Early Stage Internet Traffic Identification Using Probabilistic Neural Networks

Lizhi Peng*

Shandong Provincial Key Laboratory for Network Based Intelligent Computing, University of Jinan, Jinan, 250022, P. R. China.

*Corresponding author. Email: plz@ujn.edu.cn

Manuscript submitted September 2, 2014; accepted April 18, 2015.
doi: 10.17706/ijcce.2015.4.6.417-425

Abstract: Identifying network traffics at their early stages accurately is very important for network management and security. Recent years, more and more studies have devoted to find effective machine learning models to identify traffics with the few packets at the early stage. In this paper, we try to build an effective early stage traffic identification model by applying probabilistic neural networks. Three network traffic data sets including two open data sets are used for the study. Packet sizes and three statistics are applied as features. Six classical classifiers are employed as the comparing methods in the identification experiments. PNN outperforms the other methods for most cases in the identification experiments, and it behaves very well for both of accuracy and AUC. Thus, PNN is effective for early stage traffic identification.

Key words: Early stage traffic identification, probabilistic neural networks, machine learning.

1. Introduction

Recent years, early stage traffic identification has caught enough interests at the research community. Most traditional machine learning based traffic identification techniques extract features on a whole traffic instance [1]-[3]. The most extracting method is presented by A. W. Moore *et al*. in 2005 [4]. They extract 248 statistical features based on whole traffics, such as maximum, minimum and average values of packet size, RTT. And classifiers using these statistical features can get very high performances in traffic identification. However, in real circumstances, it makes no sense to recognize Internet traffics when they have ended. Therefore, some researchers have turned to find effective models which are able to identify Internet traffics at their early stage. And this makes early stage identification to become a hot topic in traffic identification researches [5]. B. Qu *et al*. have studied the problem of accuracy of early stage traffic identification, and found that it is possible to identify traffic accurately at its early stage [6].

It is relatively hard to recognize a flow by only using several early stage packets. Thus, the key problem of early stage traffic identification is to find out effective features in early stage of traffics. L. Bernaille *et al*. presented a famous early stage traffic identification technique in 2006 [7]. They use the size of the first few data packets of each TCP flow as the features, and by applying K-means clustering technique, they got high identification rates for 10 types of application traffics. A. Este *et al*. have proved in 2009 [8] that early stage packets of an Internet flow carry enough information for traffic classification. They analyzed round trip time (RTT), packet size, inter-arrival time (IAT) and packet direction of early stage packets and found that packet size is the most effective feature for early stage classifications. N. Huang *et al*. have studied the early stage application characteristics and used them for classification effectively in 2008 [9]. Recently, they extracted early stage traffic features by analyzing the negotiation behaviors of different applications. They use packet size (PS) and inter packet time (IPT) of the first 10 packets for some classifiers, while for other classifiers, they use average and standard deviation values of PS and IPT of the early packets. They applied these features for machine learning based classifiers with high performances [10]. B. Hullár *et al.* proposed an automatic machine learning based method consuming limited computational and memory resources for P2P traffic identification at early stage [11]. A. Dainotti *et al.* [12] construct high effective hybrid classifiers and apply a hybrid feature extraction method for early stage traffic classification. T. T. T. Nguyen *et al.* use statistical features derived from sub-flows for timely identification of VoIP traffics [13], they extend the concept of early stage to ”timely”, since a sub-flows refers to a small number of most recent packets taken at any point in a flow’s lifetime. A. Rizzi *et al.* proposed a highly efficient neuro-fuzzy system for early stage traffic identification [14].

Probabilistic neural network (PNN) [15] is a kind of neural network which uses Bayes inference theory for classification tasks. It has been widely applied for classification and pattern recognition [16]-[18]. Comparing with other neural networks, PNN has many attractive and unique characteristics.

Contributions: In this paper, we set out to create an effective early stage traffic identification model by applying probabilistic neural networks. Three network traffic data sets including two open data sets are used for the study. Packet sizes and three statistics are applied as features. Six classical classifiers are employed as the comparing methods in the identification experiments. The experimental results show that PNN outperforms the other methods for most cases.

The rest of the paper is organized as follows: Section 2 illustrates the basic model of probabilistic neural networks. We introduce the characteristics of the selected data sets and features in Section 3 and 4, respectively. The experimental settings including the comparing methods and the performance measurements are given in Section 5. And the details of experimental results and analysis are given in Section 6, and we also do some discussions in this section. Finally, we make some conclusions in Section 7.

2. Probabilistic Neural Networks

Probabilistic neural network (PNN) [15], a widely used nonlinear pattern classification technique, is a kind of neural network which uses Bayes decision rule for Bayes inference. Consider a *m* categories classification task for which $\theta_1, \theta_2, ..., \theta_m$ is its *m* categories. The decision of objective θ is based on a set of measurements represented by the n-dimensional vector $X^T=[x_1, x_2, ..., x_n]$. Then for a category θ_q, the Bayes decision rule is:

$$d(X) = \theta_q \text{ if } h_q l_q f_q(X) > h_k l_k f_k(X),\ k \neq q. \tag{1}$$

where $f_q(X)$ and $f_k(X)$ are the probability density functions of category *q* and *k* respectively. l_q and l_k are the priori probabilities of category θ_q and θ_k. l_q is the loss function associated with the decision $d(X) \neq \theta_q$ when $\theta = \theta_q$. l_k is the loss function associated with the decision $d(X) \neq \theta_k$ when $\theta = \theta_k$.

For most actual cases, *l* is known can be estimated accurately. *h* is the same for each category. Therefore the key to use the Bayes decision rule is to estimate the probability density functions based on training patterns. A widely used and effective estimator proposed by Parzen [19] is as follows:

$$f_q(x) = \frac{1}{2\pi^{n/2}\sigma^n n_q}\sum_{i=1}^{n_q}[-\frac{(X-X_{qi})^T(X-X_{qi})}{2\sigma}] \tag{2}$$

where X_{qi} is the *i*th training sample vector from category θ_q, n_q is the number of samples from θ_q and σ is the smooth parameter.

Fig. 1 shows the organization of a typical probabilistic neural network which has three layers: input layer, hidden layer and output layer. The hidden layer is also called pattern layer in PNN. It should be noted that some literatures applies an additional layer which simply makes decision by comparing probabilities. This decision can be accomplished by a very simple numeric comparing algorithm. Therefore we omit this layer for concise network structures. The input layer comprises n units which merely supply inputs to all of the pattern units. Each pattern unit is a pattern neuron shown in Fig. 2. A pattern neuron firstly form a dot product Z_i using input vector X and a weight vector W_i^{xh}, i. e. $ $Z_i = X \bullet W_i^{xh}$. Here W_i^{xh} belongs to W_{xh} which is the set of weights between input and pattern layer. And then the pattern neuron perform a nonlinear operator on Z_i before outputting its activation level to the output units. The nonlinear operator applied in the pattern neuron is usually a radial basis function, so the input-pattern layer is usually called radial basis layer. A commonly used nonlinear operator is $e^{(Z_i-1)/\sigma^2}$. If X and W_i are both normalized to unit length, then the output of the ith pattern unit is

$$net(H_i) = e^{-\frac{(W_i^{xh}-X)^T(W_i^{xh}-X)}{2\sigma^2}} \tag{3}$$

An output unit uses the outputs of pattern units as its inputs, and calculates its output as follows:

$$y_j = \sum_{i=1}^{h} W_{ij}^{hy} net(H_i), \quad j=1,\ 2,\ \ldots,\ m. \tag{4}$$

where h is the number of pattern unit, W_{ij}^{hy} is the weight between ith pattern unit and jth output unit. The output y_j denote the probability of X from category θ_j. Therefore, if

$$y_q = MAX\{y_1, y_2, \ldots, y_m\} \tag{5}$$

then X is classified by PNN as θ_q.

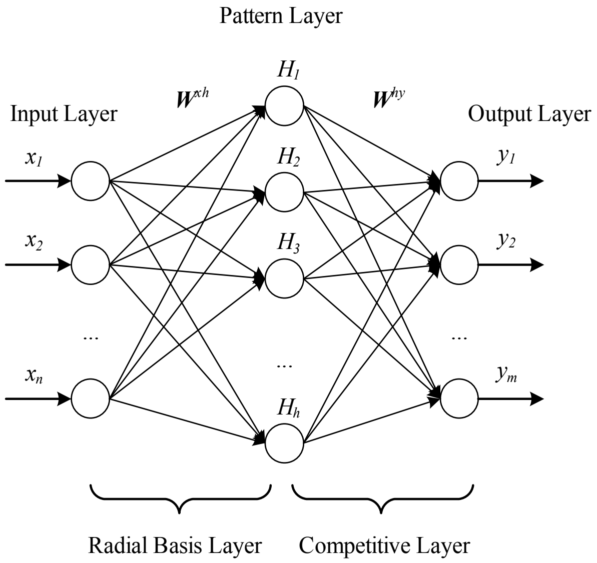


Fig. 1. Structure of probabilistic neural network.

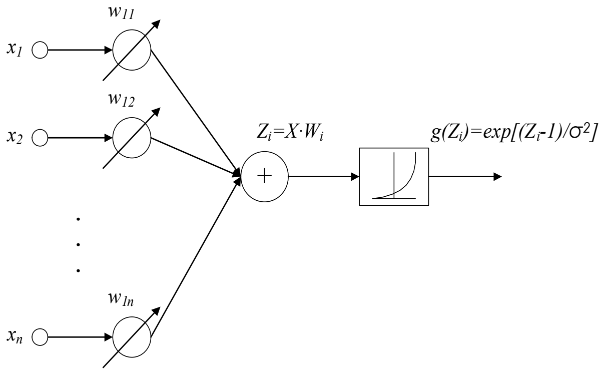


Fig. 2. The pattern neuron.

3. Data Sets

3.1. Auckland II Traffic Traces

Auckland II is a collection of long GPS-synchronized traces taken using a pair of DAG 2 cards at the University of Auckland which is available at [20]. There are 85 trace files which were captured from November 1999 to July 2000. Most traces were targeted at 24 hour runs, but hardware failures have resulted in most traces being significantly shorter. We selected two trace files captured at Feb. 14 2000 (20000214-185536-0.pcap and 20000214-185536-1.pcap) for our study. The traces include only the header bytes, with a maximum amount of 64 bytes for each frame, while the application payload is fully removed. And all IP addresses anonymised using Crypto-Pan AES encryption. We selected eight main types from Auckland II traces for our studies.

3.2. UNIBS Traffic Traces

UNIBS is another opening traffic traces developed by Prof. F. Gringoli and his research team, available at [21]. They developed a useful system namely GT [22] to application ground truths of captured Internet traffics. The traces were collected on the edge router of the campus network of the University of Brescia on three consecutive working days (Sept. 30, Oct. 1 and Oct. 2 2009). They are composed of traffic generated by a set of twenty workstations running the GT client daemon. Traffics were collected by running Tcpdump [23] on the Faculty's router, which is a dual Xeon Linux box that connects the network to the Internet through a dedicated 100Mb/s uplink. 99% flows in UNIBS are TCP flows. We also chose eight main types in UNIBS for our study.

3.3. UJN Traffic Traces

The third data set is collected in a laboratory network of University of Jinan using Traffic Labeler (TL) [24]. We deployed 10 TL instances on Windows user hosts in the laboratory network of Provincial Key Laboratory for Network Based Intelligent Computing. A mirror port of the uplink port of the switch was set, and a data collector was deployed at the mirror port. The deployed TL instances ran at work hours every day. The data collecting process lasted two days in May 2013. Again, flows with no more than six non-zero payload packets are also filtered.

4. Features

Packet size: Packet size has been proven to be the most effective original packet level feature in early stage of traffics [8]. We use the packet sizes of the first six packets as the packet-level features since we have proven that the first six packets are most effective for early stage feature extraction [25].

Average: The average is also known as the arithmetical mean, which is an extensively used statistical indicator. This feature is calculated as follows:

$$avg = \sum_{i=1}^{n} ps_i \tag{6}$$

Standard deviation: The standard deviation shows how much variation or dispersion from the average exists. And the feature is defined as:

$$stdev = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(ps_i - avg)^2} \tag{7}$$

where *n* is the number of packets, i. e. 6 in this study.

Maximum and minimum: The maximum and minimum packet sizes are also applied in the study, and we use the abbreviations of *max* and *min* respectively.

5. Experimental settings

5.1. Compared Methods

We execute our identification experiments using six well-known machine learning classifiers. We use Weka data mining software [26] as our experiment tool. All classifiers are run in Weka and all processed data sets are formatted into the Weka data file with the extension name of “arff”. The classifiers we selected fall into five categories according to Weka:

Bayes: Bayes classifiers are based on Bayes theorem, which is widely applied in many engineering areas. In this study, we choose Naive Bayes classifier and Bayesian network (BayesNet) as Bayes classifiers.

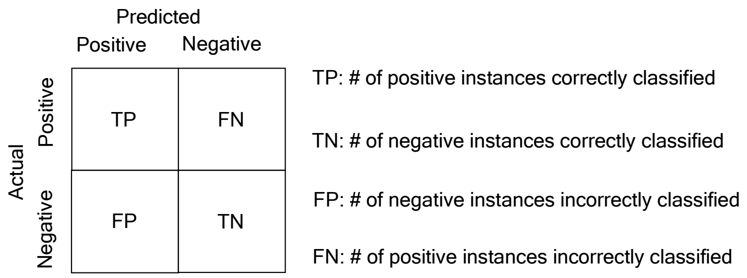
Meta: Strictly speaking, meta classifier is a kind of classification framework based on a specific classifier. This technique firstly trains a group of “weak learn”, and then generate a “strong learn” based on the weak learns. We choose adaptive Boost M1 (AdaBoost) for our study.

Rule: As the name suggests, a rule based classifier extracts rules using a specific policy, e. g. probability and decision trees, and uses the rules to classify testing data. OneR is selected for this category in this study.

Functions: Weka refers all classifiers based on specific functions to this category. We choose support vector machine (SVM) and radial basis function neural network (RBFNetwork) for this category.

5.2. Performance Measures

The confusion matrix is the basis in measuring a classification task. Fig. 3 shows a typical confusion matrix of a binary classification. In this study, the following measures are used:



TP: # of positive instances correctly classified

TN: # of negative instances correctly classified

FP: # of negative instances incorrectly classified

FN: # of positive instances incorrectly classified

Fig. 3. Confusion matrix.

- **Accuracy**

Classification accuracy (Acc) is defined as the total proportion of all correctly classified instances:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{8}$$

- **Area Under Curve**

The receiver operating characteristic (ROC) curve [27] is a 2D graphical illustration of the trade-off between the TP rate (TPR) and FP rate (FPR). The TPR is also called sensitivity (*Sens*), and the FPR is related to another general measure namely specificity (*Spec*), and they are defined as follows:

$$Sens = \frac{TP}{TP + FN}, \quad Spec = \frac{TN}{TN + FP} \tag{9}$$

The ROC curve illustrates the behavior of a classifier without considering the class distribution or misclassification cost. The area under the ROC curve (AUC) is computed by the confusion matrix values in relation to the TPR and FPR:

$$AUC = \frac{1+TPR-FPR}{2} = \frac{Sens+Spec}{2} \tag{10}$$

6. Results and Analysis

6.1. Results

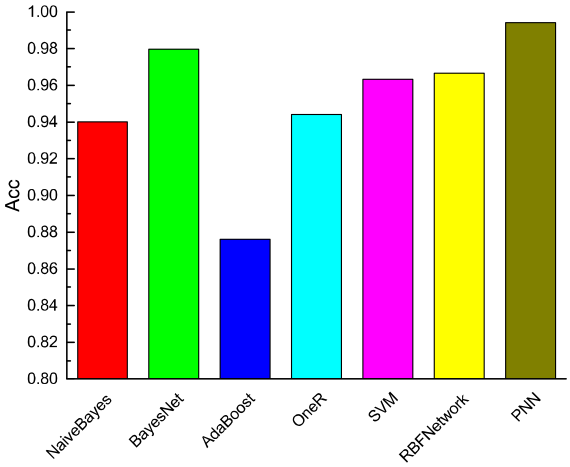


Fig. 4. Accuracy results of Auckland II data set.

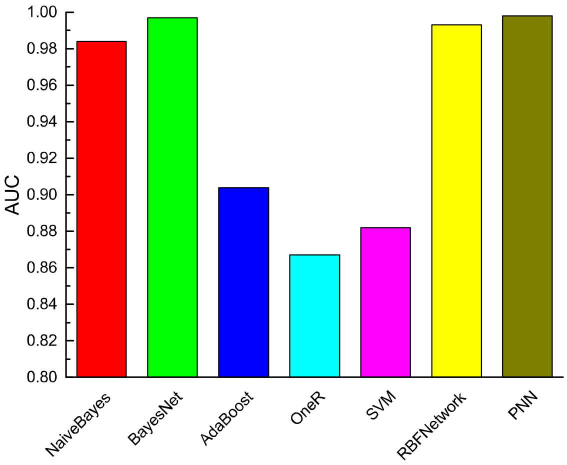


Fig. 5. AUC results of Auckland II data set.

Fig. 4 and Fig. 5 show the experimental results of the Auckland II data set. Firstly, we can see that PNN outperforms all the other algorithms for both accuracy and AUC. It means that PNN is the best performed algorithm for this data set. Secondly, NaiveBayes and RBFNetwork do not show very high performances for the accuracy. However, they get high AUC values. So, we say that the two algorithms are able to conduct good trade-off among the eight kinds of class in Auckland II data set. On the contrary, OneR and SVM show good accuracy performances and poor AUC performances: Their accuracies are far higher than that of AdaBoost, but their AUC values are lower that of AdaBoost.

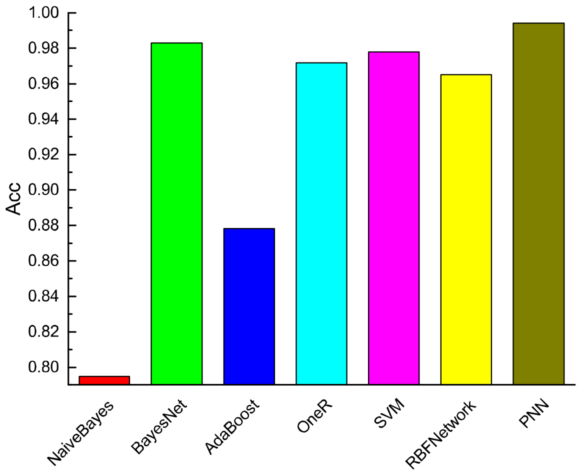


Fig. 6. Accuracy results of UNIBS data set.

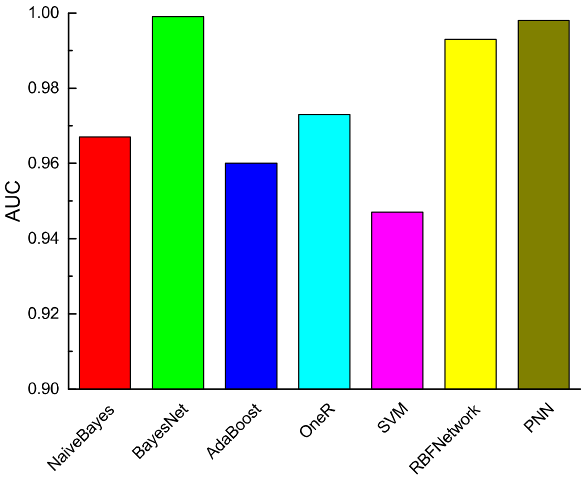


Fig. 7. AUC results of UNIBS data set.

The results of the UNIBS data set that given in Fig. 6 and Fig. 7 show different patterns. PNN again gets the highest accuracy. However, BayesNet defeats PNN for AUC with a slight edge. BayesNet shows good class trade-off ability for the UNIBS data set. NaiveBayes also shows good class trade-off ability: Its accuracy is the lowest one, but its AUC is fairly high. AdaBoost and RBFNetwork show stable performances for accuracy and AUC. And again OneR and SVM do not perform very well for AUC in contrast with their good behaviors for the accuracy.

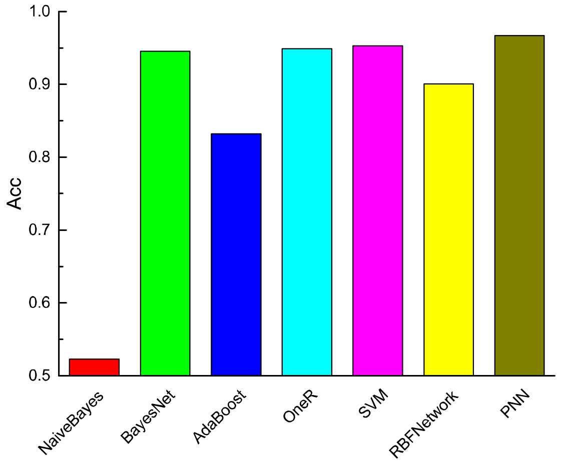


Fig. 8. Accuracy results of UJN data set.

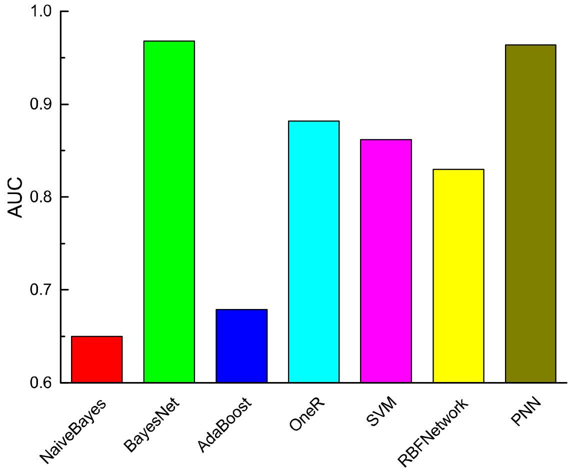


Fig. 9. AUC results of UJN data set.

Different from the results of the former two data sets, the results of the UJN data set in Fig. 8 and Fig. 9 show a unique pattern. Firstly, NaiveBayes shows poor performances for both accuracy and AUC. Secondly, AdaBoost does not behave so stably as it does for the former two data sets: It get an accuracy value of 0.8322, while its AUC value is 0.6790. It can be inferred that this algorithm is not able to get good class trade-off abilities for the UJN data set. RBFNetwork also does not show good performances for AUC this time. Similar to the results of the UNIBS data set, PNN again gets the highest accuracy, but was defeated by BayesNet for AUC with a slight edge.

6.2. Analysis and Discussions

According to the experimental results, some lessons can be learned:

- First of all, the probabilistic neural networks shows high performances in early stage traffic identification. As can be seen, PNN gets the highest accuracy and AUC values for all of the three data sets. The high accuracy values mean that PNN is able to achieve high total identification rates, and the high AUC values say that PNN is able to get good trade-off among different traffic types, especially for imbalanced traffic type distributions.
- For an early stage traffic identification method, performance evaluation using AUC is as important as the total identification rate, as AUC is a measurement to evaluate the tade-off abilities of a method. In our experiments, some classifiers show high performances for the accuracy measurement, but they do not perform well for the AUC measurement. e. g. AdaBoost gets a considerably high accuracy value for the UJN data set. However, its AUC value of this data set is very low. And the situation of OneR is the same for the Auckland II data set.

7. Conclusion

In this paper, we have tried to build an effective early stage traffic identification model using probabilistic neural networks. We use three traffic data sets include two opening data sets for the experimental evaluations. And six classical classification algorithms are applied for experimental comparisons. According to the experimental results, we conclude that probabilistic neural network is effective for early stage traffic identification. As can be seen from the experimental results that PNN outperformed the other six classifiers for most experimental cases. Furthermore, PNN does not only get high total identification rates, but also show good trade-off among different traffic types. And this is very important for traffic identification, especially for the cases with an imbalanced data distribution.

Acknowledgment

This research was partially supported by the National Natural Science Foundation of China under grant No. 61472164, No. 61402475, No. 61373054, the Provincial Natural Science Foundation of Shandong under grant No. ZR2012FM010, No. ZR2011FZ001, and No. ZR2012FQ016.

References

[1] Este, A., Gringoli, F., & Salgarelli, L. (2009). Support vector machines for TCP traffic classification. *Computer Networks*, *53(14)*, 2476-2490.

[2] Li, W., & Moore, A. W. (2007). A machine learning approach for efficient traffic classification. *Proceedings of IEEE MASCOTS'07* (pp. 310-317). IEEE Press.

[3] Moore, A. W., & Zuev, D. (2005). Internet traffic classification using Bayesian analysis techniques. *Proceedings of ACM SIGMETRICS'05* (pp. 50-60). Banff, AB, Canada.

[4] Moore, A. W., Zuev, D., & Crogan, M. (2005). *Discriminators for Use in Flow-Based Classification*. London, UK: Intel Research Tech.

[5] Dainotti, A., Pescapé, A., & Claffy, K. C. (2012). Issues and future directions in traffic classification. *IEEE Network*, *26(1)*, 35-40.

[6] Qu, B., Zhang, Z., Guo, L., *et al.* (2012). On accuracy of early traffic classification. *Proceedings of IEEE 7th International Conference on Networking, Architecture and Storage (NAS)* (pp. 348-354). Xiamen, Fujian, China.

[7] Bernaille, L., Teixeira, R., Akodkenou, I., *et al.* (2006). Traffic classification on the fly. *Proceedings of ACM SIGCOMM'06* (pp. 23-26). Pisa, Italy.

[8] Este, A., Gringoli, F., & Salgarelli, L. (2009). On the stability of the information carried by traffic flow features at the packet level. *ACM SIGCOMM Computer Communication Review*, *39(3)*, 13-18.

[9] Huang, N., Jai, G., & Chao, H. (2008). Early identifying application traffic with application characteristics. *Proceedings of IEEE International Conference on Communications* (pp. 5788-5792). Beijing, China.

[10] Huang, N., Jai, G., Chao, H., *et al.* (2013). Application traffic classification at the early stage by characterizing application rounds. *Information Sciences*, *232(20)*, 130-142.

[11] Hullár, B., Laki, S., & Gyorgy, A. (2011). Early identification of peer-to-peer traffic. *Proceedings of IEEE International Conference on Communications* (pp. 1-6). Kyoto, Japan.

[12] Dainotti, A., Pescapé, A., & Sansone, C. (2011). Early classification of network traffic through multi-classification. *Lecture Notes on Computer Science*, *6613*, 122-135.

[13] Nguyen, T. T. T., Armitage. G., Branch, P., *et al.* (2012). Timely and continuous machine-learning-based classification for interactive IP traffic. *IEEE/ACM Transactions on Networking*, *20(6)*, 1880-1894.

[14] Rizzi, A., Colabrese, S., & Baiocchi, A. (2013). Low complexity, high performance neuro-fuzzy system for Internet traffic flows early classification. *Proceedings of International Wireless Communications and Mobile Computing Conference* (77-82). Sardinia, Italy.

[15] Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks*, *3(1)*, 109-118.

[16] Mishra, S., Bhende, C. N., & Panigrahi, B. K. (2008). Detection and classification of power quality

disturbances using S-transform and probabilistic neural network. *IEEE Transactions on Power Delivery*, *23(1)*, 280-287.

[17] Song, T., Jamshidi, M. M., Lee, R. R., & Huang, M. (2007). A modified probabilistic neural network for partial volume segmentation in brain MR image. *IEEE Transactions on Neural Networks*, *18(5)*, 1424-1432.

[18] Rutkowski, L. (2004). Adaptive probabilistic neural networks for pattern classification in time-varying environment. *IEEE Transactions on Neural Networks*, *15(4)*, 811-827.

[19] Parzen, E. M. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, *33*, 1065-1076.

[20] Waikato Internet Traffic Storage (WITS). From http://www.wand.net.nz/wits

[21] UNIBS: Data sharing. From http://www.ing.unibs.it/ntw/tools/traces/

[22] Gringoli, F., Salgarelli, L., Dusi, M., *et al.* (2009). Gt: Picking up the truth from the ground for internet traffic. *ACM SIGCOMM Computer Communication Review, 39(5)*, 12-18.

[23] Tcpdump/Libpcap. From http://www.tcpdump.org

[24] Peng, L., Zhang, H., Yang, B., *et al.* (2014). Traffic labeller: Collecting internet traffic samples with accurate application information. *China Communications*, *11(1)*, 82-91.

[25] Peng, L., Yang, B., Chen, Y., *et al.* (2014). How many packets are most effective for early stage traffic identification: An experimental study. *China Communications*, *11(9)*, 206-216.

[26] Weka 3: Data mining software in Java. From http://www.,cs.,waikato.,ac.,nz/ml/weka/

[27] Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*, 1145-1159.

Lizhi Peng was born in 1975. He received the bachelor degree in engineering from Xi'an Jiaotong University in 1998, and received the master degree of engineering from University of Jinan in 2006. He is currently working toward the Ph.D. degree in the School of Computer Science and Technology, Harbin Institute of Technology.

He is currently an associate professor of School of Information Science and Engineering, University of Jinan. His researching interesting includes network information processing, machine learning and intelligent computing and parallel computing.