IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 18, 2019, accepted March 6, 2019, date of publication March 11, 2019, date of current version April 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2904039

Common Knowledge Based and One-Shot Learning Enabled Multi-Task Traffic Classification

HAIFENG SUN[1], YUNMING XIAO[1], JING WANG[1], JINGYU WANG[1], QI QI[1], JIANXIN LIAO[1], AND XIULEI LIU[2]

[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Beijing Advanced Innovation Center for Materials Genome Engineering, Beijing Information Science and Technology University, Beijing 100083, China

Corresponding author: Haifeng Sun (hfsun@bupt.edu.cn)

This work was supported in part by the National Key R&D Program of China under Grant 2016YFC0801407, in part by the National Natural Science Foundation of China under Grant 61771068, Grant 61671079, Grant 61471063, Grant 61421061, and Grant 61601039, and in part by the Beijing Municipal Natural Science Foundation under Grant 4152039.

ABSTRACT Deep neural networks have been used for traffic classifications and promising results have been obtained. However, most of the previous work confined to one specific task of the classification, where restricts the classifier potential performance and application areas. The traffic flow can be labeled from a different perspective which might help to improve the accuracy of classifier by exploring more meaningful latent features. In addition, deep neural network (DNN)-based model is hard to adapt the changes in new classification demand, because of training such a new model costing not only many computing resources but also lots of labeled data. For this purpose, we proposed a multi-output DNN model simultaneously learning multi-task traffic classifications. In this model, the common knowledge of traffic is exploited by the synergy among the tasks and improves the performance of each task separately. Also, it is showed that this structure shares the potential of meeting new demands in the future and meanwhile being able to achieve the classification with advanced speed and fair accuracy. One-shot learning, which refers to the learning process with scarce data, is also explored and our approach shows notable performance.

INDEX TERMS Traffic classification, multi-task model, transfer learning, one-shot learning.

I. INTRODUCTION

As the promotion of the hardware capacity and the growing demands in communications, the simple systems of the traditional internet network are gradually replaced with some new management systems, e.g. the SDN (Software-Defined Network) [1]. A rising computation is implemented in the network. Hence, the network traffic classification has become one of the key points to the efficient network traffic management and QoS (Quality of Service) guarantees [2].

For past several decades, many theories were proposed to better predict the diverse requirements for bandwidth, latency, and quality of different flows. It is widely recognized that many properties of a flow can be well predicted with a small set of the flow features. Among all the approaches, some machine learning tools have progressed promising results using the port number or some statistical features of the flows [3]. Recently, some studies have turned to the deep learning techniques and pleasing scores of accuracy have been obtained [4]. However, in many studies, different flow classification tasks were usually regarded as different areas and most of the studies were focusing on one specific area.

Our work was motivated by the fact that the demands will always be changing and that most of the previous works have shown that some features can be used to discriminate different tasks of flows. For instance, the features for classifying either elephant or mice flows and that for identifying the applications of flow, are surprisingly similar [5]–[7]. This alludes to that there might be some common knowledge for different traffic classification tasks and the mapping functions from the selected features to the different flow properties that we need to predict are of close relations. Based on the common knowledge, we would like to explore an all-in-one solution, a tool that is supposed to adapt to continuous changes of demands and a tool which has the potential of being applicable for various related tasks of flow classification.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhanyu Ma.

To address above issues, we compared many machine learning tools and selected the DNN as the basic technique. In order to extract and implement the common knowledge to the training, we propose the Multi-Output DNN structure which is a reform of the simple DNN. Our proposed structure has two phases: the common knowledge is extracted at the first phase by the common layers, and then, the various flow classification tasks are able to be trained and computed at the same time with inter-independent branches of private layers respectively. The common layers are also supposed to be effective for saving the calculation time and memory. Each of the classification tasks owns an extended and relative simple branch of private layers for task specialization. Besides, with this proposal, we suppose that new related classification tasks might be able to be solved by adding a new branch of private layers connecting to the well-trained common layers.

In the real-world implementations, adopting techniques like the DNN are quite hard due to the great computation complexity and time cost of the neural networks. But our proposed structure might be a solution. It is supposed to be not only suitable for the centralized data center controllers which are equipped with huge volumes of computational power but be also a potential solution in many scenarios that devices have weak computing power. On the one hand, in the centralized data center, our proposed algorithm is supposed to handle several classification tasks at the same time with one calculation unit and much less cost on either the time or the memory. Moreover, a new demand can be tested without influencing the established business and share the common knowledge extracted from a great amount of previous related data. On the other hand, for the weak end devices, using our hierarchical structure, the common learning layers can be placed remotely in the cloud or in some specialized neural network computation units. And then, only a simple cost of calculation for several neurons can the end devices complete the complex and new tasks by connecting to the common layers.

Our main contributions are summarized as follows:

- We propose the Multi-Output DNN structure as a means to extract the common knowledge of network traffics and hence a solution to the multi-task problems in the traffic classification.
- We demonstrate experimentally that our proposed algorithm could be capable of transfer learning handling different related tasks in traffic classification and thus meets our vision of the potential ability to address new demands.
- In the extensive experiments of scenarios that learning a category from few examples, our proposed algorithm shows notable performances on both the accuracy and efficiency by using the common knowledge from the previous trained common layers.

The rest of the paper is organized as follows. The background of flow classification and related works on multi-task learning are reviewed in Section 2. The basic algorithms and our proposing algorithm are presented in Section 3. Section 4 lists the description and statistics of dataset we have used in the experiments. Section 5 gives our experimental settings, and Section 6 reports the details of our experiments and the results. The paper concludes in Section 7.

II. BACKGROUND AND RELATED WORKS

There has been a lot of approaches to the traffic classification. The machine learning tools have been the most promising techniques and they were well researched over the past several decades. There has been a large number of researchers that focus on predicting different flow features. And the studies on the traffic classification were primarily focused several perspectives.

The detection of potential elephant flows would avoid some potential network congestions [8]. And many works focused on the precise classification of multimedia traffic or P2P (Peer-to-Peer) traffic [9] accounting for their potential large proportion of all the traffic. The discrimination of specific application layer information of the flows is expected to provide better QoS based on different demands of the applications. And the early detection of malicious network traffic is believed to provide an appealing improvement to the network safety [10].

Supervised learning was explored on the one hand. The kNN (k-Nearest Neighbor) is a simple but useful tool [11]. Naive Bayes estimator [12] was introduced, and Auld *et al.* [13] proposed the Bayesian neural network based classification later. SVM (Support Vector Machine) was proven to be one of the most promising techniques in many tasks of classification [14]. C4.5 Decision Tree was another solution [15]. Besides, some researchers have tried the Deep Learning technique [16]. For other areas, deep learning could train a model by an end-to-end fashion, which decrease the suffering of expert modeling [17]. On the other hand, Bernaille *et al.* [18] suggested that an unsupervised learning model could be helpful for traffic classification due to its independence form predefined labels.

Many data mining and machine learning algorithms make predictions using models which are trained with previously collected labeled or unlabeled dataset. Nevertheless, in many real-world applications, the tasks to explore or the distribution of feature spaces are not always constant [19]. In contrast, the transfer learning allows the training and testing datasets to be of different domains, tasks, or distributions [20].

The weakly supervised learning which could use the other related task labels to enhance the target task performance is attracted more and more researchers. One of the main issues is reducing the cross-domain discrepancies. Many weakly supervised learning has been used [21] in other areas, such as computer vision [22] and natural language processing [23]. For natural language processing, Perera *et al.* [24] combines many related tasks, such as semantic annotation and name entity recognition, to learn better sentence representation. For image classification, Han *et al.* [25] proposed a method uses unsupervised learning algorithm to encoder the image background and then separate the salient object from background

by reconstruction residuals. Zhang *et al.* [26] integrate self-paced learning into multi-instance learning for co-Saliency detection. Recently, Sun *et al.* [27] used transfer learning and Moustafa *et al.* [28] used ensemble learning to improve the network traffic classification.

It is noted that the transfer learning has many branches of studies depending on the availability of datasets for source and target tasks. Most of the cases are well explored. However, the multi-task classification was rarely noticed for traffic classification problem. Besides, some studies have focused on the one-shot learning, where the labeled training set in the target task is quite small. The key point of the one-shot learning is that one can take advantage of knowledge from previously trained data. Fei-Fei *et al.* [29] mentioned this word at the first time.

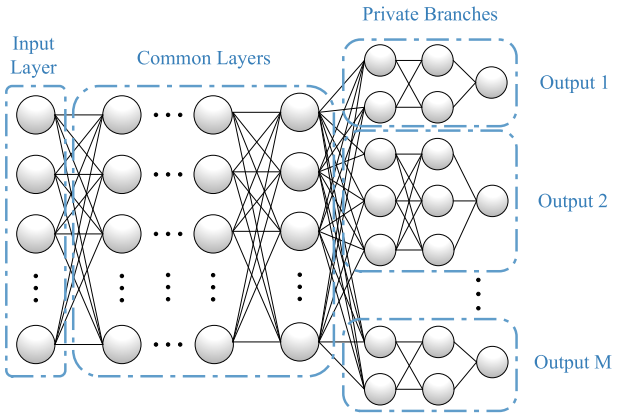

FIGURE 1. The multi-output DNN structure.

III. METHODOLOGY

A. PRELIMINARY

In this section, we introduce some notations and definitions which are used in this paper. At the first place, a *task* $\mathcal{D}$ is a set of two components: the feature space $\mathcal{X}$ and the corresponding label space $\mathcal{Y}$. Given a task dataset D which consists the features $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$ and the corresponding labels $Y = \{y_1, \ldots, y_n\} \in \mathcal{Y}$, the relation of them can be denoted as a mapping function $f : X \rightarrow Y$. For instance, in classifying the flow rate, the port number and a set of statistics of a flow is a feature vector X, and their real flow rate can be an instance of the label space as $Y \in \mathcal{Y}$. From the perspective of discriminative model, the $f(\cdot)$ can be written as $P(y|x)$. The deep learning is a process of fitting a function which is somehow likely to the *real* function. And it usually uses a loss function $L[f(X), Y]$, which presents the difference between the prediction and the ground truth, to judge the performance of the fitting function.

B. MULTI-OUTPUT DNN APPROACH

The mathematical intuition of DNN is a highly nonlinear mapping function. Given a specific trace dataset D, now consider that we want to classify the duration, flow rate and many other flow properties which are denoted as $Y_1, Y_2, \ldots, Y_n$, the corresponding mapping function $f_1, f_2, \ldots, f_n$ may be similar in some degree with the same X. With an allowable affordability deviation Δ_i, we can have that

$$f_i = f_i' \cdot f_c + \Delta_i. \tag{1}$$

The f_i' part refers to the mapping function of different branch of private layers respectively and f_c denotes the general function of the common layers. The common knowledge C in this structure can be expressed as

$$C = f_c(X, \theta_c), \tag{2}$$

where the θ_c denotes the parameters in the common layers. It is shown in Fig. 1 that a Multi-Output DNN contains common layers and several branches of private layers. In the structure of the neural network, all the common layers are fully connected, and the first layers of all private branches are also fully connected to the last common layer respectively. For each private branch, layers are fully connected to each other. But there are no connections between any pair of neurons from different private branches.

Assuming that there are totally N tasks of classification, for each task, i.e. each branch of the neural network, we use a loss function $L_i[f_i(X, \theta), Y_i]$ to measure the performance of the corresponding fitting function, where θ denotes the parameters in the neural networks. Hence, the general optimization process is shown as follows:

$$\begin{aligned} \theta^* &= \underset{\theta=\{\theta_c,\theta_i\}}{arg\,min} \sum_{i}^{N} L_i[f_i(X, \theta), Y_i] \\ &= \underset{\theta=\{\theta_c,\theta_i\}}{arg\,min} \sum_{i}^{N} L_i[(f_i' \cdot f_c)(X, \theta), Y_i]. \end{aligned} \tag{3}$$

The θ^* denotes the parameters of final fitting function. As the deviation Δ_i is beyond control, we ignore it here and later this section in the equations for simplicity. And the θ_c and θ_i refers to the parameters in the common layers and i^{th} to the i^{th} branch of private layers respectively.

The training process of Multi-Output DNN is then slightly different to the simple DNN. A direct method to the Eq. 3 is the RRBP (Round-Robin Backpropagation), which is illustrated as Algorithm 1. In RRBP, the loss function of each branch is calculated respectively, and for every batch of inputs, each task complete training its private part and the common part in turn. And the other is PPSCBP (Parallel-Private and Sum-Common Backpropagation). In contrast, PPSCBP defines a general loss function

$$L = \sum_{i=1}^{N} w_i L_i, \tag{4}$$

where L_i refers to the loss function of i^{th} private branch and w_i is the weight of the branch. Then, the corresponding

optimization process is

$$\theta^* = \underset{\theta=\{\theta_c,\sum_i^N \theta_i\}}{arg\,min} \; L[\sum_i^N w_i f_i(X,\theta), \sum_i^N w_i Y_i]. \tag{5}$$

For every batch training, the general loss is calculated, and all the private and common layers are updated. Algorithm 1 and 2 demonstrates it specifically.

Algorithm 1 RRBP

Input: The training data set X and labels $Y_1, \ldots, Y_N$
The parameters $\theta_c, \theta_i, \ldots, \theta_N$
the lost function $L_1, \ldots, L_N$ and maximum loss thread ϵ
Output: θ^*

```
Pre-training for the θ_c
while Σ_i^N L_i(f(X), Y_i) > ε do
    for i := 1 to N do
        θ_i := θ_i − ∂/∂X L_i[f(X), Y_i]
        θ_c := θ_c − ∂θ_i/∂L_i ∂/∂X L_i[f(X), Y_i]
    end for
end while
```

Algorithm 2 PPSCBP

Input: The training data set X and labels $Y_1, \ldots, Y_N$
The parameters $\theta_c, \theta_i, \ldots, \theta_N$, the weights $w_1, \ldots, w_N$
the lost function $L_1, \ldots, L_N$ and maximum loss thread ϵ
Output: θ^*

```
Pre-training for the θ_c
while Σ_i^N L_i(f(X), Y_i) > ε do
    L := 0
    for i := 1 to N do
        θ_i := θ_i − ∂/∂X L_i[f(X), Y_i]
        L := L + w_i · ∂/∂X L_i[f(X), Y_i]
    end for
    θ_c := θ_c − L Σ_i^N ∂θ_c/∂θ_i
end while
```

For either the training function, a presumption behind this proposal is that training several related classes at a time would update the parameters in the neural network from different directions in the multidimensional space from the perspective of mathematics. Hence, it might be helpful for the learning not dropping into local optimums.

C. TRANSFER LEARNING

In some cases, a new demand, e.g. a new label classification task, is proposed, but it might be costly to train a new model for solving the problem. By the conventional machine learning methods, there might be faced with difficulties collecting the sufficient data for the target task while abundant data for related tasks were available. We may find it practical to train the new classifier using the data from related tasks or using other tasks' previously trained layers to speed up the training process. Such a new task is called the *target task* $\mathcal{D}_\mathcal{T}$ and the previous trained tasks are referred to the *source task* $\mathcal{D}_\mathcal{S}$.

The structure we proposed is satisfying dealing with the meeting new demands of a new flow labeling. Typically, given the condition that

$$\min_{\theta=\{\theta_c,\theta_S\}} \sum_{i\in S} L_i[(f_i' \cdot f_c)(X,\theta), Y_i], \tag{6}$$

where the S is the set of source tasks, one training process can be expressed as follows:

$$\theta^* = \underset{\theta=\{\theta_c,\theta_S,\theta_T\}}{\arg\min} \{\sum_{i\in S} L_i[(f_i' \cdot f_c)(X,\theta), Y_i] + \sum_{j\in T} L_j[(f_j' \cdot f_c)(X,\theta), Y_j]\}. \tag{7}$$

T denotes the set of target tasks. In other words, the common layers should be trained and parameters in both the source tasks and the target tasks need to be optimized using backpropagation. The Eq. 7 can also be rewritten similar to Eq. 5 by Eq. 3. We would refer to this scheme using the abbreviation SCT (to train Source, Common and Target layers) in later sections.

One way to simplify the above algorithm is to subtract the calculation of previous trained tasks:

$$\theta^* = \underset{\theta=\{\theta_c,\theta_T\}}{\arg\min} \sum_{j\in T} L_j[(f_j' \cdot f_c)(X,\theta), Y_j] \tag{8}$$

In the implementation of OCT (to train Only Common and Target layers), one of the ways is to connect the target tasks directly to the original common layers. But this behavior might cause some negative effects to the source tasks. Another way is to sacrefice the advance in saving the memories – to copy all the parameters in the trained common layers and build a new neural network with them. And train the new neural network as usual.

Pursuing a more clear and simple function, the last method is to ignore the common layers.

$$\theta^* = \underset{\theta_T}{\arg\min} \sum_{j\in T} L_j\{f_j'[f_c(X,\theta_c), Y_j], \theta_j\} \tag{9}$$

As Eq. 9 illustrates, this training scheme adds the new task to the trained neural network but trains only the private layers of the new branch. In such an approach, the common layers and the previous private layers will not be influenced by OT (training Only Target layers). It meets the demand of testing a new task without influencing the established business. General, the common layers learnt by source tasks would like to be biased towards the source tasks. If the source tasks are diverse enough, the common layers try to learn a neutral traffic representation [30]. The Fig. 2 illustrates a simplified version of the three training schemes.

D. ONE-SHOT LEARNING

Most previous data mining techniques require a large amount of data. However, it is often difficult to acquire large sets of training examples. The one-shot learning refers to predicting with a very small labeled training set [31]. In the field of flow

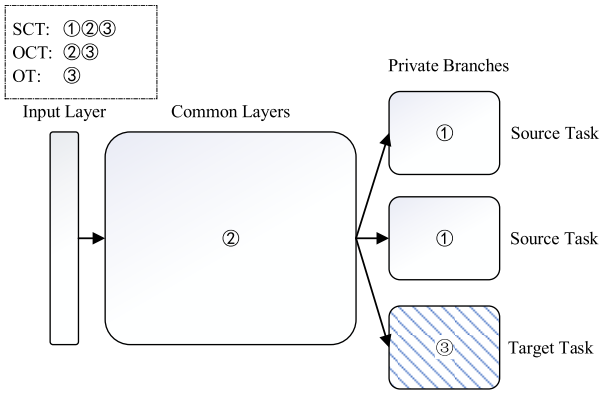

FIGURE 2. Transfer training schemes. SCT refers to train the source, common and target layers; OCT refers to train only the common and target layres whereas OT refers to train only the target layers.

classification, the dataset of some label tasks, e.g. the elephant or mice flow, are quite plentiful while requiring others, e.g. information of application layer, are expensive or rarely dated due to historical miss or legal reasons.

Hence, under the condition of Eq. 6, a new task $\mathcal{D}_{\mathcal{T}}$ is introduced with a dataset $D_o = (X_o, Y_o)$, where $X_o \subsetneqq X$ and $|X_o| \ll |X|$. The training schemes here is quite similar to the Eq. 7, 8 and 9. The only difference is to replace original X and Y with X_o and Y_o respectively.

In other words, a training set with plenty data is adopted for the training of source tasks in the first place. After the convergence of them, a relative small data set is adpoted for the training of target task. Also, three training schemes the same as the one in the last section are proposed.

IV. DATASETS AND FEATURE SELECTION

We selected several real-time network traffic traces dataset to evaluate our algorithms. Some of the important features of the traces are listed in Table 1.

- WITS: ISPDSL-I and ISPDSL-II was entirely contiguous packet header traces captured from a New Zealand ISP using a single DAG 3.7G card by WITS [32]. In our study, we selected 3 trace files respectively. The trace contains all the traffics for the ISP's customers in both directions. The packets were truncated four bytes after the end of the transport header and thus available for acquiring layer 7 information. We collected all the flows with layer 7 information and with features that there were totally more than 5 packets.
- Moore and Zuev [12] collected several sets of flows using the high-performance network monitor. There are totally 10 sets of data publicly available. We selected the Entry09 and Entry10 in our experiments. For each of the flow record, there are 249 features included containing almost all the statistics that a flow can be characterized by and the information is provides based on both directions and each direction individually.

The selection of features to classify different flows has been well discussed [33]. In this paper, the feature extraction

TABLE 1. Statistics of the dataset.

Trace	Start Time	Duration	Flow	App Num
ISPDSL-I	10:54, Jan 6, 2009	1h 36min	23545	36
ISPDSL-II	16:09, Jan 6, 2010	1h 21min	23892	50
Entry09	13:45, Aug 20, 2003	27min	66248	11
Entry10	14:55, Aug 20, 2003	27min	65036	11

is not a major topic of what we're exploring. And due to the discriminative ability of the supervised learning, we would select common and comprehensive features as input to provide sufficient statistics for different tasks of classifications. Thus, we concluded from the previous studies [5]–[7] and decided to implement totally 16 features promising the unbias from feature selection (which has been evaluated by matrix factorization methods, such as variational bayesian matrix factorization [34]): transport layer source port, destination port, number of packets with PUSH flag, ratio of upload and download, the first quartile of inter-arrival time and packet size, the statistical properties of the inter-arrival times and packet sizes (the minimum, the maximum, the mean, the variance and the rooted mean square).

In later experiments, three tasks of classification are discussed: the duration, the flow rate, and the application type. For the first two tasks, we divide them into two sets with the median values and label them respectively.

V. EXPERIMENTAL SETTINGS

A. EXPERIMENTAL ENVIRONMENTS

A series of experiments were designed to evaluate our new traffic classifier using a Multi-Output DNN technique. In these experiments, a typical 3-branch multi-output neural network is implemented, and many different situations are discussed. In the evaluation, we adopt the cross entropy for all the loss function of DNN. The activation function is the sigmoid function. And the batch normalization is implemented in constructing the neural networks in later experiments for improving the accuracy. The common layers consist of four fully-connected layers with 30 neurons separately. In each private branch, two fully-connected layers are implemented (The number of first layer neurons is 30 neurons, the number of output layer neurons equal to the classification numbers).

The platform used for this experiment is: Windows 10 operating system running on a laptop equipped with an Intel 2.4GHz i5-6300U CPU, 8GB RAM, and a custom variant of NVIDIA GeForce GPU with 1GB memory. Most of the algorithm implementations in later experiments are completed with the skikit-learn tools [35].

B. EVALUATION

The performance of the classifier can be measured by the accuracy and the time cost.

A common way to illustrate a classifier's OA (overall accuracy) is through the well-known metrics of TP, TN, FP and FN (T and F for True or False, and P and N for

TABLE 2. Traditional techniques and multi-task learning experiments.

Domain	Dataset	KNN	SVM	Decision Tree	Random Forest	Linear Reg	DNN	Maxent	Ensemble	RRBP	PPSCBP
Duration	Trace I	92.70%	94.64%	94.08%	**96.32%**	60.57%	96.08%	94.21%	96.27%	96.09%	96.23%
	Trace II	92.44%	95.52%	94.16%	95.58%	58.70%	95.82%	95.51%	95.80%	**95.89%**	95.71%
	Entry09	86.86%	88.16%	57.37%	83.54%	22.95%	94.90%	89.51%	94.41%	**95.15%**	95.06%
	Entry10	87.42%	86.26%	42.72%	85.67%	21.89%	94.16%	93.74%	94.60%	**94.72%**	94.68%
Flow Rate	Trace I	93.61%	95.68%	98.45%	97.76%	67.15%	97.71%	97.37%	**98.51%**	98.05%	**98.09%**
	Trace II	92.46%	95.95%	96.01%	95.88%	65.22%	**98.17%**	96.87%	98.10%	97.81%	97.93%
	Entry09	90.10%	90.11%	91.93%	93.26%	20.11%	98.39%	96.51%	98.40%	98.38%	**98.49%**
	Entry10	90.48%	88.02%	92.88%	91.19%	21.51%	97.77%	95.08%	97.71%	**98.38%**	98.02%
Application	Trace I	96.36%	96.13%	94.38%	96.50%	49.99%	**97.09%**	95.41%	97.07%	96.54%	96.91%
	Trace II	96.50%	96.67%	87.14%	91.05%	52.99%	**97.26%**	96.17%	97.21%	96.16%	96.98%
	Entry09	97.96%	96.80%	85.88%	85.67%	37.46%	98.36%	91.50%	98.80%	98.20%	**98.45%**
	Entry10	97.96%	94.00%	88.65%	95.68%	36.60%	98.67%	89.97%	**98.71%**	98.54%	98.60%

(Maxent is transfer learning based on [27], Ensemble is ensemble learning based on [28])

Positive or Negative respectively).

$$OA = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FP_i)}, \tag{10}$$

where n denotes the number of categories. The OA for different classification function were discussed respectively in later experiments.

To evaluate the time cost, we record the training time for each training epoch. As concerned as the DNN, the criterion for the convergence of a training process is thought to be that there's no more than 0.5 percent of increment of the accuracy in 100 later training epochs over test dataset. Then, the accuracy and the time cost are recorded and considered as final results. For a Multi-Output DNN, the time cost is set to equal to the one task which spends the most time to converge.

As for the analysis of performance of the knowledge space representing the traffics, the Euclidean distance is adopted in later experiments. Considering in a multi-dimensional mathematical space, using d_{ij} to denote the normalized distances to the geometrical center of label j for instances with label i, the perplexity is considered to be

$$perplexity = \frac{\sum_{i}^{N} d_{ii}}{\sum_{i}^{N}\sum_{j}^{N} d_{ij} - \sum_{i}^{N} d_{ii}}. \tag{11}$$

The less the perplexity is, the better the knowledge space represents the traffics.

Moreover, we split each the dataset randomly and then adopted the hold-out validation.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

A. THE MULTI-OUTPUT DNN

We firstly experiment to check if our proposed algorithm is feasible to the multi-task traffic classification. Three tasks – duration, flow rate, and flow application classification – are discussed in the following discussion. We implemented five traditional machine learning techniques and three recently proposed techiniques: the kNN (k-Nearest Neighbor), the SVM (Support Vector Machine), Decision Trees, Linear Regression, the Random Forest, Deep Neural Networks, Maxent [27], Ensenble Learning [28]. Also, we normalize the input data before training. We constructed three different single DNNs respectively, each with 4 layers and 30 neurons for each layer.

Then, we continue to the Multi-Output DNN. In order to avoid the question that the private layers play too many roles and thus eliminates the validity of the common layers, we adopt a very simple structure for each private branch with only one hidden layer with 30 neurons and one output layer. Given that these classifications are not very complex problems, we construct the Multi-Output DNN with 4 common layers and 30 neurons for each layer. We tested 2 different training techniques of the Multi-Output DNN that we have discussed in Section III. One is the RRBP (round-robin backpropagation), and the other is PPSCBP (Parallel-Private and Sum-Common Backpropagation).

Table 2 illustrates the result of the overall accuracies for three tasks of classification with above tools and the 2 schemes for the Multi-Output DNN in four trace datasets. From the results, we can observe that the KNN is mostly a good tool for the application classification. But it doesn't share fair results in other two tasks. The SVM performs fairly in Trace I and Trace II, but it seems to fail in the Entry09 and Entry10. Decision Tree shows a remarkable score of accuracy on the flow rate on Trace II, though its results don't seem well in other experiments. It seems that the simple Linear Regression is not suitable for this problem. The Random Forest algorithm is not as stable as the DNN, especially in the experiments using Entry09 and Entry10. The ensemble learning has stable performance in all the experiment and slightly worse than the proposed algorithm.

As far as the DNN is concerned, it keeps very stable performances on all three tasks of different traces. Besides, the DNN presents either the best or the second-best results in all the experiments. As the second player on the classification of duration for Trace I, the difference between the DNN and the best one is no more than 0.2 percent. Therefore,

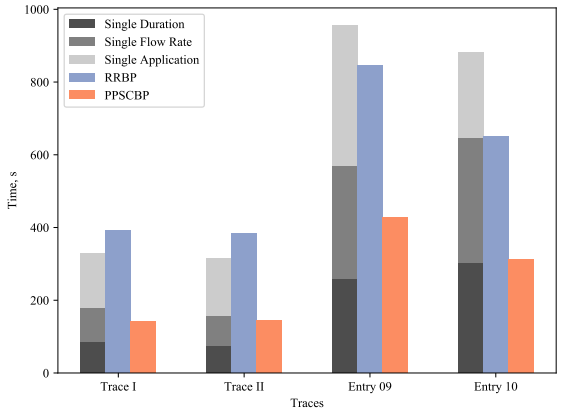

FIGURE 3. Multi-output DNN and single DNN comparison.

the DNN advances in traffic classifications compared with other machine learning techniques.

As for the Multi-Output DNN, from the last two columns in Table 2, it can be concluded that the accuracies of both 2 training schemes are approximately equal to that of the single structure. The PPSCBP has a slightly better accuracy than RRBP in most cases and ties the single DNN generally. The differences are mostly within the limit of the biases of datasets. Therefore, the Multi-Output DNN should share the same behavior on traffic classification as the single DNN. Besides, its performance stabilization on different tasks and traces shows the potential ability to handle multi-class problems and thus meets some of our proposals properly.

Now considering the time cost shown in Fig. 3, The training time for variant tasks of different datasets are varying a lot. Generally, the RRBP costs a little more than 3 times as the average standard of simple DNNs on all three tasks in Trace I and Trace II but advances them in Entry09 and Entry10. PPSCBP costs 1 to 1.5 much time as the average one of single structures. Considering that the Multi-Output DNN outputs three results together, we would sum up the time cost of single DNN over three tasks. Hence, the RRBP training scheme shares approximately equal time cost with the sum of single structures. On the other hand, PPSCBP has advantaged than the simple structure in the convergence speed. Moreover, specifically in this experiment, the Multi-Output DNN with PPSCBP can be more than 2 to 3 times faster while saving less neurons' space. And predictably, the more tasks the DNN trains, our proposal structure with PPSCBP training scheme would save more time and memory. In later experiments, we would choose the PPSCBP to train the other Multi-Output DNNs in later experiments.

B. COMMON KNOWLEDGE

Then, we would discuss the common knowledge extraction with the Multi-Output DNN. After training the three tasks using Multi-Output DNN, the values of the last common layer in the Multi-Output DNN are recorded. Similarly, we record the values of the second-last layer and the corresponding

TABLE 3. Perplexity metrics of the common knowledge.

Task \ Technique	Multi	Duration	Flow Rate	App
Duration	**0.7440**	0.7623	0.7677	0.9618
Flow Rate	**0.7972**	0.8486	0.8379	0.9603
App	**0.0672**	0.0783	0.0783	0.0684

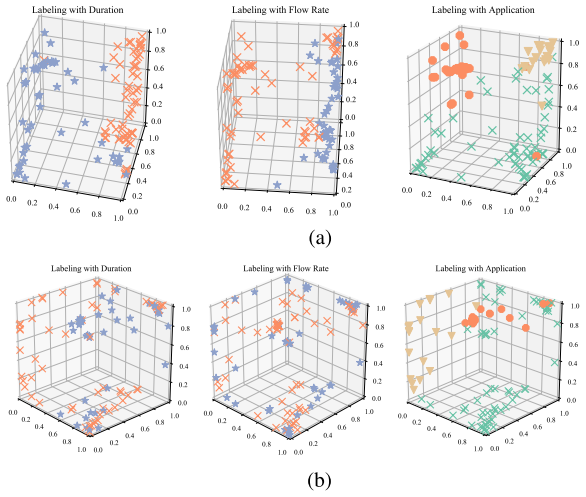

FIGURE 4. The knowledge space of multi-task and single-task training. Different marks and colors refer to different labels. (a) The extracted common knowledge for different labeling. (b) The knowledge spaceöf single-task application.

labels for simple DNNs as well (The values of DNN is not the de facto common knowledge but just is considered as control groups).

Table 3 demonstrates the perplexities of the different tasks represented by the common knowledge. It can be read that the perplexities of common knowledge extracted by Multi-Output DNN are the least in all three tasks. Then, each of the single DNN has the second-least perplexity for the corresponding training task but shares relatively higher ones for the other tasks. In addition, it can be drawn that the tasks of duration and flow rate are more similar while that of application is some more different. It is shown that our proposed algorithm is able to extract the common knowledge of the network traffics properly. In this experiment, the common knowledge represents the flows even better than the corresponding knowledge space of single task training.

We go through the result spaces for all neurons and select the three of them as delegation. It is illustrated in 3-dimensional views labeling with three tasks as Fig. 4. The Fig. 4a shows that the values of different labels are properly represented and clustered separately after some training epochs. As a contrast, Fig. 4b demonstrates the knowledge space in a single-task training scenario, where the DNN can extract the knowledge labeling with application types. However, it behaves barely satisfactorily in the other two tasks.

TABLE 4. Transfer learning experiments.

		Single DNN		SCT		OCT		OT	
Source v.s. Target Task	Dataset	Accuracy	Time, s	Accuracy	Time, s	Accuracy	Time, s	Accuracy	Time, s
flow rate, app v.s. duration	Trace I	96.08%	85	96.36%	67	**96.52%**	**23**	95.21%	50
	Trace II	95.82%	65	95.93%	54	**96.18%**	46	95.28%	**37**
	Entry09	94.90%	260	94.72%	157	**95.73%**	142	94.35%	**104**
	Entry10	94.16%	304	94.62%	177	**94.97%**	127	94.03%	**81**
duration, app v.s. flow rate	Trace I	97.71%	94	**97.75%**	95	97.74%	81	94.91%	**51**
	Trace II	**98.17%**	73	97.76%	54	98.06%	41	96.07%	**38**
	Entry09	98.39%	311	98.30%	229	**98.62%**	145	97.54%	**36**
	Entry10	97.77%	344	97.54%	260	**97.94%**	187	97.05%	**78**
duration, flow rate v.s. app	Trace I	**97.09%**	151	97.03%	120	96.94%	91	95.17%	**54**
	Trace II	97.26%	157	97.05%	132	**97.45%**	123	94.88%	**87**
	Entry09	98.36%	385	98.39%	295	**98.46%**	215	97.34%	**138**
	Entry10	98.67%	235	98.68%	198	**98.80%**	135	97.62%	**133**

C. TRANSFER LEARNING

The structure of the Multi-Output DNN brings remarkable benefits to eliminating the time and memory cost. Still, there're more to explore. In many cases, the demands in a field are not always constant. In the traffic classification, there are often several tasks to discuss. We've applied three types of classical questions in the previous study. But it is unknown that whether the Multi-Output DNN would be able to handle some new demands. Due to the limit of the dataset, it is hard to set new task labels. So, we adopted the 2-phase strategy: firstly, we trained two tasks as source tasks, and then, the third task was raised as target task and be added into the established neural network. We hence evaluate the effectiveness and the accuracy of the target task.

More specifically, in the first place, we apply a Multi-Output DNN with 4 common layers and 2-layer private branch for two source task classifications. After 200 training epochs, the accuracies of the two tasks on the test dataset were supposed to be stable enough. Then, we added the private branch of the target task and adopted three types of methods mentioned in Section 3 respectively. As a standard control, we chose a simple neural network with the same number of architecture, i.e. the same number of layers and neurons, and applied the same learning rate. Table 4 illustrates the results of three approaches and the standard control experiment.

In the results, we can observe that the SCT shares approximately equal accuracies to the single DNN with a less time cost over all the tests. This phenomenon would lead to that the common knowledge extracted by previously trained common layers would speed up the training process of a new task with the fair results. Besides, this scheme requires much less memory compared to constructing a new DNN.

Now considering the OCT, it achieves slightly better accuracies than the SCT while taking even less time as for all the tasks and traces. Generally, the single DNN costs as 2 to 4 times as the OCT to convergence. This is a remarkable result. Still, considering its disadvantage to the SCT, it would be a compromise problem.

Finally, the OT takes the least time to converge in most cases. Though its achieved accuracies are not as good as the other 2 training schemes and the single structure, this scheme is equipped some special feature that allows testing a new classification without influencing current business. It would be functional in the real industrial environments.

To sum up, adopting the Multi-Output DNN benefits the process of training when faced with a new traffic classification demand. The SCT is a good solution and it achieves our initial proposal in this experiment. Ignoring the source tasks, the OCT is evaluated to be much efficient and accurate than the traditional DNN approach and other training schemes. Besides, the OT strategy can be used as a quick testing tool for new demands.

D. ONE-SHOT LEARNING

For a more realistic experiment, the features or labels for some tasks are hard to obtain. Again, we choose the duration, flow rate, and application to be the target task in turn while the other to play the source task. A Multi-Output DNN with the same structure as the last experiment is adopted. In the first place, we use all the data to train both the duration and flow rate tasks and thus try to obtain the common knowledge. Then, 1 percent of total recorded flows were randomly selected for the training of application classification. Three different training schemes are tested and compared with each other. Also, a single DNN with the same structure is constructed as the standard control. Table 4 shows the result of our one-shot experiment.

Under the condition of insufficient data, the DNN performs not very well in some of the cases. Accuracies of 70 to 80 percent are obtained in half of our tests. In the contrast, the average accuracy of the Multi-Output DNN is around 94 percent whichever the training scheme is used. More specifically, the SCT performs quite well in the experiment that it not only shows faster speed to converge but also shares higher accuracies than the standard control in most cases. It is noted that a 15 percent difference in the accuracy compared to the

TABLE 5. The one-shot learning experiments (one percent training set).

		Single DNN		SCT		OCT		OT	
Source v.s. Target Domain	Dataset	Accuracy	Time, s	Accuracy	Time, s	Accuracy	Time, s	Accuracy	Time, s
flow rate, app v.s. duration	Trace I	87.44%	5.1	88.50%	**1.8**	**90.75%**	3.8	88.90%	3.9
	Trace II	90.03%	3.6	**92.76%**	2.7	92.07%	1.8	88.65%	**0.9**
	Entry09	85.73%	13.8	90.75%	5.0	**92.22%**	3.0	90.78%	**2.5**
	Entry10	75.62%	2.1	90.37%	1.5	**90.40%**	1.8	89.65%	**1.4**
duration, app v.s. flow rate	Trace I	87.61%	2.1	90.95%	2.0	**91.21%**	2.7	90.87%	**1.6**
	Trace II	91.39%	4.5	92.84%	4.1	**93.56%**	4.8	92.47%	**3.0**
	Entry09	79.92%	11.4	95.75%	6.1	**96.02%**	3.1	95.48%	**1.5**
	Entry10	79.83%	15.2	92.63%	3.4	94.92%	**2.2**	**95.64%**	3.5
duration, flow rate v.s. app	Trace I	86.81%	5.0	**90.43%**	7.5	89.16%	3.4	89.17%	**2.9**
	Trace II	90.17%	10.8	**91.02%**	10.5	90.49%	9.5	89.29%	**3.1**
	Entry09	95.21%	9.6	**95.93%**	12.2	95.86%	**9.4**	95.66%	10.6
	Entry10	89.29%	14.4	92.10%	**5.1**	**93.76%**	6.8	92.09%	5.4

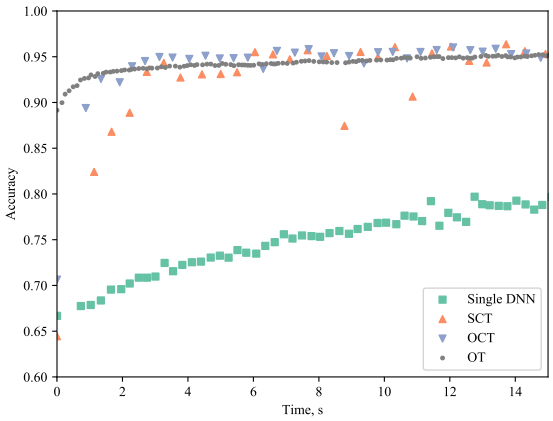

FIGURE 5. An one-shot learning training example.

single DNN is obtained in some of the experiments. As for the OCT, similar to the transfer learning experiments, it exceeds the SCT on both the accuracies and saving times. An average 1 to 2 percent of accuracy promotion is obtained for the OCT even to the SCT. As far as the OT, though its accuracy is slightly lower than other two schemes, it outweighs the single DNN in a big advantage. And it converges much faster than the others. Predicting flow rate as the target task and using the Entry09, Fig. 5 illustrates a typical case of the training process for different algorithms.

Generally, the average accuracy promotion for the Multi-Output DNN to the single DNN is around 5.4 percent and the average time is reduced to around 36 percent using our proposed structure. It is considered that the common knowledge from the trained common layers improves the general performances of the training of target task in the one-shot learning experiments. In such scenarios, our proposed structure has apparently advanced the simple DNN quite a lot.

VII. CONCLUSION

In this paper, we proposed a novel traffic classification structure based on the DNN classifier. Firstly, we evaluated the effectiveness and the validation of the Multi-Output DNN within the field of traffic classification. We adopted three regular tasks: predicting the time duration, the flow rate, and the application type of flows. The PPSCBP training scheme is proved to be effective in both reducing time cost and saving memory compared to the simple DNN structure and it shows a higher overall accuracy than other machine learning tools. Then we have explored the performance of common knowledge extraction. It is shown that our proposed structure can extract the common knowledge of traffics properly and the knowledge space is less perplexed than all the corresponding single-task DNNs. Moreover, we evaluate the potential of our proposed algorithm to address new demands. All the three training schemes have achieved addressing the demands of new classification on related tasks at a faster speed in our experiments. Besides, in one-shot learning scenes, the Multi-Output DNN shows remarkable results that it can be applied to acquire an even better accuracy result and a faster training speed with the support of common knowledge.

REFERENCES

[1] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: An intellectual history of programmable networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 87–98, Apr. 2014. doi: 10.1145/2602204.2602219.

[2] A. Anand and G. de Veciana, "Invited paper: Context-aware schedulers: Realizing quality of service/experience trade-offs for heterogeneous traffic mixes," in *Proc. WiOpt*, Tempe, AZ, USA, May 2016, pp. 1–8. doi: 10.1109/WIOPT.2016.7492916.

[3] T. T. T. Nguyen and G. Armitage, "A survey of techniques for Internet traffic classification using machine learning," *IEEE Commun. Surveys Tuts.*, vol. 10, no. 4, pp. 56–76, 4th Quart., 2008. doi: 10.1109/SURV.2008.080406.

[4] K. Behrendt, L. Novak, and R. Botros, "A deep learning approach to traffic lights: Detection, tracking, and classification," in *Proc. ICRA*, Singapore, May/Jun. 2017, pp. 1370–1377. doi: 10.1109/ICRA.2017.7989163.

[5] R. C. Jaiswal and S. D. Lokhande, "Machine learning based Internet traffic recognition with statistical approach," in *Proc. Annu. IEEE India Conf. (INDICON)*, Dec. 2013, pp. 1–6.

[6] S. S. L. Pereira, J. L. E. de Castro e Silva, and J. E. B. Maia, "NTCS: A real time flow-based network traffic classification system," in *Proc. CNSM*, Rio de Janeiro, Brazil, Nov. 2014, pp. 368–371. doi: 10.1109/CNSM.2014.7014196.

[7] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, "Robust network traffic classification," *IEEE/ACM Trans. Netw.*, vol. 23, no. 3, pp. 1257–1270, Aug. 2015. doi: 10.1109/TNET.2014.2320577.
[8] T. Mori, M. Uchida, R. Kawahara, J. Pan, and S. Goto, "Identifying elephant flows through periodically sampled packets," in *Proc. IMC*, Taormina, Italy, Oct. 2004, pp. 115–120. doi: 10.1145/1028788.1028803.
[9] T. Karagiannis, A. Broido, M. Faloutsos, and K. C. Claffy, "Transport layer identification of P2P traffic," in *Proc. IMC*, Taormina, Italy, Oct. 2004, pp. 121–134. doi: 10.1145/1028788.1028804.
[10] B. Anderson and D. A. McGrew, "Machine learning for encrypted malware traffic classification: Accounting for noisy labels and non-stationarity," in *Proc. SIGKDD*, Halifax, NS, Canada, Aug. 2017, pp. 1723–1732. doi: 10.1145/3097983.3098163.
[11] H. Kim, K. C. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: Myths, caveats, and the best practices," in *Proc. CoNEXT*, Madrid, Spain, Aug. 2008, p. 11. doi: 10.1145/1544012.1544023.
[12] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *Proc. SIGMETRICS*, Banff, Alberta, Canada, Jun. 2005, pp. 50–60. doi: 10.1145/1064212.1064220.
[13] T. Auld, A. W. Moore, and S. F. Gull, "Bayesian neural networks for Internet traffic classification," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 223–239, Jan. 2007. doi: 10.1109/TNN.2006.883010.
[14] A. Este, F. Gringoli, and L. Salgarelli, "On-line SVM traffic classification," in *Proc. IWCMC*, Istanbul, Turkey, Jul. 2011, pp. 1778–1783. doi: 10.1109/IWCMC.2011.5982804.
[15] W. Li and A. W. Moore, "A machine learning approach for efficient traffic classification," in *Proc. MASCOTS*, Oct. 2007, pp. 310–317. doi: 10.1109/MASCOTS.2007.2.
[16] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015. doi: 10.1109/TITS.2014.2345663.
[17] Z. Ma, H. Yu, W. Chen, and J. Guo, "Short utterance based speech language identification in intelligent vehicles with time-scale modifications and deep bottleneck features," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 121–128, Jan. 2019.
[18] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, "Traffic classification on the fly," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 2, pp. 23–26, 2006. doi: 10.1145/1129582.1129589.
[19] Z. Ma, Y. Lai, W. B. Kleijn, Y. Song, L. Wang, and J. Guo, "Variational Bayesian learning for dirichlet process mixture of inverted Dirichlet distributions in non-Gaussian image feature modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 449–463, Feb. 2019.
[20] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010. doi: 10.1109/TKDE.2009.191.
[21] J. Han, D. Zhang, G. Cheng, L. Guo, and J. Ren, "Object detection in optical remote sensing images based on weakly supervised learning and high-level feature learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 6, pp. 3325–3337, Jun. 2015.
[22] H.-C. Shin *et al.*, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Trans. Med. Imag.*, vol. 35, no. 5, pp. 1285–1298, May 2016. doi: 10.1109/TMI.2016.2528162.
[23] Z. Yang, R. Salakhutdinov, and W. W. Cohen. (2017). "Transfer learning for sequence tagging with hierarchical recurrent networks." [Online]. Available: https://arxiv.org/abs/1703.06345
[24] V. Perera, T. Chung, T. Kollar, and E. Strubell, "Multi-task learning for parsing the alexa meaning representation language," in *Proc. 22nd AAAI Conf. Artif. Intell., (AAAI), 30th Innov. Appl. Artif. Intell. (IAAI), 8th AAAI Symp. Educ. Adv. Artif. Intell. (EAAI)*, New Orleans, LA, USA, Feb. 2018, pp. 5390–5397.
[25] J. Han, D. Zhang, X. Hu, L. Guo, J. Ren, and F. Wu, "Background prior-based salient object detection via deep reconstruction residual," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 8, pp. 1309–1321, Aug. 2015.
[26] D. Zhang, D. Meng, and J. Han, "Co-saliency detection via a self-paced multiple-instance learning framework," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 5, pp. 865–878, May 2017.
[27] G. Sun, L. Liang, T. Chen, F. Xiao, and F. Lang, "Network traffic classification based on transfer learning," *Comput. Elect. Eng.*, vol. 69, pp. 920–927, Jul. 2018.
[28] N. Moustafa, B. Turnbull, and K.-K. R. Choo, "An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of Internet of Things," *IEEE Internet Things J.*, to be published.
[29] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006. doi: 10.1109/TPAMI.2006.79.
[30] Z. Ma, J.-H. Xue, A. Leijon, Z.-H. Tan, Z. Yang, and J. Guo, "Decorrelation of neutral vector variables: Theory and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 129–143, Jan. 2018.
[31] H. Al-Sahaf, M. Zhang, and M. Johnston, "A one-shot learning approach to image classification using genetic programming," in *Proc. AI*, Dunedin, New Zealand, Dec. 2013, pp. 110–122. doi: 10.1007/978-3-319-03680-9_13.
[32] *Waikato internet traffic storage*. [Online]. Available: https://wand.net.nz/wits/index.php
[33] Y. Liu and Y. F. Zheng, "FS_SFS: A novel feature selection method for support vector machines," *Pattern Recognit.*, vol. 39, no. 7, pp. 1333–1345, 2006. doi: 10.1016/j.patcog.2005.10.006.
[34] Z. Ma, A. E. Teschendorff, A. Leijon, Y. Qiao, H. Zhang, and J. Guo, "Variational Bayesian matrix factorization for bounded support data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 876–889, Apr. 2015.
[35] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.

HAIFENG SUN received the Ph.D. degree from the Beijing University of Posts and Telecommunications, in 2017, where he is currently a Lecturer. His research interest includes data mining, information retrieval, and next generation networks.

YUNMING XIAO is currently pursuing the bachelor's degree with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. His research interest includes next generation networks and network intelligence.

JING WANG received the Ph.D. degree from the Beijing University of Posts and Telecommunications, in 2009, where she is currently an Associate Professor. Her research interests include consumer electronic, network intelligence, the mobile Internet, and ubiquitous services.

JINGYU WANG received the Ph.D. degree from the Beijing University of Posts and Telecommunications, China, in 2008, where he is currently an Associate Professor. His research interests include span broad aspects of performance evaluation for Internet and overlay networks, consumer electronic, traffic engineering, image/video coding, and multimedia communication over wireless networks.

QI QI received the Ph.D. degree from the Beijing University of Posts and Telecommunications, in 2010, where she is currently an Associate Professor. Her research interests include SIP protocol, communications software, next generation networks, ubiquitous services, and multimedia communication.

JIANXIN LIAO received the Ph.D. degree from the University of Electronics Science and Technology of China, in 1996. He is currently the Dean of the Network Intelligence Research Center and a Full Professor of the State Key laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. He has published 100s of research papers and several books, and has been granted dozens of patents for inventions. His current research interests include mobile intelligent networks, service network intelligent, networking architectures and protocols, and multimedia communication. These achievements conferred the National Prize for Progress in Science and Technology in 2004 and 2009, respectively. He has received number of prizes, which include the Premier's Award of Distinguished Young Scientists from the National Natural Science Foundation of China in 2005, and the Specially-invited Professor of the Yangtse River Scholar Award Program by the China Ministry of Education, in 2009.

XIULEI LIU received the Ph.D. degree in computer science from the Beijing University of Posts and Telecommunications, in 2013. He was a Visiting Ph.D. Student with CCSR, University of Surrey, from 2008 to 2010. He has been a Lecturer with the Computer School, Beijing Information Science and Technology University, China, since 2013. His research interests include semantic sensor, semantic web, knowledge graph, semantic information retrieval, and so on.