applied sciences

Article

A Deep Similarity Metric Method Based on Incomplete Data for Traffic Anomaly Detection in IoT

Xu Kang, Bin Song * and Fengyao Sun

The State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China; xkang0591@gmail.com (X.K.); rayfengsofay@163.com (F.S.)

* Correspondence: bsong@mail.xidian.edu.cn; Tel.: +86-29-8820-4409

Received: 31 October 2018; Accepted: 23 December 2018; Published: 2 January 2019

Abstract: In recent years, with the development of the Internet of Things (IoT) technology, a large amount of data can be captured from sensors for real-time analysis. By monitoring the traffic video data from the IoT, we can detect the anomalies that may occur and evaluate the security. However, the number of traffic anomalies is extremely limited, so there is a severe over-fitting problem when using traditional deep learning methods. In order to solve the problem above, we propose a similarity metric Convolutional Neural Network (CNN) based on a channel attention model for traffic anomaly detection task. The method mainly includes (1) A Siamese network with a hierarchical attention model by word embedding so that it can selectively measure similarities between anomalies and the templates. (2) A deep transfer learning method can automatically annotate an unlabeled set while fine-tuning the network. (3) A background modeling method combining spatial and temporal information for anomaly extraction. Experiments show that the proposed method is three percentage points higher than deep convolutional generative adversarial network (DCGAN) and five percentage points higher than AutoEncoder on the accuracy. No more time consumption is needed for the annotation process. The extracted candidates can be classified correctly through the proposed method.

Keywords: anomaly detection; incomplete data; similarity metric; transfer learning; background modeling

1. Introduction

Pattern recognition is one of the essential components of the Internet of Things (IoT) system and service. IoT provides an extensive range of services with a complex architecture of multiple sensors integrating together, one of the important applications is the intelligent transportation service. With the development of urbanization, more and more people choose to drive their own cars. The fast-growing private cars bring more convenience to ordinary families, but they also bring great pressure to the urban road conditions. Bad road conditions will waste drivers' time as well as threaten the safety of people on roads. With all kinds of image sensors installed at crossroads and bayonets, the cloud computing center can gather and process the image data through the IoT. Based on the analysis of the captured image and video data with computer vision and image processing algorithms, the traffic anomalies can be automatically detected and recognized in IoT. When anomalies take place, the sensors can catch the unusual situation and notify the drivers and /passengers in time by IoT to help them make the right decisions. Meanwhile, it also ensures the safety of other travelers on the road. Besides, the detected anomalies can be sent to the traffic center in time, assisting the traffic police to arrive at the scene in time to regulate the traffic.

Various behaviors in the IoT can be considered as abnormal events. In the traffic scene, the anomaly can be a car accident, an illegally stopped vehicle, or abnormal falling objects on the road such as a falling tire, a floating paper, or a road cone. These anomalies have some common characteristics. They usually take place with a small probability rather than happening frequently. The anomalies on the road do not appear in the previous frames or stopped from moving in previous frames. Anomalous objects are usually small, and their visual features are not easily distinguishable. Considering the characteristics of video abnormal events, computer vision methods like background modeling can be employed to model specific areas to find objects from moving to stationary in videos. Then the machine learning methods can be exploited to discriminate which categories the anomalies belong to. However, the task seems to be challenging because of the complex traffic environment. Moreover, anomalies are difficult to capture, the number of training samples for them is insufficient compared with the normal objects. Therefore, traditional machine learning methods cannot deal with this kind of incomplete data with few effective samples. Some transfer learning and unsupervised learning methods can be applied to detect multiple kinds of anomalies.

The IoT is a significant part of the new generation of information technology. According to the agreed protocol, it links any articles with the Internet to implement information exchange and communication. Therefore, intelligent identification, location, tracking, monitoring, and management can be realized in the Internet. In consideration of the IoT security, a number of challenges are indicated in [1–3]. An in-depth analysis of four major categories of anomaly detection techniques, including classification, statistics, information theory, and clustering are summarized in [4]. Zhang et al. utilize D2D communication and the IOV to analyze the preferences of vehicle users, so as to protect the users' privacy to prevent accidents [5]. Wang et al. introduced the deep reinforcement learning methods in 5G and IoT technology for users' security and privacy [6]. Riveiro et al. presented a visual analytics framework for multidimensional road traffic data and explored the detection of anomalous events [7]. Hoang proposed a new PCA technique and a new detection method for IoT with lower complexity [8]. A spatial localization constrained sparse coding approach for anomaly detection in traffic scenes is introduced in [9]. A fast and reliable method for anomaly detection and localization in video data showing crowded scenes is described in [10].

In recent years, Convolutional Neural Network (CNN) has been popularized in computer vision and image processing [11–13]. A new CNN architecture employing an adaptation layer and an additional domain confusion loss to learn a representation that is both semantically meaningful and domain invariant is proposed in [13]. Increasingly, researchers apply CNN to anomaly detection tasks [14–18]. Combining hand-crafted appearance and motion features with deep features, unsupervised deep learning methods are presented in [14,17]. Erfani et al. reduce the feature dimension when the model is fused, and the nonlinear expression ability of the model is improved [16]. Carrera et al. exploit a convolutional sparse model by learning a dictionary of filters from normal images [15]. Anomalies are then detected by identifying outliers from the model. Hasan et al. learn a generative model for regular motion patterns using multiple sources with very limited supervision [17]. Li et al. apply compressed-sensing (CS) theory to generate the saliency map to label the vehicles and exploit the CNN scheme to classify them [18]. Then a fully convolutional feed-forward AutoEncoder is built to learn both the local features and the classifiers. Several works based on similarity metric CNN have achieved excellent results in many scenarios and tasks [19–23]. Generator Adversarial Networks (GAN) can continuously learn the distribution of simulated data and generate samples with similar distribution of the training data [24–27]. Radford et al. proposed a class of CNNs called deep convolutional generative adversarial networks (DCGANs) which can learn a hierarchy of representations from object parts to scenes in both the generator and discriminator [24]. Odena et al. extended GAN to the semi-supervised context by outputting class labels from the discriminator network [25]. Bousmalis et al. utilized GAN to learn a transformation in the pixel domain from one domain to the other [26]. Kim et al. proposed the DiscoGAN to exploit relations between different domains so that the style from one domain to another can be successfully transferred [27].

In the past two years, AutoEncoders (AEs) are often used to detect abnormal events, especially in video surveillance [28–33]. Kodirov et al. proposed Semantic AutoEncoder (SAE) which project a visual feature vector into the semantic space for zero-shot learning [28]. Their model can also be applied to the supervised problem. Potapov et al. combined CNN and AE for cross-dataset transfer learning in person re-identification [29]. Spatio-temporal AutoEncoders are applied to detect the abnormal events [31,33]. Zhao exploited 3-dimensional convolutions to extracts features from both spatial and temporal dimensions for video representation [31]. Chong et al. proposed an architecture including a component for spatial feature representation and another component for temporal learning [33]. Fan exploited Gaussian Mixture Variational AutoEncoder (GMVAE) to model the video anomalies as the Gaussian components of the Gaussian Mixture Model (GMM) [30]. A two-stream network framework is employed to combine the appearance and motion anomalies. A dynamic anomaly detection and localization system including a sparse denoising AutoEncoder is proposed in [32].

In this article, we propose a similarity metric CNN based on a channel attention model for a road anomaly detection task. The method mainly includes (1) A Siamese network with a hierarchical attention model by word embedding so that it can selectively measure similarities between anomalies and the template. (2) A deep transfer learning method can automatically annotate unlabeled data while fine-tuning the network, improving the accuracy of transfer learning and reducing the additional time consumption compared with DCGAN method and AutoEncoder method. (3) A background modeling method combining spatial-temporal information which can detect the anomaly areas accurately.

The rest of the article is organized as follows: Section 2 introduces the traditional CNN and the similarity neural network. Section 3 describes the proposed channel attention model based on a Siamese structure and the transfer learning method about it. Section 4 shows the effectiveness of our model on incomplete datasets and some contrast experiments. We conclude this article in Section 5.

2. Materials

A large number of images are generated from the distributed sensors all over the IoT system. The CNN is one of the effective methods to deal with these massive data. It imitates the construction of a biological visual perception mechanism and can be used for supervised and unsupervised learning. By calculating grid-like topology features, CNN can learn a variety of data distributions steadily without additional feature engineering. In this section, we first introduce the basic structure and principle of CNN, as shown in Section 2.1. Then a CNN based on similarity measurement is described in Section 2.2. It can not only learn the probability distribution of a single picture, but also can measure the similarity of two input pictures. Several tasks like human face verification or object tracking have employed with this structure to get the most similar samples from a template image.

2.1. CNN

The traditional CNN structure is shown in Figure 1. In general, the basic structure of CNN consists of two layers, one is the feature extraction layer, the other is the feature mapping layer. The feature extraction layer is mainly composed of a convolutional layer and a pooling layer. The convolutional layer consists of several convolution units, and the parameters of each convolution unit are optimized by the backpropagation algorithm. The convolution kernel has the same number of channels as the feature maps. When implementing convolution, convolutional kernels slid on the feature map with fixed stride, and the sum of the inner product at the corresponding positions are mapped to the corresponding positions of the new feature maps. Each pixel in the feature map is only connected to the part of the previous feature map. This is called weight sharing. If the stride of convolution increases, the amount of computation in convolution will be remarkably reduced. Once the local feature is extracted, the location relationship between it and other features is also determined. The pooling layer often reduces the size of the feature map output of the convolution layer by down-sampling. It can improve the over fitting problem of the model while reducing the subsequent computation without changing the feature structure of the image. The sliding method of the down-sampling area is the same as it of the convolution kernel sliding in the convolution layer. When the current sampling area slides to a position, the maximum value or the average value of all the pixels in the sampled area is taken instead of the whole region. The feature mapping layer usually refers to the fully connected layer. Each neuron of the fully connected layer is connected to all the neurons of the previous layer to integrate the features extracted from multiple feature maps extracted by convolutional layer. Since the full connection layer is connected to all the output of the previous layer, the number of parameters it contains is much more than the convolution layer. In order to reduce the redundancy, some neurons need to be suppressed with some specific activation functions during training. For example, a dropout layer can inhibit neurons from propagating forward with a certain probability (usually 0.5). In the classification task, the length of the output vector from the last fully connected layer is the number of categories. The output vector is mapped to 0 and 1 through a softmax activation function, and its meaning is the predicted probability of this category.

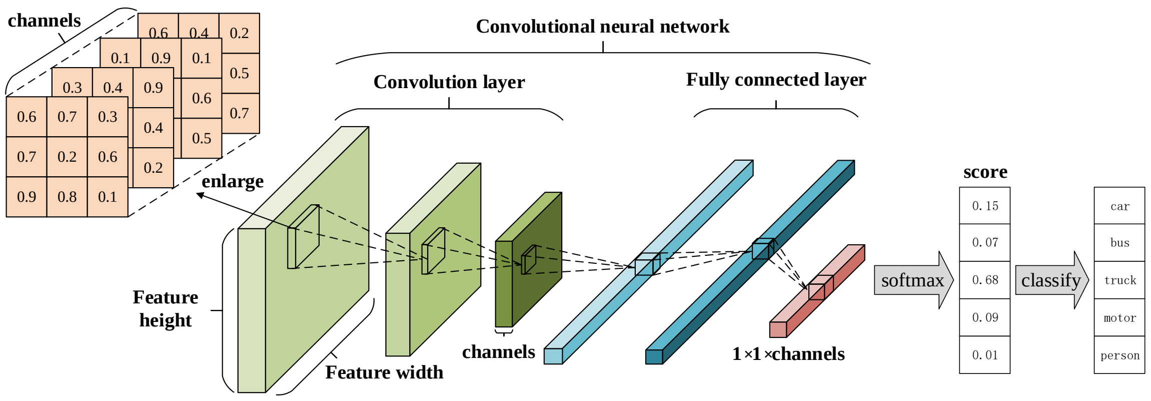


Figure 1. Architecture of a Convolutional Neural Network (CNN). The traditional CNN structure is mainly composed of convolution layers, pooling layers, fully connected layers, and some activation functions. Each convolution kernel is connected to the part of feature maps. The input is connected to all of the output elements in the fully connected layer.

2.2. Similarity Neural Network

The structure of similarity neural network is a little more complicated than the traditional CNN. Two image blocks are input to the convolution layer and pooling layer with same or different parameters, and then the high-dimensional features of the map are obtained. Then the decision network with shared parameters figures out the similarity of two image blocks [19]. The decision network is usually a fully connected layer. The output result is a similarity value, representing the matching degree of the two image blocks. The matching degree of two image blocks needs to be manually annotated before training. If the two image blocks are matched, the ideal output value is y = 1. To the contrary, if the two image blocks are different categories, the desired decision network output is Y = 0. When dealing with classification tasks, the loss function of CNN is the cross-entropy between the real probability distribution of the samples and the predicted probability distribution. It consists of two parts, one is the information entropy of the real distribution of the samples, the other is the Kullback–Leibler divergence between the real distribution of the sample and the predicted distribution:

$$\begin{aligned} H(p,q) &= -\sum_x p(x)\log q(x) \\ &= -\sum_x p(x)\log p(x) - \sum_x p(x)\log \tfrac{q(x)}{p(x)} \\ &= H(p) + KL(p\|q) \end{aligned} \tag{1}$$

Equation (1) represents the information difference between the two probability distributions. If the value is smaller, the two probability distributions are closer to each other.

Figure 2a indicates the schematic diagram of the bilinear similarity neural network. The main idea is to make two image blocks go through different branches to a fully connected decision network with shared weights. If the two branches have completely different structure and weight values, the mapping function of the two branches is irrelevant. This structure is called Pseudo-Siamese. Then the values and the size of the extracted features are also different. The information from the two images is fused together when the two high-level feature maps are transmitted to the first fully connected layer. The number of weights is over two times larger than that of the Siamese structure with shared weights. In addition, there have been demands for a greater amount of data and longer time for training.

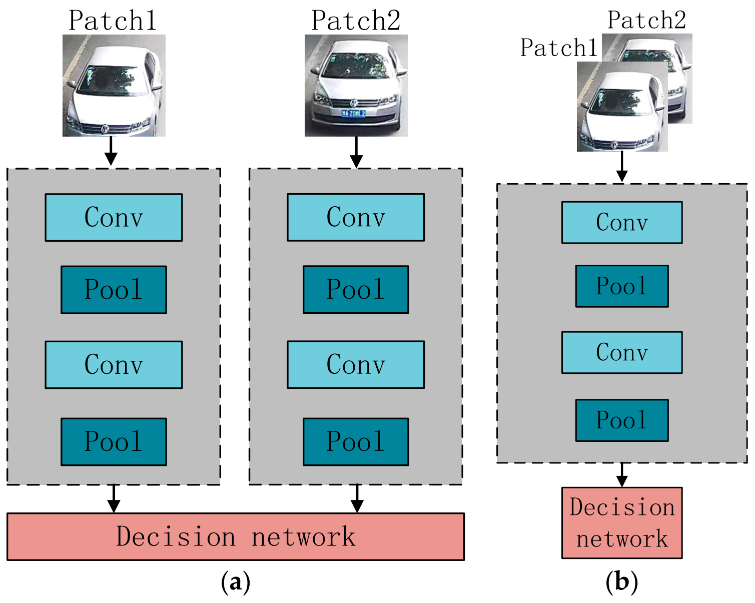


Figure 2. Similarity network. (**a**) Bilinear structure. The two image blocks are input into two branches to generate two features respectively. (**b**) Double channel structure. The fused two blocks are input with the network with shared weights and shared feature maps.

Figure 2b shows the illustration of the dual channel similarity neural network. The two image blocks are superimposed as the input. They share both the weights and feature maps during forwarding propagation. The scale of the model is remarkably reduced. In the process of forward propagation, the first layer of the feature map is generated by the convolution of the superimposed images, which contains common information of both images. After that, each feature is generated by the convolution of the previous layer, which integrates the information of both in the same way.

Several typical structures of CNN to measure the similarity between two image patches are indicated in [19]. A similar deep metric learning framework is employed for face verification in [20]. The works in [17,18] both use the Siamese structure for object tracking by measuring the similarity between the instance and a searching area. A cascade Siamese architecture which can learn spatial and temporal information separately is presented for person re-identification in [23].

3. Methodology

The traditional CNN has been introduced in the previous section. Adopting the bilinear structure shown in Figure 2a as the baseline, we provide our similarity metric CNN based on a channel attention model for road anomaly detection task in this section. At the beginning of this section, we start with the channel attention model for Siamese architecture. Two main components of the metric model are the residual blocks and the hierarchical channel attention model. Then aiming at our incomplete data-sets, we propose a deep transfer learning method. First, the model is trained on the data-set of limited samples, and then the remaining similarity matching network is trained on the new samples. At the end of this section, an anomaly detection method based on background modeling is introduced. By modeling the pixel structure of video frames, the objects from moving to stationary on the road can be detected as a candidate. Then the proposed model can classify these candidates selectively.

3.1. Channel Attention Model for Siamese Architecture

In recent years, ResNet [12] has been used in many fields such as detection, segmentation, recognition and so on. It was first introduced in 2015 and won the first place in the ImageNet classification competition. This kind of structure is for the sake of the problem that the accuracy of the network decreases on the validation set with the deepening of the network. As with the block module shown in Figure 3, two feature mapping functions are proposed in ResNet, one is the identity mapping, the other is residual mapping. The block modules are cascaded to form the overall structure of ResNet. Because only very few samples of anomaly objects on the road have been provided for training the model, there will be a serious over-fitting problem if a single ResNet structure is chosen to repeatedly train the net on this incomplete data-set. Under this circumstance, the model trained in this way has insufficient generalization ability to accurately predict the new test samples on the road.

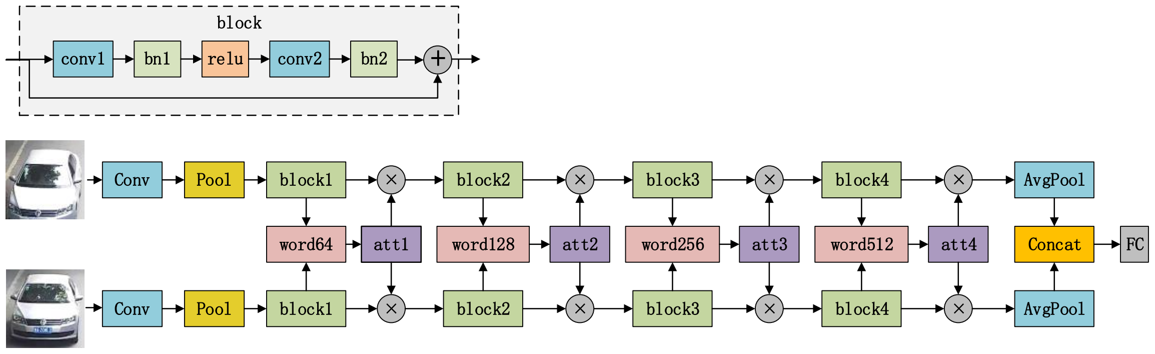


Figure 3. Overall framework of the proposed method. The simplest version of ResNet is selected as the main body of the two branches. The block indicates a deep residual structure in the dotted line. A soft attention model is appended after each block for feature location (att1, att2, att3, att4). The word vectors for each block have different lengths (word64, word128, word256, word512) according to hierarchical feature channels.

As shown in Figure 3, we choose ResNet18 as the main body of the two branches, where four blocks are interconnected with each other. The weights of the two branches are shared in both training and testing stages. The features output from the last block are down-sampled by an average pooling layer. Then the pooled features are concatenated together for high dimensional feature fusion. Afterwards, the fused feature is inputted into the full connected layer to get a vector for discriminating whether the two image patches belong to the same category. The two inputs indicate the images to be detected and the image already exists in the incomplete database, respectively. The candidate image is extracted from a specific area in the successive video frames when an anomaly object is detected in the subsequent frames.

An attention model is designed after each block, including a channel attention layer and a soft attention layer. The visual attention model is the brain signal processing mechanism peculiar to human vision. Human vision scans the global image quickly to get the focus area, which is generally called the focus of attention. Then, more attention resources are invested in this area to obtain more details of the target and suppress other useless information. We hope to input a word vector to the model through the attention mechanism, which can be used to select the categories of the anomaly object, and indirectly assist in identifying the similarity between the two images. The attention map generated from the attention model can mask the irrelevant areas on the feature map and highlight the salient region to calculate the similarity between two branches. The word vectors of different levels have different lengths while the attentional maps of different hierarchies have different resolutions.

The implementation details of the hierarchical attention mechanism are shown in Figure 4. $R1$ and $R2$ are the raw features extracted by the blocks and $R^l_{i,j,k}$ indicates the feature value at the position (i,j) of the k'th channel map extracted by l'th block. $Att^l_{i,j}$ denotes the attention value at the position (i,j) on the soft attentional map generated after l'th block. $Ratt1$ and $Ratt2$ represent the attentional features masked by the attention map of both branches while $Ratt^l_{i,j,k}$ is the value at the position (i, j) of the k'th feature map after l'th block. R and $Ratt$ are both three-dimensional and the Att is two-dimensional. The word vector W^l is a one-dimensional vector generated from a dictionary. Dictionaries of different hierarchies are randomly generated, and each dictionary has several row vectors, the number of which is the same as the number of categories defined for the anomaly objects. The dictionary is generic, it will be used together in both training and testing stages. Once the dictionaries are initialized, the values in it are determined and will not change again. The proposed model has four dictionaries for word embedding. In the training stage, when the inputs of the two branches are the same category, the word vectors are consistent with the image categories. When the inputs of the two branches are different categories, the word vectors are consistent with the image category of one branch. In the testing stage, one input of the network is a template image block and the other is an image block to be detected. The word vectors are consistent with the category of the template image block. The lengths of vectors in the dictionary are the same as the number of channels in the corresponding level feature map for subsequent channel-attentions calculations. The weight W^l_k is the k'th value of the word vector corresponding to the k'th feature map in raw features R after each block. Then, the attentional map can be calculated as:

$$Att^l = \sigma\Big(CONV_{3\times3}\Big(W^l \oplus R1^l \oplus R2^l\Big)\Big) \tag{2}$$

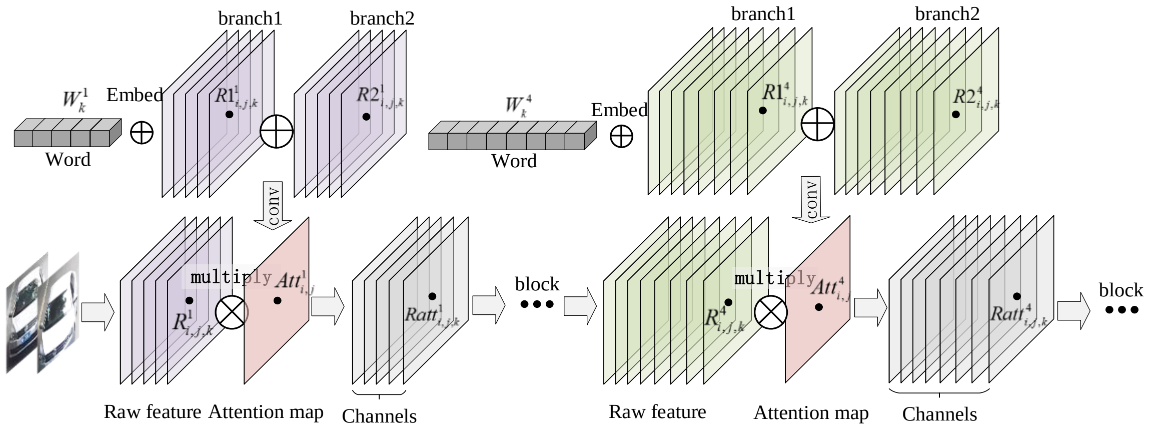


Figure 4. Architecture of multiresolution attention model. A word vector is embedded in the fused features of two branches for the generation of the shared attention map. Then, soft attention is implemented for both branches with the shared attention map. The same process is performed after each block.

The symbol $\oplus$ denotes the element-wise summation. The summation of $R1^l$ and $R2^l$ represents the feature fusion of the two branches. The add o l'th f W^l and R^l can be implemented as the summation of W^l_k and each element of $R^l_{i,j,k}$, $i \in [0, w-1]$, $j \in [0, h-1]$, indicating that the word feature is embedded in the fused features. A 3×3 convolution is employed after these operations, meaning that each element $Att^l_{i,j}$ on the map is generated from the $3 \times 3 \times C$ neighbourhood pixels of the feature at the same location (i, j) on R^l. The values on the attention map reflect the influence power of the fusion feature to the contribution of prediction in the adjacent area. σ is the sigmoid activation function, it maps all the values to $[0, 1]$. After that, the attentional features can be calculated as:

$$Ratt1^l = R1^l \otimes Att^l, Ratt2^l = R2^l \otimes Att^l \tag{3}$$

The symbol $\otimes$ denotes the element-wise multiplication, which can be implemented as the multiplication of $Att^l_{i,j}$ and each element of $R^l_{i,j,k}$, $k \in [0, C-1]$. It indicates that the raw features are filtered by the attention map. Information that is useful for classification and discrimination is preserved, while useless information is suppressed. It is worth mentioning that the attention map is shared for both branches, and the areas that the model pays attention to are the same at each hierarchy. The parameters, kernel size, kernel stride and the feature dimensions of our deep similarity metric model have been illustrated in Table 1.

Table 1. The parameters, kernel size, kernel stride and the feature dimensions of proposed model.

Layer		Kernel Size	Stride	Feature Size
Input		-	-	$224 \times 224 \times 3$
Conv1		$3 \times 7 \times 7 \times 64$	2	$64 \times 112 \times 112$
Pool1		3×3	2	$64 \times 56 \times 56$
Block1	conv	$64 \times 3 \times 3 \times 64$	1	$64 \times 56 \times 56$
	conv	$64 \times 3 \times 3 \times 64$	1	
	conv	$64 \times 3 \times 3 \times 64$	1	
	conv	$64 \times 3 \times 3 \times 64$	1	
Att1		$64 \times 3 \times 3 \times 1$	1	$56 \times 56 \times 1$
Block2	conv	$64 \times 3 \times 3 \times 128$	2	$128 \times 28 \times 28$
	conv	$128 \times 3 \times 3 \times 128$	1	
	conv	$128 \times 3 \times 3 \times 128$	1	
	conv	$128 \times 3 \times 3 \times 128$	1	
Att2		$128 \times 3 \times 3 \times 1$	1	$28 \times 28 \times 1$
Block3	conv	$128 \times 3 \times 3 \times 256$	2	$256 \times 14 \times 14$
	conv	$256 \times 3 \times 3 \times 256$	1	
	conv	$256 \times 3 \times 3 \times 256$	1	
	conv	$256 \times 3 \times 3 \times 256$	1	
Att3		$256 \times 3 \times 3 \times 1$	1	$14 \times 14 \times 1$
Block4	conv	$256 \times 3 \times 3 \times 512$	2	$512 \times 7 \times 7$
	conv	$512 \times 3 \times 3 \times 512$	1	
	conv	$512 \times 3 \times 3 \times 512$	1	
	conv	$512 \times 3 \times 3 \times 512$	1	
Att4		$512 \times 3 \times 3 \times 1$	1	$7 \times 7 \times 1$
AvgPool		7×7	1	$512 \times 1 \times 1$
FC		1024×2	-	2

3.2. Deep Transfer Learning Method

The goal of machine learning is to build a model as general as possible, so that the model can be well satisfied for different users, different devices, different environments and different requirements. However, the training and updating of machine learning models rely on data annotation. For our anomaly detection task, the most difficult is to continuously capture the abnormal events in the monitoring video, and labelling them manually in turn. Through the transfer learning method, we can find the similarity between annotated data and unannotated data, and then transfer the new model through knowledge reasoning. Fine-tuning in a deep neural network can help us save training time and improve learning accuracy, but it cannot deal with the different distribution of training data and test data. Deep neural network adaptation can make the data distribution of source domain and target domain closer, so that the classification accuracy of the network is better. Therefore, we propose a deep transfer learning method with two stages for our similarity metric network.

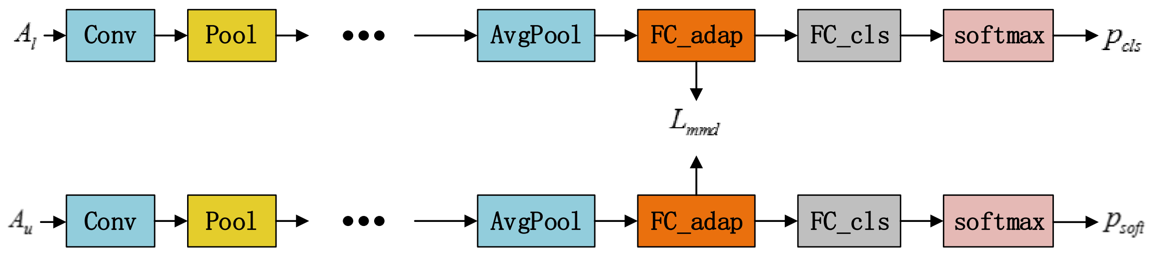
Our model transfer learning method is mainly divided into two training stages. The first stage is shown in Figure 5a. A_l and A_u in the figure indicate the labeled anomalies and the unlabeled anomalies later acquired. According to [12], an adaptive layer set up before classification layer achieves a good transfer learning effect. The prediction layer in the figure is initialized with the fully connected layer from original ResNet. We implement an adaptive layer after the average pooling layer in order to prevent the model from overfitting to the distribution of the labeled anomalies. According to maximum mean discrepancy (MMD) criterion, we want to achieve the same distribution as far as possible after the features of the two branches are mapped through the adaptive layer. L_{mmd} is used to measure the approximate degree of the distribution of annotated and unannotated anomalies in a mini-batch. L_{cls} is the softmax with cross-entropy of the classification loss.

$$L_{stage1} = L_{cls} + \lambda_1 L_{mmd} + \lambda_2 \|W\|^2 \tag{4}$$

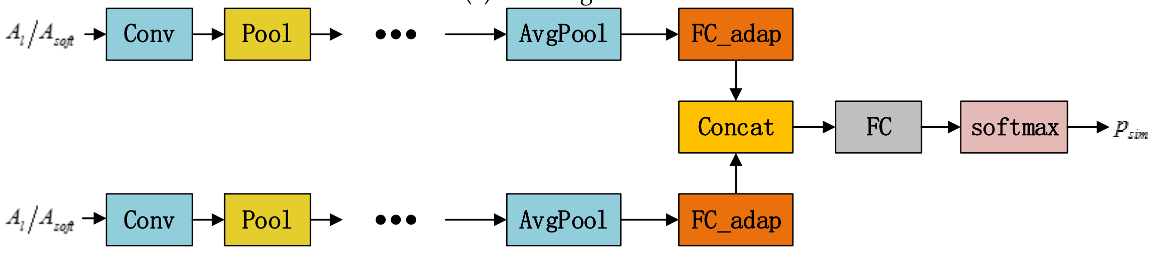
$$L_{mmd} = \frac{1}{B} \| \sum_{i=1}^{B} \left[fca\left(A_l^i\right) - fca\left(A_u^i\right) \right] \|^2 \tag{5}$$

$$L_{cls} = -\frac{1}{B} \sum_{i=1}^{B} \sum_{c=0}^{C-1} \delta\left(Y^i = c\right) \log p_{cls}^c \tag{6}$$

λ_1 and λ_2 in Formula (4) are two balanced parameters. $\|W\|^2$ is the regularization term to prevent overfitting. B is the size of the mini-batch during training. C is the number of categories defined for anomalies. $fca(\cdot)$ represents the output of adaption layer. p_{cls}^c is the predicted probability of c'th category. δ is a 0–1 loss function. Y^i is the label of i'th anomaly.



(**a**) First stage transfer



(**b**) Second stage transfer

Figure 5. Transfer learning in two stages. (**a**) First stage transfer: fine-tuning with an adaptive fully connected layer. (**b**) Second stage transfer: knowledge transfer with soft labels for joint learning. In the first stage, the network is fine-tuned by the labeled image while predicting the soft labels for the unlabeled image. In the second stage, the network is trained by both labeled image and unlabeled image. A fully connected adaptive layer is adopted for feature transfer.

In the first training stage, we use the classification layer to predict the unannotated anomalies simultaneously, in order to presumably infer the categories of anomalies. A soft label is defined as the category of the maximum probability predicted by the model in the whole process of transfer learning. The mathematical form is shown as follows:

$$SL = \underset{c \in [0, C-1]}{\operatorname{argmax}} \left[\frac{1}{T} \sum_{i=1}^{T} p_{soft}^c \right] \tag{7}$$

where T is the testing times of all the unannotated anomalies. p_{soft} is the predicted probability vector of the unannotated data.

In fact, the unlabeled data is predicted by the model every time the transfer training is performed. The output of the softmax function is a probability vector, the values of which reflects the possibility that it belongs to each category. Finally, categories corresponding to maximum values are considered as the soft labels according to the mean predictions throughout the training process.

In the second training stage, both annotated data and unannotated data are used to train the similarity metric network. Except for the FC layer, the parameters of each layer are initialized with these trained in the first stage. When transferring the model from the first stage to the second stage, only the FC layer need to be completely trained while other layers just need to be fine-tuned. The outputs of the FC layer are two values, indicating their similarity and dissimilarity probabilities of the two inputs respectively. The softmax with cross-entropy loss is used to measure the similarity:

$$L_{simi} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{s=0}^{1}\delta\left(S^i = s\right)\log p_{sim}^{s}\left(A1_l^i, A2_l^i\right) - \frac{\lambda}{B}\sum_{i=1}^{B}\sum_{s=0}^{1}\delta\left(S^i = s\right)\log p_{sim}^{s}\left(A1_l^i, A2_u^i\right) \tag{8}$$

where $A1$ and $A2$ are the two inputs of two branches.

The loss function contains two parts. If one of the inputs is the anomalies with the soft label, a coefficient λ is set to weaken the influence of this part. S^i is the target similarity. If two inputs and the embedded word vector belong to the same category, we set $S^i = 1$. In order to make the network converge quickly on the incomplete data-sets, we control the class of input samples of one branch to be the same as that of word vectors. In the same way, the categories of samples for comparison and the word vector are unified. In this way, the similarity measurement of anomalies based on the complementary text is realized.

3.3. Anomaly Detection Based on Background Modeling

For anomaly detection, locating the object is one of the most critical parts. In this paper, the image difference between background masks based on background modeling is adopted to locate the abnormal objects. In surveillance videos, the traditional backgrounds usually include road surface, roadside cornerstone, and green belts. All kinds of image blocks have the characteristics of pixel similarity and continuity. According to the situation above, the combination of spatial-temporal information for background modeling is adopted. The conventional background modeling method such as the Gaussian Mixture Model (GMM) depends on statistics of a single pixel, without considering the spatial distribution of pixels in the background. In view of this problem, we propose the spatial-temporal information modeling algorithm for the improved K-Means pixel clustering to process the pixels in videos.

The image is converted to greyscale and processed by average down-sampling. Then n random pixels are considered as the initial clustering center. K-Means clustering on all pixels is performed through Euclidean distance. The efficiency of the category center is determined by the Calinski-Harabasz criterion:

$$CH(k) = \frac{trB(k)/(k-1)}{trW(k)/(n-1)} \tag{9}$$

where n is the number of clusters, k is the current class, $trB(k)$ is the trace of the inter-class covariance matrix, and $trW(k)$ is the trace of the intra-class covariance matrix. The larger CH indicates that the inter-class distance is closer while the intra-class distance is farther, leading to a better clustering result. The pixels closest to the center of the category in each class are recorded. Then the center points of n transferred categories are found in the original image, and their RGB pixel values are recorded as the category center in the subsequent clustering process. Afterwards, n transferred category pixel values are used to classify each block in the original image and the location distribution of each category on the modeled image is obtained.

In the proposed method, the temporal background information is modeled by the K-nearest neighbor method. The nearest K instances in the training data-set are found to classify a new input instance. An adaptive background modeling method is performed to get a real-time model and a time-delayed model. Specifically, the KNN background modeling is to find a hypersphere with volume V and diameter D centered on a sample point in the RGB space, which satisfies the requirement that there are more than K (generally $0.1 * T$) of the T samples inside the sphere. The Formula (10) is used to calculate whether the sample points $\vec{x}$ are the background points. That is, each element in the sample set X_T should be compared to determine whether it belongs to the background. The comparison between the two-pixel points is to calculate the Euclidean distance between them, and the initialization probability is as follows:

$$P_{init}\left(\vec{x}|X_T, BG_{Init}\right) = \frac{1}{TV}\sum_{m=t-T}^{t} K\left(\frac{\left\|\vec{x}^{(m)} - \vec{x}\right\|}{D}\right) = \frac{k}{TV} \tag{10}$$

where T is the number of historical frames, and t is the number of current video frames, K is a Gaussian kernel function. The point is considered as the background when its probability is greater than the probability of the background.

Each pixel is classified into different categories according to its distance to each class center acquired from spatial information. Whether each category is in the background depends on the initialization of the spatial information and the spatial distribution. Then, we can discriminate whether the pixel is the background point by the class and location of the pixels.

A comparison is made between the binary instant mask and the binary delayed mask in order to get the target bounding box in foreground whose overlap ratio is lower than a preset value (usually set to 0.1). The target bounding box appearing in instant mask but not appearing in the time-delayed mask can be considered as the potential anomalous targets. The overlap ratio formula is as follows:

$$IOU = \frac{region1 \cap region2}{region1 \cup region2} \tag{11}$$

where IOU is the overlap ratio of two bounding boxes in the image, defined as the ratio of intersection area to the merged area.

The bounding boxes of the target is continuously recorded in N successive frames. When the number of occurrences of detected objects exceeds $\lfloor 0.9 * N \rfloor$ in the N successive frames, the target is identified as an anomaly. Here, N needs to be set autonomously according to the given video and requirements, and $\lfloor \cdot \rfloor$ is a rounding down symbol. At the same time, the anomaly in the image is extracted from the original frame to the local hard disk for subsequent tasks such as classification and recognition.

4. Experiments

In the previous section, we describe the channel attention model based on Siamese architecture for traffic anomaly classification, our deep transfer learning method and background modeling method for anomaly detection. In this section we first introduce the incomplete data for our experiments. Then several experiments are implemented to validate the effectiveness of the proposed model and transfer learning method. After testing under various conditions, we finally find that the proposed method is superior to DCGAN method and AutoEncoder method in practical application. Finally, we show the anomalies detected and classified in real traffic data-sets.

The experiments are all performed by Pytorch (Pytorch 0.4.1: Facebook, Menlo Park, CA, USA) deep learning framework on Windows 10 Pro (Microsoft, Redmond, WA, USA) operating system. The experimental platform is equipped with Intel® Core(TM) i7-6700 CPU (Intel, Santa Clara, CA, USA) @ 3.40GHz, 32GB RAM and NVIDIA GeForce GTX 1060 GPU (NVIDIA, Santa Clara, CA, USA). Part of the training samples are shown in Figure 6. In the process of transfer learning, 39 images were selected as the labeled anomalies for training (11 cars, 10 trucks, 10 cones, and 8 tyres), 72 images were selected as unlabeled data, 746 images were considered as the testing data (199 cars, 225 trucks, 210 cones, and 112 tyres). The target is to transfer from labeled data to unlabeled data. The generalization ability depends on the performance of the model on testing data-sets.

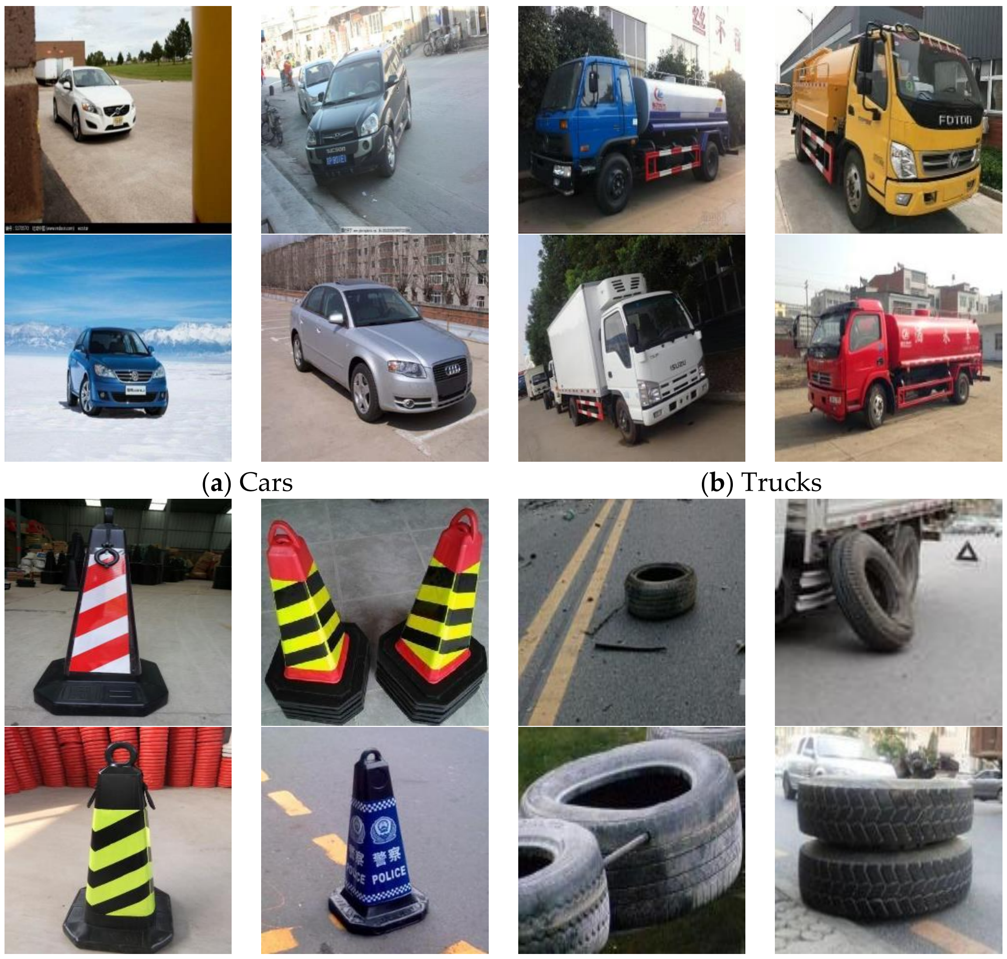


Figure 6. Several categories of training samples (**a**) cars (**b**) trucks (**c**) cones (**d**) tyres. 11 cars, 10 trucks, 10 cones, and 8 tyres are selected as the incomplete training samples. 199 cars, 225 trucks, 210 cones, and 112 tyres. 72 images were selected as unlabeled data.

The first stage fine-tuning with different layers is shown in Figure 7. According to the overall framework of the proposed model, the layers are divided into six parts: conv1, block1, block2, block3, block4, and fc layer. The contents displayed in the legend of each line represent the layers fine-tuned while other layers do not list in the legend denote that their parameters are fixed during the first stage fine-tuning. The adaptive moment estimation (Adam) method is employed to optimize the objective problem. The batch size was set to 16 and the learning rate of the fine-tuned parameters is set to 0.0001. The model is trained for 100 epochs. It is obvious from the figure that fine-tuning only the fully connected layer leads to poor performance on the testing set. No significant improvement is achieved when fine-tuning all layers. The two situations above both converge at a very slow rate. Fine-tuning only block4 and fc achieves fastest convergence rate and highest accuracy. Therefore, the parameters of conv1, block1, block2, and block3 are fixed in the subsequent experimental stages.

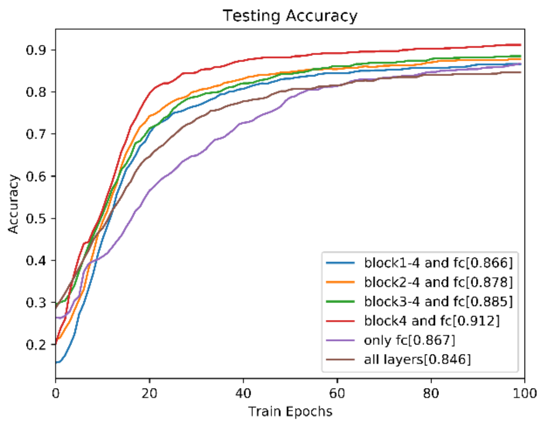
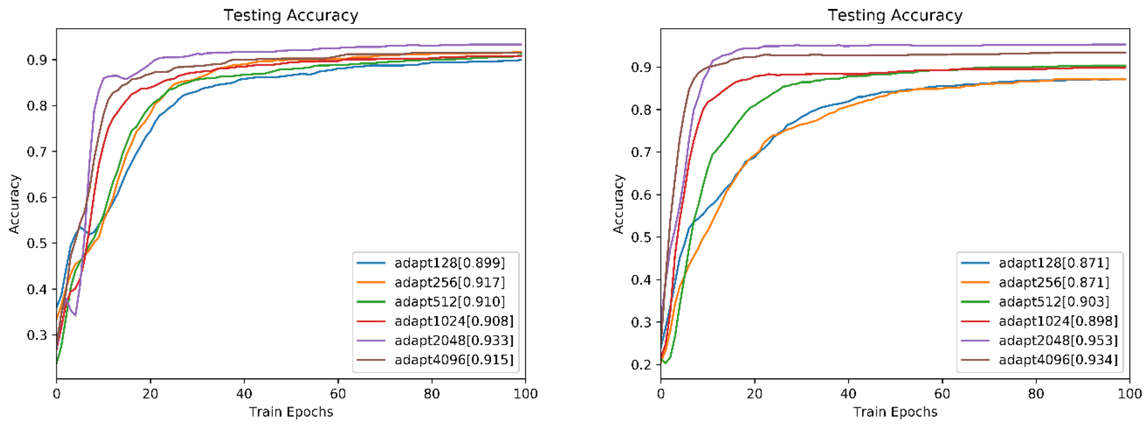


Figure 7. The first stage fine-tuning with different layers. When fine-tuning our network in the first stage, six situations were tested. The layers in the legend indicate that these layers were trained while the parameters of other layers were fixed during fine-tuning. Fine-tuning only block4 and fc achieves best classification results.

The effects of different adaptive layer sizes on the performance of the model are tested in Figure 8. Six different sizes of adaptive layers (128, 256, 512, 1024, 2048, 4096) are attempted in two stages of transferring respectively. The mini-batch gradient descent (MBGD) method is employed to optimize the objective problem. The batch size is set to 16 and the learning rate of the fine-tuned parameters is set to 0.0002. The model is trained for 100 epochs. In the first stage, the model with adaptive layers with lengths of 2048 and 4096 have the fastest convergence speed. The adaptive layers with a length of 2048 achieve the highest accuracy (just over 0.9). The curve of adaptive layer 4096 is next only to adaptive layer 2048. In the second stage, the model with adaptive layers length 4096 has the fastest pace of convergence. Finally, the adaptive layers of length 2048 achieved the highest accuracy (just over 0.95). The accuracy curves of adaptive layers 128 and 256 in the second stage decreases compared to their corresponding curves in the first stage. This is because the soft labels predicted in the first stage are not accurate, this directly affects the performance of transfer learning in the second stage.



(**a**) Fine-tuning in the first stage (**b**) Transfer learning in the second stage

Figure 8. Transfer learning with different adaptive layers: (**a**) Fine-tuning with different adaptive layers in the first stage; (**b**) Transfer learning with different adaptive layers in the second stage; Different lengths of adaptive fully connected layers were tested in both stages. Transfer learning with adaptive layers 2048 achieves best results.

The comparison of testing accuracy proposed method, DCGAN, and AutoEncoder is shown in Figure 9. The accuracy curve of the proposed method is the curve of the adaptive layer 2048 in last experiment in Figure 8. Because the label generation process of the incomplete data is simultaneous with the first stage of fine-tuning, there is no more extra time consumption.

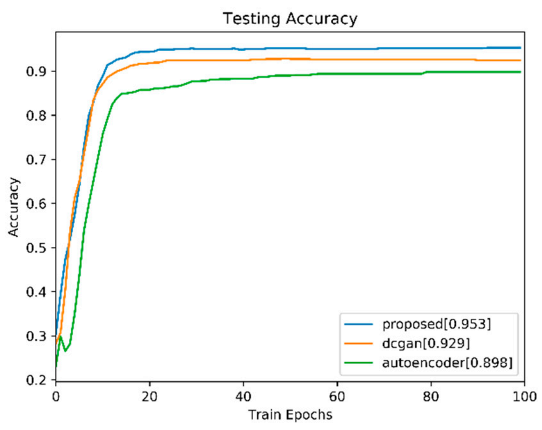


Figure 9. Comparison of testing accuracy of the proposed method, DCGAN, and AutoEncoder in second transferring stage. The batch size is set to 16 while the learning rate of the fine-tuned parameters is set to 0.0002. The three methods are all tested for 100 epochs.

The DCGAN is implemented by the methods in [24]. The labeled anomalies described in Figure 6 are used to train the discriminator and the generator. After 800 iterations, more than a thousand images are generated on these categories. Because most generated images are too similar to the original image, only a few images (2 cars, 8 cones, 3 trucks, and 8 tyres) are selected as the complementary samples to differ from the original training set for subsequent training. The mini-batch gradient descent (MBGD) method is employed to optimize the objective problem. The batch size is set to 4 and the learning rate of the fine-tuned parameters is set to 0.0002. The momentum is set to 0.5. It takes 4152 s for the DCGAN method to generate extra training samples for the second transferring stage.

The AutoEncoder is implemented by the methods in [28]. The labeled anomalies introduced in Figure 6 are used to train the AutoEncoder. We consider ResNet as the feature extraction method. The extracted features of training samples are considered as the input data matrix of the encoder, the parameters of the projection are trained by the reconstruction error. After training, the features of the train set and the features from the unlabeled set are all input into the encoder to get the semantic features. We use the semantic features of each category to cluster and generate the center of each category in the semantic space. The unlabeled features from ResNet then are projected to the semantic to get the annotation by $\underset{i \in \{car,truck,cone,tyre\}}{\operatorname{argmin}} \|s_{unlabel} - c_i\|_2$, where $s_{unlabel}$ is the semantic features of unlabeled data and c_i is the clustering center. The size of the raw features output from ResNet is 2048 and the size of the semantic space is 200. It takes 134 s for the AutoEncoder method to extract deep features, train the encoder and decoder, and make clustering.

In the second transferring stage, the batch size is set to 16 while the learning rate of the fine-tuned parameters is set to 0.0002 for the three methods. The result of the proposed method is shown in the blue curve and the highest accuracy rate reaches 0.953. The curve of the method using DCGAN is shown in orange. The maximum accuracy achieved was 0.929. The method performed with AutoEncoder is shown by the green curve. Its testing accuracy can run up to 0.898. Compared with the final stable accuracy, the proposed method is 3 percentage points higher than DCGAN and 5 percentage points higher than the AE method. The comparison of all the performances of the three methods is shown in Table 2.

Table 2. Performances comparison of the three methods.

Methods	Proposed	DCGAN [24]	AutoEncoder [28]
Accuracy of transfer learning	0.953	0.929	0.898
Extra processing time for incomplete data (seconds)	0	4152	134

Figure 10 shows the application of this model in road traffic scenarios. Our background modeling method can extract abnormal objects from moving to stationary in the road. There are usually two classes of anomalies, one class is the falling objects, the other class is the illegally stopped vehicles. Because the original frame is too large to display, the first column only shows part of original frame. The image blocks in the middle column shows the anomalies extracted by the background modeling method. The values in the last column indicate the similarities between each image block and each class. As can be seen from the figure, the anomaly areas can be extracted by the proposed background modeling method. Most extracted candidates can be classified correctly through the proposed deep similarity metric neural network. The appearance of the white minivan in sequence 2 is between a car and a truck. It is difficult to discriminate whether it's a car or a truck. The training set has only four classes so that the person is classified as the closest type to him.

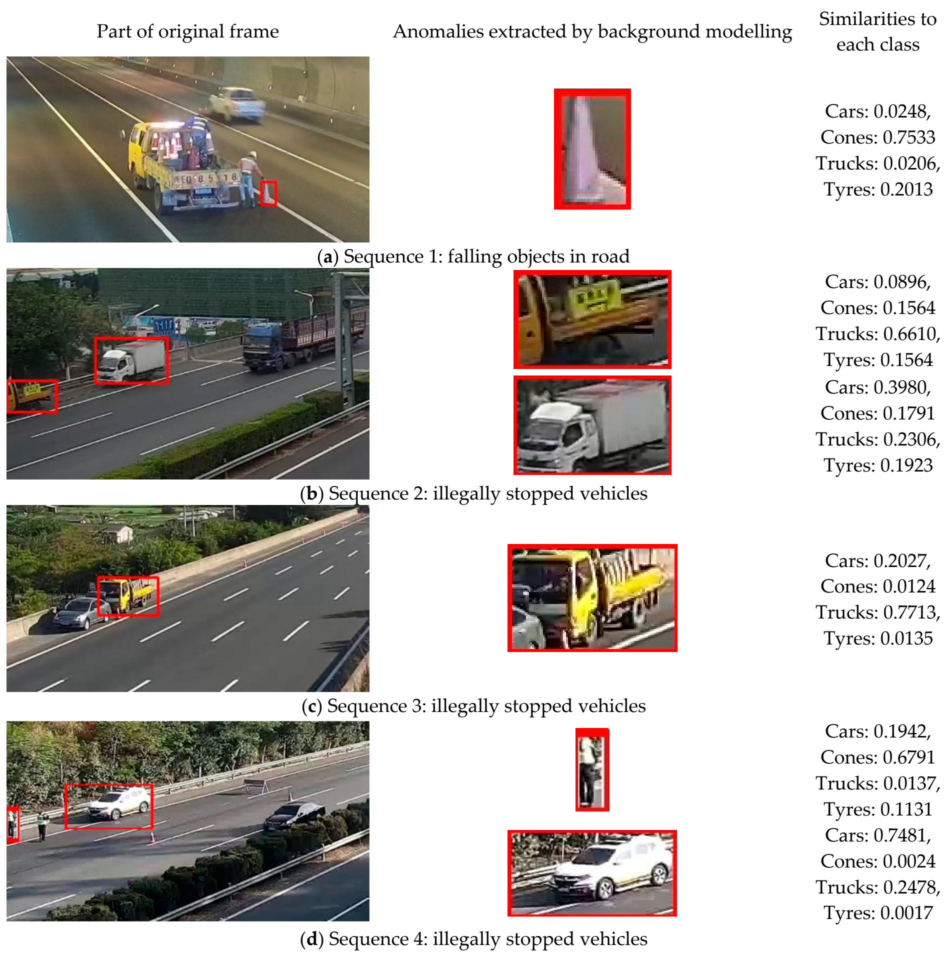

Part of original frame
Anomalies extracted by background modelling
Similarities to each class
Cars: 0.0248, Cones: 0.7533 Trucks: 0.0206, Tyres: 0.2013
(a) Sequence 1: falling objects in road
Cars: 0.0896, Cones: 0.1564 Trucks: 0.6610, Tyres: 0.1564
Cars: 0.3980, Cones: 0.1791 Trucks: 0.2306, Tyres: 0.1923
(b) Sequence 2: illegally stopped vehicles
Cars: 0.2027, Cones: 0.0124 Trucks: 0.7713, Tyres: 0.0135
(c) Sequence 3: illegally stopped vehicles
Cars: 0.1942, Cones: 0.6791 Trucks: 0.0137, Tyres: 0.1131
Cars: 0.7481, Cones: 0.0024 Trucks: 0.2478, Tyres: 0.0017
(d) Sequence 4: illegally stopped vehicles

Figure 10. Road anomaly detected by background modeling: (**a**) Sequence 1: falling objects in road; (**b**) Sequence 2: illegally stopped vehicles; (**c**) Sequence 3: illegally stopped vehicles; (**d**) Sequence 4: illegally stopped vehicles. The first column shows the Region of Interests (ROIs) from original frame. The image blocks in the middle column shows the anomalies extracted by the background modeling method. The values in the last column indicate the similarities between each image block and each class.

5. Conclusions

In this paper, we propose a similarity metric CNN based on a channel attention model for a traffic anomaly detection task. By extending ResNet to two branches, a Siamese structure can measure the similarity between two anomalies. Combining an attention model to generate a soft attentional mask from an embedded word vector, another modal information is complementary to the final decision. In order to solve the shortage of training samples and the overfitting problem, we employ a deep transfer learning method with two stages. In the first stage, the model is transferred from the original ResNet to the five classification model, while the soft labels are estimated adaptively. In the second stage, the model is transferred to two branches with the auxiliary attention model. Finally, a background modeling method is employed for anomaly extraction. Experimental results show that the proposed method achieves excellent results in our traffic data-set.

Author Contributions: X.K. and F.S. conceived and designed the experiments; X.K. and F.S. performed the experiments; X.K. and B.S. analyzed the data; B.S. contributed experimental data, tools and devices; X.K. wrote the paper. B.S. reviewed and edited the paper.

Funding: This research was supported by the National Natural Science Foundation of China under Grant (No. 61772387 and No. 61802296), Fundamental Research Funds of Ministry of Education and China Mobile (MCM20170202), the Fundamental Research Funds for the Central Universities, and supported by ISN State Key Laboratory.

Acknowledgments: The authors would like to thank anonymous reviewers for their valuable comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rakesh, N. Performance analysis of anomaly detection of different IoT datasets using cloud micro services. In Proceedings of the 2016 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 26–27 August 2016; pp. 1–5.
2. Fuller, T.R.; Deane, G.E. IoT applications in an adaptive intelligent system with responsive anomaly detection. In Proceedings of the 2016 Future Technologies Conference (FTC), San Francisco, CA, USA, 6–7 December 2016; pp. 754–762.
3. Vijayalakshmi, A.V.; Arockiam, L. A Study on Security Issues and Challenges in IoT. *Eng. Sci. Manag. Res.* **2016**, *3*, 34–43.
4. Ahmed, M.; Mahmood, A.N.; Hu, J. A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* **2016**, *60*, 19–31. [CrossRef]
5. Zhang, Y.; Tian, F.; Song, B.; Du, X. Social vehicle swarms: A novel perspective on socially aware vehicular communication architecture. *IEEE Wirel. Commun.* **2016**, *23*, 82–89. [CrossRef]
6. Wang, D.; Chen, D.; Song, B.; Guizani, N.; Yu, X.; Du, X. From IoT to 5G I-IoT: The Next Generation IoT-Based Intelligent Algorithms and 5G Technologies. *IEEE Commun. Mag.* **2018**, *56*, 114–120. [CrossRef]
7. Riveiro, M.; Lebram, M.; Elmer, M. Anomaly detection for road traffic: A visual analytics framework. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 2260–2270. [CrossRef]
8. Hoang, D.H.; Nguyen, H.D. A PCA-based method for IoT network traffic anomaly detection. In Proceedings of the 2018 20th International Conference on Advanced Communication Technology (ICACT), Chuncheon-si, Gangwon-do, Korea, 11–14 February 2018; pp. 381–386.
9. Yuan, Y.; Wang, D.; Wang, Q. Anomaly detection in traffic scenes via spatial-aware motion reconstruction. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 1198–1209. [CrossRef]
10. Sabokrou, M.; Fayyaz, M.; Fathy, M.; Klette, R. Deep-cascade: Cascading 3D deep neural networks for fast anomaly detection and localization in crowded scenes. *IEEE Trans. Image Process.* **2017**, *26*, 1992–2004. [CrossRef] [PubMed]
11. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.

12. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
13. Tzeng, E.; Hoffman, J.; Zhang, N.; Saenko, K.; Darrell, T. Deep domain confusion: Maximizing for domain invariance. *arXiv* **2014**, arXiv:1412.3474.
14. Xu, D.; Ricci, E.; Yan, Y.; Song, J.; Sebe, N. Learning deep representations of appearance and motion for anomalous event detection. *arXiv* **2015**, arXiv:1510.01553.
15. Carrera, D.; Boracchi, G.; Foi, A.; Wohlberg, B. Detecting anomalous structures by convolutional sparse models. In Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2015; pp. 1–8.
16. Erfani, S.M.; Rajasegarar, S.; Karunasekera, S.; Leckie, C. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognit.* **2016**, *58*, 121–134. [CrossRef]
17. Hasan, M.; Choi, J.; Neumann, J.; Roy-Chowdhury, A.K.; Davis, L.S. Learning temporal regularity in video sequences. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 733–742.
18. Li, Y.; Song, B.; Kang, X.; Du, X.; Guizani, M. Vehicle-Type Detection Based on Compressed Sensing and Deep Learning in Vehicular Networks. *Sensors* **2018**, *18*, 4500. [CrossRef] [PubMed]
19. Zagoruyko, S.; Komodakis, N. Learning to compare image patches via convolutional neural networks. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 4353–4361.
20. Sankaranarayanan, S.; Alavi, A.; Chellappa, R. Triplet similarity embedding for face verification. *arXiv* **2016**, arXiv:1602.03418.
21. Tao, R.; Gavves, E.; Smeulders, A.W. Siamese instance search for tracking. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1420–1429.
22. Bertinetto, L.; Valmadre, J.; Henriques, J.F.; Vedaldi, A.; Torr, P.H. Fully-convolutional siamese networks for object tracking. In Proceedings of the 2016 European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016; pp. 850–865.
23. Chung, D.; Tahboub, K.; Delp, E.J. A two stream siamese convolutional neural network for person re-identification. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1992–2000.
24. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
25. Odena, A. Semi-supervised learning with generative adversarial networks. *arXiv* **2016**, arXiv:1606.01583.
26. Bousmalis, K.; Silberman, N.; Dohan, D.; Erhan, D.; Krishnan, D. Unsupervised pixel-level domain adaptation with generative adversarial networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 95–104.
27. Kim, T.; Cha, M.; Kim, H.; Lee, J.K.; Kim, J. Learning to discover cross-domain relations with generative adversarial networks. *arXiv* **2017**, arXiv:1703.05192.
28. Kodirov, E.; Xiang, T.; Gong, S. Semantic AutoEncoder for zero-shot learning. *arXiv* **2017**, arXiv:1704.08345.
29. Potapov, A.; Rodionov, S.; Latapie, H.; Fenoglio, E. Metric Embedding AutoEncoders for Unsupervised Cross-Dataset Transfer Learning. In Proceedings of the International Conference on Artificial Neural Networks (ICANN), Island of Rhodes, Greece, 4–7 October 2018; pp. 289–299.
30. Fan, Y.; Wen, G.; Li, D.; Qiu, S.; Levine, M.D. Video anomaly detection and localization via gaussian mixture fully convolutional variational AutoEncoder. *arXiv* **2018**, arXiv:1805.11223.
31. Zhao, Y.; Deng, B.; Shen, C.; Liu, Y.; Lu, H.; Hua, X.S. Spatio-temporal AutoEncoder for video anomaly detection. In Proceedings of the 2017 ACM on Multimedia Conference, Mountain View, CA, USA, 23–27 October 2017; pp. 1933–1941.

32. Narasimhan, M.G.; Kamath, S. Dynamic video anomaly detection and localization using sparse denoising AutoEncoders. *Multimedia Tools Appl.* **2017**, *77*, 1–23. [CrossRef]
33. Chong, Y.S.; Tay, Y.H. Abnormal event detection in videos using spatIoTemporal AutoEncoder. In Proceedings of the International Symposium on Neural Networks, Sapporo, Japan, 21–26 June 2017; pp. 189–196.