SVM can favor the classification accuracy of congestion loss or wireless loss.

4 Simulations and evaluations

In this section, first, the relation between the eclectic factor \(\beta\) and the classifying accuracy is investigated. Then, the influence is examined of main network parameters, the number of competing flows, the wireless error rate, and the queue size of the bottleneck link on the performance of the proposed classification algorithm. Finally, the accuracy of the scheme is compared to the ZBS scheme [7].
Using the SVM trained earlier and the testing set mentioned in the last section, the authors get the relation between the accuracy and $\beta$, as shown in Fig. 4. It can be seen that:

1) The loss differentiation algorithm shows high accuracy.
2) By adjusting $\beta$, it is easy to focus on the accuracy of one type at the expense of the other type. For example, the application expects the accuracy of congestion loss to be no less than 95%. It can be seen from the dashed line in Fig. 4 that $P(W|W)$ is no less than 75%, which is still reasonably high.

In the following experiments, $\beta=0.6$ is selected to ensure the accuracy of congestion loss rather than the other.

The proposed algorithm is evaluated by classifying losses collected in simulations using the ns-2 network simulator. The simulations are configured to have a network topology as shown in Fig. 2. The following network parameters that may affect the accuracy of loss classification are examined.

1) The number of competing flows
The network parameters are the same as network configuration 1 in Table 3. The authors vary the number of competing TFRC flows from 2 to 10 in increments of 2 to examine its impact on accuracy. Under each condition, 1000 losses are collected. The results are shown in Fig. 5 and Fig. 6. From the figure, it can be seen that as the flow number increases, the classification accuracy has a slight decrease. This is because when too many flows compete, the bottleneck link is overloaded; the average values of ROTTs around a congestion loss and around a wireless loss both increase, and their varying trends are not obvious. When 8 or 10 flows are competing, the $P(W|W)$ can still maintain a high level (above 90%). A high accuracy of congestion losses is helpful to maintain friendliness to other TCP flows. Though $P(W|W)$ decreases slightly, it may not adversely affect the performance of other network traffic in heavily loaded network environment.

2) Wireless packet loss rate
The wireless packet loss rate (1%, 3%, and 7%) is changed by varying the duration in the BAD state of the Gilbert error model. The other parameters are the same as network configuration 1 in Table 3. The results show that the classification accuracy of wireless losses is increased with the increase of the packet loss rate. A possible explanation is that longer burst loss length is conducive to making the indicators of IATs (for example $P_{\text{IAT}}^\text{min}(\text{IAT})$ and $\text{IAT}^\text{avg}(\text{IAT})$ in Table 2) more feasible.

3) Queue size of bottleneck link
The queue size in the bottleneck link is varied, and other parameters are maintained to be the same as network configuration 1 in Table 3. There are 6 TFRC competing flows for observation. From Fig. 7, it can be seen that the queue size has strong influence on the classification accuracy. The accuracy of congestion loss is improved with the increase of queue size. The reason may be that a remarkably long queuing delay can be observed with a large queue size when congestion exists. If the queue size is very small, packets are soon discarded because of overflow of the queue in congestion, and may not be distinguished from wireless loss in ROTTs. However, the queue size of the 5 packets is only a fictitious
value used to examine the extreme condition. In practical networks, very small queue size is seldom used. The scheme performs well when the queue size is larger than 10.

![Fig. 7 Accuracy with different queue size](image)

From the above examinations, it is demonstrated that the scheme is a feasible approach to discriminating two classes of losses under most network conditions. In addition, it is observed that the wireless link position (WLH or WB) has little influence on classification accuracy under various simulated conditions.

4) Comparison with ZBS scheme

The scheme is compared with the ZBS [7] scheme. The authors have evaluated the ZBS hybrid LDA across different topologies with different flows, and found that in most cases, it closely matches or exceeds the performance of the best base algorithm under the same scenario [7]. The ZBS scheme chooses one base algorithm a time from ZigZag, mBiaZ, and Spike according to the current network condition. The choice is mainly based on the relationship between the IAT and its minimum, as follows:

```plaintext
if (ROTT < (min(ROTT) + 0.05*min(IAT))
    use Spike;
else
    if (IAT < 0.875)
        use ZigZag;
    else if (IAT < 1.5)
        use mBiaz;
    else if (IAT < 2.0)
        use ZigZag;
    else
        use Spike;
```

The proposed scheme is tested in WLH network under the same condition as in Ref. [7]. The simulation parameters are under network configuration 2 in Table 3. The bottleneck bandwidth is always set to be 86% of the bandwidth required by all flows. Therefore, the congestion level is approximately unvaried. From the results shown in Fig. 8, it can be seen that the scheme is more stable under varying network conditions. Both the accuracies of congestion loss and wireless loss are considerably higher than that of the ZBS scheme in most of the testing cases.

![Fig. 8 Accuracy comparison with ZBS scheme](image)

5) Other issues

In practice, the proposed classifier can first be trained by data collected from an ns-2 simulated scenario, which has the same configuration as real network scenario. After the training, the classifier can obtain optimal parameters for this scenario and conduct classification in the real network. The integration of the classifier into the receiving device is convenient without requiring any support from intermediate nodes or modifications on standard protocols.

The computational complexity in the training process mainly exists in the decomposition method and reconstructing the gradient [http://www.csie.ntu.edu.tw/~cjlin/libsvm]. When caching is used to store recently used $Q_j$, the cost per iteration is only $O(nl)$, here, $l$ will gradually decrease during iterations with shrinking incorporated. The complexity in the classification process is $O(1)$.

5 Conclusions

In this article, a machine learning method is proposed to differentiate the losses caused by congestion and wireless link errors in wired-cum-wireless networks. It is based on the outstanding learning algorithm, SVM, and the ratios of measured indicators are organized as the input features of the SVM to avoid dependence on particular application scenarios. In addition, an eclectic factor is introduced to the trained
SVM to enhance the classification accuracy of congestion loss. The loss differentiation algorithm is evaluated under various network configurations and comparisons indicate that the classifier is effective in most scenarios.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (60772114).

References


(Editor: ZHANG Ying)