A Transfer Learning Approach for Rapid Classification of Networks Traffic

Umer Majeed, Choong Seon Hong
Department of Computer Science and Engineering,
Kyung Hee University, Yongin, Korea
Email: {umermajeed, cshong}@khu.ac.kr

Abstract
Traffic classification is a preliminary step to ensure reliable network service provision and effective resource management. Deep learning-based traffic classification schemes are trendy as a result of their capability to recognize even encrypted traffic. Transfer learning is an effective method to share knowledge between interconnected domains. In this paper, we implemented transfer learning to enhance the accuracy as well as decrease the learning time of the target model for traffic classification. The simulation results show that the target model has better accuracy than the baseline model. Moreover, the convergence time and the corresponding number of epochs of the target model are less than the base model.

I. INTRODUCTION
Traffic classification as part of effective network management is the process to categorize network traffic into relevant classes automatically for improved Quality of Service (QoS), security, routing, and network diagnostics. Traffic classification allows corporations to maintain compliance with organizational network access policies. Virtual Private Network (VPN) offers businesses, data confidentiality by establishing an end-to-end private tunnel over third-party networks. However, VPN is a serious obstacle for conventional traffic classification schemes.

Deep learning-based traffic classification frameworks [1] are valued for their capability to reliably identify the normal traffic as well as VPN encrypted traffic without explicit feature search.

Transfer learning [2] is an effective way for knowledge sharing between domains of two related tasks. In model-based transfer learning, we have “Source model” and “Target model”. The weights from the source model are assigned to the target model as initial weights. Afterward, the source model is further trained as per its domain. In this paper, we employed transfer learning for traffic classification.

The contribution of this study is summarized as follows:
• We employed transfer learning for traffic classification. For this, we trained the source model and assigned its weight to the target model as initial weights.
• The target model trained using transfer learning outperforms the baseline model trained from scratch in terms of time efficiency.

The rest of the paper is organized as follows: Section II illustrates the system model for transfer learning for rapid network traffic classification. Section III formulates the transfer learning problem for rapid network traffic classification. Section IV briefly describes the dataset used. Section V gives the simulation results and Section VI concludes our work.

II. SYSTEM MODEL
The system model consists of a source module and a target module. The source module trains the source model $M_S$ on the source dataset $D_S$. The target module trains the target model $M_T$ on the target dataset $D_T$. Source dataset $D_S$ contains the flow-based time-related features with label space indicating both application level and VPN/non-VPN traffic characterization. While, the target dataset $D_T$ contains the flow-based time-related features with label space indicating VPN/non-VPN traffic classification only.

III. PROBLEM FORMULATION
Consider source dataset $D_S$ having sample space $I_S = (X_S, Y_S)$, where $X_S$ is feature space and $Y_S$ is label space.
Similarly, the target dataset $D_T$ has sample space $\mathcal{I}_T = (\mathcal{X}_T, \mathcal{Y}_T)$, where $\mathcal{X}_T$ is feature space and $\mathcal{Y}_T$ is label space.

The two datasets have same feature space but the label space is different. Formally:

\[ \mathcal{X}_S = \mathcal{X}_T, \ \mathcal{Y}_S \neq \mathcal{Y}_T, \ \mathcal{I}_S \neq \mathcal{I}_T, \ D_S \neq D_T, \ S \neq T. \]  

For source domain $D_S = \{\mathcal{X}_S, P(\mathcal{X}_S)\}$, We have source task $\mathcal{T}_S = \{\mathcal{Y}_S, f_S(.)\}$. Where $P(\mathcal{X}_S)$ is marginal probability distribution, $\mathcal{X}_S = \{x_{S1}, ..., x_{Sn}\} \in \mathcal{X}_S$, and $f_S(.)$ is source predictive function. Similarly, for target domain $D_T = \{\mathcal{X}_T, P(\mathcal{X}_T)\}$, We have target task $\mathcal{T}_T = \{\mathcal{Y}_T, f_T(.)\}$. Where $P(\mathcal{X}_T)$ is marginal probability distribution, $\mathcal{X}_T = \{x_{T1}, ..., x_{Tn}\} \in \mathcal{X}_T$, and $f_T(.)$ is the target predictive function [3].

We formulate our transfer learning problem as: given source domain $D_S$ with source task $\mathcal{T}_S$ and target domain $D_T$ with target task $\mathcal{T}_T$, increase the learning accuracy of $f_T(.)$ in $D_T$ and decrease corresponding training time $t_T$ using the knowledge from $D_S$ and $\mathcal{T}_S$, where:

\[ D_T \neq D_S, \ \mathcal{T}_S \neq \mathcal{T}_T, \ S \neq T. \]  

IV. DATASET

A. Dataset Details

The UNB ISCX VPN-nonVPN network traffic dataset [4] was used for Traffic classification. The dataset has time-related features for four timeouts with time-spans 120, 60, 30, and 15 seconds. The dataset is further divided into scenario A and scenario B dataset. Scenario A dataset characterizes the traffic at the application level in addition to VPN identification and has 14 classes. Scenario A dataset distinguish the traffic between the regular traffic (non-VPN traffic) and traffic encrypted by VPN (VPN-traffic). So, Scenario A dataset has 2 classes. The source model $M_S$ was trained on the scenario B dataset. The target model $M_T$ was trained on scenario A dataset.

B. Preprocessing

Since time-related features are highly correlated with the timeout, we normalized the datasets using the standard score for each timeout independently. The standard score is calculated for each feature in feature space as $z = \frac{x - \mu}{\sigma}$, where $z$, $x$, $\mu$, $\sigma$ is standard score, raw score, mean, and standard deviation respectively.

C. Splitting

For both source and target domains, validation dataset $D_v$ is 20 percent of the whole dataset of corresponding scenario. While, training dataset is the leftover (80 percent) of the whole dataset.

V. SIMULATION RESULTS

Table. I shows the layered architecture for source model $M_S$ and target model $M_T$. Table. II also indicates the freezed and trainable layers for target model $M_T$. We used stochastic gradient descent (SGD) optimizer for the training of all models. We used Tensorflow [5] and Keras [6] library for the training of both source and target models.

We first trained the source model $M_S$ on $D_S$ for 1000 epochs. We used the model with maximum validation accuracy for further processing. Fig. 2 shows the training and validation accuracy for source model $M_S$. Table. II shows the corresponding performance metrics.

Afterward, we assigned the weights of $M_S$ to corresponding freezed layers of the target model $M_T$ and trained the trainable layers of $M_T$ on $D_T$. 

---

### TABLE I

**LAYERED ARCHITECTURE FOR SOURCE MODEL $M_S$ AND TARGET MODEL $M_T$**

<table>
<thead>
<tr>
<th>Sr</th>
<th>Layer</th>
<th>Activation</th>
<th>Value</th>
<th>Value</th>
<th>Trainable/Freezed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input</td>
<td>-</td>
<td>(23,)</td>
<td>(23,)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Freezed</td>
</tr>
<tr>
<td>3</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Freezed</td>
</tr>
<tr>
<td>4</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Freezed</td>
</tr>
<tr>
<td>5</td>
<td>Dropout</td>
<td>-</td>
<td>0.2</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Freezed</td>
</tr>
<tr>
<td>7</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Freezed</td>
</tr>
<tr>
<td>8</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Freezed</td>
</tr>
<tr>
<td>9</td>
<td>Dropout</td>
<td>-</td>
<td>0.2</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Trainable</td>
</tr>
<tr>
<td>11</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Trainable</td>
</tr>
<tr>
<td>12</td>
<td>Dense</td>
<td>Relu</td>
<td>512</td>
<td>512</td>
<td>Trainable</td>
</tr>
<tr>
<td>13</td>
<td>Dense</td>
<td>Softmax</td>
<td>14</td>
<td>2</td>
<td>Trainable</td>
</tr>
</tbody>
</table>
TABLE II
PERFORMANCE METRICS OF \( M_S \) ON VALIDATION DATASET \( D_V \)

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F-1</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M_S )</td>
<td>0.82</td>
<td>0.79</td>
<td>0.80</td>
<td>0.82</td>
</tr>
</tbody>
</table>

As a baseline, we also trained baseline model \( M_B \) on target dataset \( D_T \) from scratch. Baseline model \( M_B \) has the same architecture as of \( M_T \) except that all of its layers are trainable. We trained target model \( M_T \) and baseline model \( M_B \) for 600 epochs and selected the best model based on validation accuracy using call-backs.

Fig. 3 shows the training and validation accuracy for target model \( M_T \) as well as baseline model \( M_B \). The target model \( M_T \) gained maximum validation accuracy of 0.9135 at epoch 207 while baseline model \( M_B \) gained maximum validation accuracy of 0.8455 at epoch 595. We measured the time taken by \( M_B \) and \( M_T \) for 600 epochs and corresponding highest validation accuracy in the current paradigm. Fig. 4 shows that the target model \( M_T \) takes very less time compared to baseline model \( M_B \) for training as there are less number of training parameters in target model \( M_T \). Table III shows the corresponding performance metrics.

TABLE III
PERFORMANCE METRICS OF \( M_T \) AND \( M_B \) ON VALIDATION DATASET \( D_V \)

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F-1</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M_T )</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>( M_B )</td>
<td>0.84</td>
<td>0.84</td>
<td>0.84</td>
<td>0.84</td>
</tr>
</tbody>
</table>

VI. CONCLUSION
Efficient traffic classification is significantly important for the quality of service in cognitive radio access network management. In this paper, we did transfer learning for traffic classification. The target model outperforms the baseline model in terms of accuracy, the number of iterations, and training time.

REFERENCES