Computing and Informatics, Vol. 39, 2020, 1282–1310, doi: 10.31577/cai_2020_6_1282

TIME-SENSITIVE ADAPTIVE MODEL FOR ADULT IMAGE CLASSIFICATION

Mohammad Reza Mazinani, Seyyed Mojtaba Hoseini
Kourosh Dadashtabar Ahmadi

Faculty of Electrical and Computer Engineering
Malek Ashtar University of Technology
15875-1774, Iran
e-mail: mazinany@gmail.com, mojtabahoseini@aut.ac.ir,
dadashtabar@mut.ac.ir

Abstract. Images play an important role in modern internet communications, but not all of the images shared by the users are appropriate, and it is necessary to check and reject the inappropriate ones. Deep neural networks do this task perfectly, but it may not be necessary to use maximum power for all images. Many easier-to-identify images may be classified at a lower cost than running the full model. Also, the pressure on the system varies from time to time, so an algorithm that can produce the best possible results for different budgets is very useful. For this purpose, a deep convolutional neural network with the ability to generate several outputs from its various layers has been designed. Each output can be considered as a classifier with its own cost and accuracy. A selector is then used to select and combine the results of these outputs to produce the best possible result in the specified time budget. The selector uses a reinforcement learning model, which, despite the time-consuming learning phase, is fast at execution time. Our experiments on challenging social media images dataset show that the proposed model can reduce the processing time by 32 % by sacrificing only 1.4 % of accuracy compared to the VGG-f network. Also, using different metrics such as F1-score and AUC (the Area Under the Curve in the accuracy vs. time budget chart), the superiority of the proposed model at different time budgets over the base model is shown.

Keywords: Adult content recognition, time-sensitive, cost-sensitive model, convolutional neural network, deep learning, image classification

Mathematics Subject Classification 2010: 68T10

1 INTRODUCTION

In image and video sharing platforms, users upload images or videos to the multimedia sharing platform's servers and share them publicly. In such platforms, latency in the appearance of shared data is not acceptable to the users, and images are expected to be visible immediately after upload. Also, according to the rules of most sites, some images and videos are in the category of unacceptable images and must be deleted. These categories can vary depending on the different platform policies, for example, violence, racism, and nudity [1, 2]. Due to the huge number of images uploaded to the image-sharing platforms, it is not possible to manually check all images. So, a high-speed automatic image recognition system becomes a necessity for them.

Additionally, by increasing the speed of the Internet and improving video transfer technologies, live video streaming is becoming more and more popular. Live streaming provides video frames instantly. Therefore, in live video streaming platforms, in addition to being careful in filtering objectionable video frames, processing online is also very important, and this filtering should not cause lag or delay in the live streaming. [3, 4] have tried to identify people who abuse these platforms and then attempt to remove this content or restrict the streamer user. Many providers who publish adult content on the live streaming platform with the intention of generating revenue or harassment, are broadcasting sex-related content intelligently. They may only show nude content in part of the video, or may only produce sexual content through gesture or voice, which is harder to detect. Hence, because it is not feasible to review the video by human operators, it is necessary to use algorithms to do the review thoroughly.

The image recognition process in image sharing platforms has other challenges. Some photos are taken by professional users in the right lighting conditions and are high-quality images, but some photos are taken by amateur users with mobile phones and are blurry, poorly lit, oversaturated, or nearly dark. Another type of challenge is because of the content of the images. Some sports, such as boxing and swimming, involve half-naked bodies but are not sexual images. Nude pictures of babies also show a naked picture of a human being, but it is not considered porn. People in the image may have clothes, but due to their facial or body posture, the whole image is considered sexual.

One way to overcome these challenges is to use deep convolutional neural networks (CNN). In recent years, very deep networks have been presented with acceptable accuracy in image classification on databases with 1 000 image categories, such as googlenet [5], vgg-verydeep [6], and resnet-152 [7]. We also use CNN in our model to accurately detect adult images. But the use of neural networks causes a new challenge. These networks usually take a long time to process each image, and the deeper the network, the longer the processing time. In this article, our main focus is on reducing processing time.

Despite the various challenges, many images on image sharing servers have simple content and can be detected with simple algorithms. For example, images of objects, animals, buildings, etc. can be accurately categorized into normal classes using simple convolutional neural networks with a small number of layers. Using simple networks saves computational time and cost, but is less accurate on more complex images. Therefore, it will be very advantageous to design a model that can provide tags for simple images at a lower computational cost using a simple network. This model should use more complex networks to accurately detect complicated images.

Also, in some cases, the classifier must be able to produce the best result given the limitations of available computational sources. Conditions like:

- The volume of requests to the server: In some hours, many requests are sent to the servers, and in other hours the requests may be much less than server processing capacity.
- Limited processing capacity of different devices: Processing power on portable devices such as mobile phones, wearables, IoT devices, or laptops is different. So the algorithm must have the ability to adapt to processing power.
- Increasing the processing accuracy by a cloud processor: Simple images can be processed by the user's device, and if the result is desirable, it can be displayed to the user; otherwise, the data can be sent to a cloud server to continue processing. [8].

Given the above conditions, it will be necessary to create a model that can process images with a limited source. In this article, we consider processing time as our limited resource. Time constraints can take many forms.

- The fixed time limit for each image (hard budget): The classifier should provide the best result based on the specified time limit. This model can be useful in cases where images are given to different devices for processing. As an example, in [9] each image is assigned to a separate processor, and when it reaches the time limit for each image, these processes end, and the obtained result is reported.
- The time limit for a set of images (average budget): A set of images is sent to the processor, and it is necessary to process them in a limited time interval. In this case, the processing time of each image is not limited separately, and the only limitation is the average processing time. This model is useful when the complexity of input images varies.

The model introduced in this article is designed to meet the needs of the average budget. The general structure of the model is shown in Figure 1. As shown in Figure 1, images uploaded by users are inserted in a buffer. The server reads a set of images and sends them with the time limit to our model. If the number of images in the buffer become too large, the server considers a shorter time limit, to increase the output rate of the buffer and reduce the size of the buffer.

Our goal is to provide a novel model with the ability to classify a set of images in a specified time limit, with the highest achievable accuracy. Our method does not apply the same processing to all images like static methods [10, 11, 12, 13],

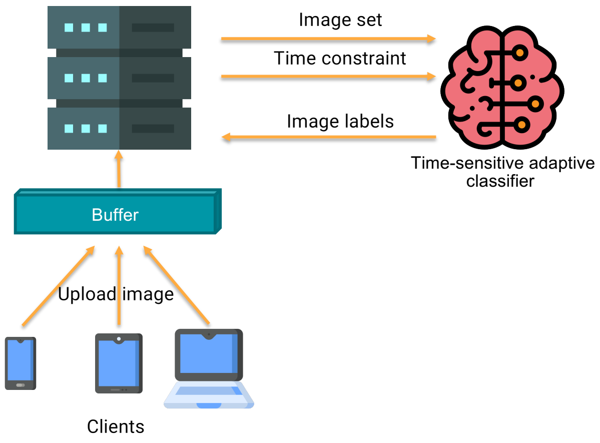


Figure 1. The overall structure of the time-sensitive adaptive model application on the image-sharing platform server (Icons made by Freepik from `https://www.flaticon.com`)

but it tracks a different process for each input instance. So our method could be considered as an input dependent method.

In this paper, a model is presented that can process a set of images with any time budget. The innovation of this research is the specific design of its selector, by which, in addition to the ability to finish the process at different time budgets, also the accuracy can be increased by combining the outputs of the different layers of the CNN classifier. The increase in speed and accuracy is due to the fact that the selector uses outputs of the earlier layers of the CNN classifier for simpler images, and continues to the deeper layers of the network for the more complex images. We used reinforcement learning to design the selector, which has many advantages for this task. The advantages of our selector model are described below.

- Improving the average speed of adult image classification: Our basic CNN contains several outputs from the middle layers. We trained the selector by the q-learning algorithm. The reinforcement learning agent tries to minimize the time cost using earlier layers of CNN, so the overall speed increased.
- Increasing accuracy by combining outputs: In cascading methods, the last classifier result is presented as the final model output. But in our model after observing the results of the outputs, the reinforcement learning agent decides which category is better to be presented as the correct final result, based on past experiences.
- Flexibility to select any subset of classifier outputs in any order: The selection of each of the classifier outputs is defined as an action in the reinforcement learning model. Therefore, unlike cascading models, there is no limit on the

order of classifier outputs selection. So, for example, when we have a lot of processing capacity, the selector may choose the most complex classifier output. This ability to freely choose between outputs allows the selector to be applied to any structure, such as a CNN with complex tree structure, or even several separate CNNs.

- The ability of the selector to work with different time budgets: The selection of classifier outputs is done in such a way that the cost of the classifier error and the cost of processing time are minimized. By increasing the time cost factor, the selector tends to select classifiers with lower cost. Therefore, by changing this factor, the selector will be trained for different time budgets. At runtime, any of these trained models can be easily used based on the given time budget.

The rest of the paper is organized as follows: Section 2 provides an overview of the previous works in the field of adult image recognition, as well as some related articles on cost-sensitive classification subject. Section 3 first demonstrates the contribution and justification of our model, and then explains the overall structure of the model and describes its parts in detail. Section 4 presents the obtained experimental results, and Section 5 discusses the properties of the proposed model and plans for future work.

2 RELATED WORK

Since our model consists of two primary parts, one is the base classifier, and the other is the selector, which is responsible for managing the outputs of this classifier. Therefore, articles related to these two areas and their advantages and disadvantages are described separately.

2.1 Adult Content Detection

Adult image recognition methods can be divided into three general categories: skin as the main feature, local features, and convolutional neural network (CNN). Due to the limitations of skin-based methods and local features, we chose deep neural networks for the classification part of the model. The limitations and weaknesses of these methods are discussed below.

2.1.1 Skin as the Main Feature

In these methods, the main focus is on finding skin regions, and then use the characteristics of these regions, such as the distribution of skin pixels, the area of skin regions, the number of skin regions, etc., to classify images [18, 19]. These methods do not recognize black and white images and do not have enough recognition capabilities in finding skin in different lighting conditions. To increase the ability to find skin, Lee et al. [20] find the faces in the image, and by sampling the skin pixels from the face, they detect skin pixels in other areas of the image. In skin-based methods, despite the increased accuracy in finding skin areas, the accuracy in detecting adult images was very low. So different methods added new features, such as the number of faces and the regions of the faces [21], texture [22], and shape [23] to increase accuracy. Although the classification error was reduced with the addition of new features, the processing time was increased significantly.

In addition to the problems mentioned, skin-based methods also have an intrinsic drawback because they usually do not take into account people's position. For instance, in some images, the skin area is very small; however, it is considered an adult image due to people's positions. Therefore, these methods have very low accuracies in practice.

2.1.2 Local Features

Classifiers based on local features, extract attributes and shapes from different parts of the image. Lopes et al. [24] use the scale invariant feature transform (SIFT) feature. First, these features are extracted locally from the image, and then all the features are formed in the form of histograms of visual features. It then uses this new feature to classify images. They use the standard SIFT feature, which is extracted from non-color images. They then use the Hue-SIFT feature, which adds color information to the previous feature and increases the accuracy of the algorithm. Hue-SIFT does not use the color characteristic properly, so to take advantage of the color feature, Deselaers et al. [25] use image patches as local features. They then reduce the size of image patches using PCA, and use the bags of visual words method to produce the final feature vector. They finally use created feature vector to classify images.

These methods are more accurate than skin based methods. Still, because they often use handcrafted features, they do not produce an accurate result on new and intricate images.

2.1.3 Convolutional Neural Network (CNN)

Recently, deep learning methods have much higher accuracy than previous algorithms, especially in image classification [26]. [27] uses deep learning to classify adult images, using a simple combination of two – Alex-net [28] and googlenet [29] networks. This paper demonstrates that the use of CNN is more accurate than pre-selected features such as SIFT.

Deep neural network training requires a large number of training images. Some articles use data augmentation, for example, flipping the image or using different images crop [30]. Another way to overcome the lack of training images problem is to use the weights of the pre-trained network to train a similar network (fine-tuning). Vitorino et al. [32] use this method to detect child pornographic images. They first choose a CNN that was trained on ImageNet [33] and fine-tuned it using 200 000 training adult images. Then, with a small dataset from children, they fine-tuned the trained network to detect child pornographic images.

We used data-augmentation and fine-tuning in our work and will explain them in Section 3.2.

Another way to increase the accuracy of classification is to combine multiple classifiers. Shen et al. [31] combined the results of several CNNs using Bayesian networks. Cheng et al. [34] combined the features extracted by the two CNNs and adopted the final feature vector for classification. In these methods, due to the processing of all CNNs, the processing time is greatly increased. Also, this processing is done for all images, regardless of the difficulty and ease of the image. In our method, the amount of processing depends on the input difficulty, so the overall processing time will be optimized.

Due to the superiority of the convolutional neural network for classification, we used this method to build our base classifier. But to generate output at different times, we added several intermediate outputs to the structure of one of the known CNNs. Table 1 shows a comparison of existing work in the field of adult image classification.

Model	Advantages	Disadvantages
adult image classification using skin region [18, 19]	classify high quality naked image	unable to recognize grayscale images. very low accuracy
adult image classification using skin plus face [20, 21] texture [22], or shape [23]	acceptable accuracy in color image	unable to recognize grayscale images. very time consuming
adult image classification using local features [24, 25]	medium accuracy on color and grayscale image	time consuming. not accurate on intricate images
adult image classification using CNN [27]	accurate	requires a large number of training images
adult image classification using CNN plus data augmentation [30], or fine-tuning CNN [32]	accurate. trainable by small training set	time consuming
adult image classification using combination of CNNs [31, 34]	very accurate	very time consuming

Table 1. Adult image classification methods comparison

2.2 Cost-Sensitive Models

In addition to trying to reduce the processing time of previous deep neural networks in a static way [10, 11, 12, 13], some input dependent work has been done to reduce time. In this section, we will review some related cost-sensitive articles. For a more informative description, these methods are divided into three categories.

2.2.1 Dynamic Pruning

Some works remove unnecessary nodes or layers within a CNN. For example, Bengio et al. [36] remove some nodes from the layers during the training and testing to reduce computations. Their method works like a dropout layer in CNN but has tried to estimate the best path in the neural network for different samples and deactivate unnecessary nodes. The other work by [16, 17] tries to reduce the processing time of the network by pruning the channels. For each input image, after each convolutional layer, the output channels that are not considered useful for image classification are pruned.

Because these methods use an integrated structure, they do not produce any results between processing steps. Therefore, after determining the result, it is not possible to increase the accuracy with more processing along the new path, and the whole network must be processed with new parameters from the beginning.

2.2.2 Decision Making with Threshold

Some works use a cascade structure to make decisions with the threshold. The cascade model is CNN with some output from the middle layers. They use the confidence value (obtained from the softmax layer output), which shows a number between zero and one for each class. At the time of inference, if the value of confidence is greater than a specific value, the model terminates the processing and shows the last obtained image label as a result of the whole network. Therefore, for some images, fewer processing steps are performed. Berestizshevsky and Even [14] used this method and selected the ResNet [7] network as the cascade model, with three outputs in the middle layers. The same method is used in [8], but the focus is on the use of CNN on devices that either have little memory to maintain all network parameters or do not have the computational power required to process in a specified short time. Therefore, fewer layers are processed on the portable device, and if more processing is required, the information is sent to the cloud server.

One of the advantages of this method is that they could change the decision threshold at test time. If the output accuracy is more important, increase the value of the decision threshold, and when the time budget is low, reduce the threshold value to increase the speed. And these methods are fast in selecting a processing path. But the disadvantage is that it can only be applied to the cascading model and cannot be applied to the network structure in the form of a tree. Also, specifying a fixed decision threshold may reduce the accuracy of the decision. Our method uses one decision-maker (selector) for all output and path of the network, and it could be used on any structure, and the final class is selected accurately using all available outputs.

2.2.3 Classifiers as Decision-Makers

The use of classifiers to make decisions increases accuracy and allows the use of non-cascade structures. In different network structures in each branch, a classifier has the task of choosing the path. For example, Bolukbasi et al. [35], used a cascading structure. At each branch, the classifier decides to exit the correctly identified samples from the processing path. Odena et al. [37] introduce a structure that has three metalayers, each consisting of two modules. Before processing each metalayer, a function uses previous outputs to decide which module to use in the metalayer so that the network can achieve the desired accuracy and speed. Liu and Deng [15] use control modules within the network to select the best processing path. Control modules are trained using the backpropagation algorithm and reinforcement learning. They tested their model on high-low, cascade, chain, and hierarchy structures.

These methods use complex classifiers to select the best route on CNN, so they have two drawbacks. First: using a complex classifier adds a time cost to the whole process. Second: the selector classifier error is added to the total image classification error. Our method at the time of testing is a lookup table, and the required processing time is very low. Also, to prevent errors, the next path is selected, after the output of the classifier in the current path is specified. Therefore, the confidence obtained in this output is used for a better decision in choosing the future path or exit the network. And unlike some methods [35], we do not predict this confidence that may cause an error.

Table 2 shows a comparison of existing work in the field of cost-sensitive image classification. To estimate the correctness of the classification, we use the criterion presented in [14]. We use a smart selection of outputs instead of using the threshold. The details of our selector algorithm are explained in Section 3.3.

Model	Advantages	Disadvantages
Dynamic pruning using pruning nodes [36], or pruning channels [16, 17]	speed up the normal CNN	unable change speed or accuracy at runtime
Decision making with threshold [14, 8]	change speed or accuracy at runtime easily	only could be applied to the cascading model. may reduce the accuracy of the decision
Classifiers as decision-makers in specified structure [35, 37]	select path in network more accurate	just used in specified structure. time consuming. may increase error.
Classifiers as decision-makers in any structure [15]	work in any structure	time consuming. may increase error.

Table 2. Cost-sensitive methods comparison

3 PROPOSED MODEL

3.1 Time-Sensitive Adaptive Classifier Structure

Our model proposes to use a dynamic classifier selection technique. The paper [39] states that the dynamic classifier selection is composed of three phases:

1. Classifiers generation,
2. Selection, and
3. Fusion.

Our proposed model combines two latter phases, so it consists of two main parts, the Classifiers generation and the Selector (a combination of selection and fusion). The Classifiers are the outputs of different layers of CNN, which are generated at different times from the start of the network execution. The Selector is a reinforcement learning agent that selects some of these outputs to achieve the best performance, taking into account the time budget.

Figure 2 shows the general structure of the model and the connection between these two parts.

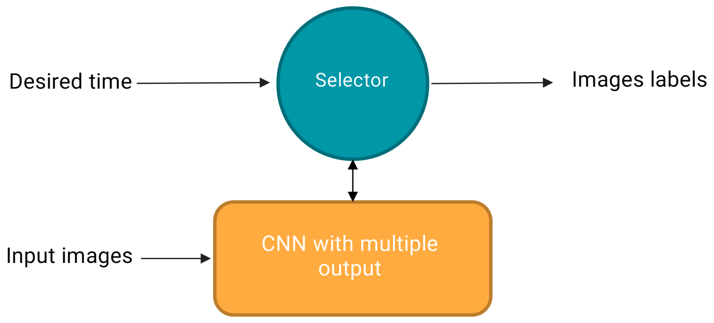


Figure 2. Structure of the proposed method: A model based on reinforcement learning is used for the selector part, and a deep convolutional neural network with several outputs from the middle layers is used for the classifier part

The goal of our cost-sensitive problem is to reduce the total cost of the classification, for any given time budget (t'). The total cost of the model ($Cost_{total}(t')$) is shown as:

$$Cost_{total}(t') = Cost_{error}(t') + Cost_{evaluation\text{-}time}(t') \tag{1}$$

where $Cost_{error}(t')$ is the classifier error cost that is shown in Equation (2) and $Cost_{evaluation\text{-}time}(t')$ is the time of running the algorithm on a set of data that is shown in Equation (3).

$$Cost_{error}(t') = \sum_{i=1}^{N} |W'(x_i) - W(x_i|t')| \tag{2}$$

$$Cost_{evaluation\text{-}time}(t') = \sum_{i=1}^{N} T(x_i|t') + T_{selector}(t') \tag{3}$$

In these formulas, x_i is the input image, N is the number of input images, $W(x_i|t')$ is the output label of the model given time budget t', and $W'(x_i)$ is the true label of image x_i, $T(x_i|t')$ is the CNN processing time for input image x_i given time budget t', and $T_{selector}(t')$ is processing time of our selector module given time budget t'. At runtime, the selector easily accesses a Q-table and selects the action with the highest value, so its processing time could be ignored. CNN processing time is calculated based on the number of operations performed in the network layers. More precisely, we show $T(x_i|t')$ as follows:

$$T(x_i|t') \simeq \sum_{l=1}^{L} P_l(x_i|t').n_{l-1}.s_l^2.n_l.m_l^2, \tag{4}$$

$$P_l(x_i|t') = \begin{cases} 1, & l^{\text{th}} \text{ layer processed,} \\ 0, & l^{\text{th}} \text{ layer not processed,} \end{cases} \tag{5}$$

where L is the number of all layers in the CNN, $P_l(x_i|t')$ is the coefficient that determines whether each layer is processed or not, n_{l-1} is the number of the input channels of the l^{th} layer, n_l is the number of output channels (filters), s_l is the spatial size of the filter, and m_l is the spatial size of the output channels. In a regular CNN, all layers are processed, so $P_l(x_i|t')$ is equal to 1 for all layers. But in our proposed method, we intend to prevent some layers from being processed. The selector accomplishes this goal using a reinforcement learning method. If the input image can be correctly recognized by the outputs of the first layers, it prevents the execution of the final layers of the network. So $P_l(x_i|t')$ will be equal to zero for the final layers of the network if the input image is easy.

According to the expressed formulas, the time complexity of the final model could be written as:

$$O\left(\sum_{l=1}^{L} P_l.n_{l-1}.s^2.n_l.m_l^2\right). \tag{6}$$

Here, P_l $(0 \leq P_l \leq 1)$ is the probability of using each layer of the model. Our method causes P_l to be less than 1 for a number of layers, as described in Section 3.3.

In the following sections, we illustrate two main parts of our proposed method named time-sensitive adaptive classifier.

3.2 Classifiers Generation

As stated in the previous section, the classifiers generation is the first phase of the dynamic classifier selection technique. We need a range of classifiers, some with faster output but less accurate, and some with slower output but more accurate. For this purpose, a deep convolutional neural network (CNN) with several outputs in the middle layers is designed. From the popular and state-of-the-art CNNs such as Googlenet [5], Resnet-152 [7], and VGG network [26], we chose the VGG network. Because this network has fewer layers and is faster than the rest, and also our problem is a two-class problem, so it does not need a complex structure for high-accuracy classification. There are different types of VGG network, and we chose the lightest and fastest one called vgg-f. For the need of the final model, we added three more outputs to the base network.

The designed network, which has four outputs, is shown in Figure 3. The output layers are shown with a red rectangle and are named softmax1, softmax2, softmax3, softmax4. In the convolutional and fully connected layers, we show the characteristic of each layer by "$Convs * s * n$" and "$FCs * s * n$" respectively, where $s * s$ is the size of the filter and n is the number of the input channel. The processing time required to produce each output is the sum of the processing times of the layers on the path to it. Therefore, to achieve the softmax1 output, the convolutional layer, the pooling layer, the fully connected layer, and finally, the softmax1 layer must be processed, so according to Figure 3, the processing time will be $t_1 + t_3$. Normally to reach the softmax3 output, the processing time will be $t_1 + t_2 + t_4 + t_7$. However, if softmax1 output is already generated on the same image, the output of the first layer of pooling is ready, so there is no need to spend time t_1. In this case, the processing time for the softmax3 output will be $t_2 + t_4 + t_7$.

Different network outputs produce their results with different accuracies at different times. The softmax1 output has the highest processing speed and the lowest accuracy, and the softmax4 output has the lowest processing speed and the highest accuracy.

To train our network, we use fine-tuning of the vgg-f network [26]. The vgg-f network has been trained on the ImageNet dataset [33] containing 1 000 classes. We train each of the four outputs separately. To fine-tune the network, first, remove the last fully connected layer before softmax4 from the vgg-f network, and replace it with a fully connected layer with two output nodes for both normal and adult classes. To train only the newly inserted layer, we keep the weights of all the previous layers constant, then train the network with our dataset. The fine-tuning method is also used to train the branch layers. For example, to train the fully connected layer before softmax 1, the fully connected layer with two output nodes is connected to the end of the first pooling layer, as shown in Figure 3. Then, by keeping the weight of the main network layers constant, this new layer is trained with our training dataset. The same goes for training the other two branches.

Before training or testing, we resized all the images to 224×224. We also used the data augmentation method to improve the training of the neural network. So

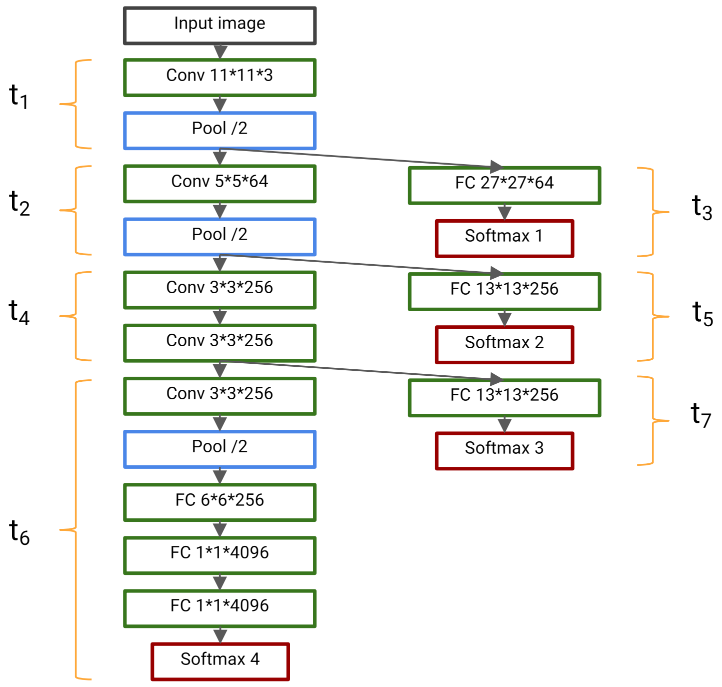


Figure 3. Designed CNN as a classifier with multiple outputs: the green rectangles show the convolutional and fully-connected layers, the blue rectangles represent the pooling layers, and the red rectangles represent the softmax (Network output) layers. $t_1, t_2, \ldots, t_7$ indicate the processing time of each part of the network.

each time the images were called during the training, a slight shift in image cropping, a change in image brightness, and image flipping has been applied.

3.3 Selector

The second phase of a dynamic classifier selection technique is the design of the selector that task is to select and fuse the outputs. After training our CNN, a selector is required to select and combine the best outputs that can get the best result in the shortest time for each image. In other words, for simpler images, output with less processing time should be selected, and for more difficult images, more complex outputs with higher accuracy should be selected. The selector should also be able to keep the average processing time of the images within a specified time budget.

Among the reinforcement learning (RL) methods, Q-learning has been selected. By embedding the accuracy and the time-cost to the reward of the RL method, we lead the RL agent to select the outputs in which the answer with the highest accuracy and the lowest cost will be achieved. At execution time, when the RL agent selects an output, the CNN executes the path to that output, then the RL agent observing the output decides whether to select another output or terminate the process and produce a resulting label. The agent observes and considers both the selected classifier output and its confidence about that output.

The degree of confidence in the output is obtained using the softmax layer output. The softmax formula produces a number between zero and one for each class so that the sum of all the probabilities is equal to one. This number represents the score that determines how much each input belongs to the class:

$$P_i = \frac{e^{s_i}}{\sum_{c=1}^{N} e^{s_c}} \tag{7}$$

where P_i is the probability of each output, s_i is the score value of class i, and N is the number of classes.

The probability given to each class is considered as the network's confidence in the output of that class. Therefore, if this confidence is high, the path selected in CNN has achieved the desired result, and the selector can report the final selected class. But, if this probability, for both classes is close to 0.5, it indicates that the classifier is not sure of the correctness of its output, so the selector selects another output of the neural network to get a more accurate result.

The overall structure of the reinforcement learning algorithm and a number of its states are shown in Figure 4. To further explain our reinforcement learning model, action, state, and reward for the designed model are described below.

Action: In the beginning, the agent can select each of the CNN outputs and process the path leading to that output. In the next step, if more processing is required, the agent can choose another output among the unselected outputs. For example, action $a1$ means to select the softmax1 output. After performing the action $a1$, we reach one of the states related to this output according to the resulting confidence. Also, in each state, the agent can choose one of the normal and adult classes and then go to the final state, which itself is defined as an action. Therefore, according to the confidence obtained at each output, the agent decides to select one of the classes or to continue processing a new path in the neural network.

State: Different states are defined for the proposed model. There is a start and finish state displayed by red rectangles in Figure 4. The two states before the finish state represent the number of model classes (normal and adult). After leaving each of the middle states, the agent (to complete the processing) could decide to choose one of these two classes. Other states also specify which output had been selected in the neural network, and what classes with what confidence is selected. Since we have only two classes in this case, if the confidence level for the output of the normal class is above 0.5, the normal class is selected, otherwise, the adult class is selected. In Figure 4, $S1 = R1$ shows the selection of the normal class at the softmax1 output with confidence above 0.5, and $S1 = R2$

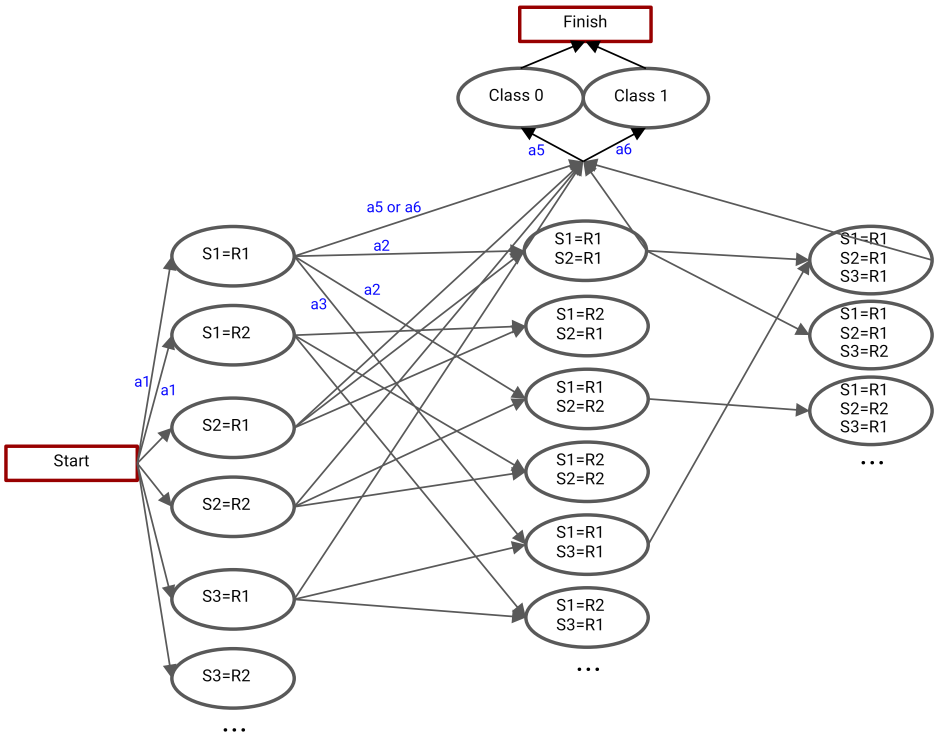


Figure 4. The general structure of the reinforcement learning algorithm in a simple mode: The start and finish states are displayed with red rectangles, and middle states are displayed with ellipses. The actions that shown by arrows named as $a1$, $a2$, $a3$, $a4$ indicate the selection of each of the CNN outputs, and $a5$, $a6$ specify the actions of selecting each of the final classes. S1 represents softmax1 output, S2 represents softmax2 output, and so on. $R1$ and $R2$ represent two value interval for confidence.

shows the selection of the adult class with confidence above 0.5 (equal to the confidence below 0.5 at the output of the normal class). That is the easiest way to quantize the confidence value into two intervals. In Section 4, this quantization is done with more intervals, and its effect on the achieved accuracy is shown.

Reward: In the Q-learning algorithm, the value of each state and action is updated according to the current reward and the highest reward obtained in the next state. Our presented reward depends on two factors: the calculation time of the processed CNN layers, and the misclassification error. Q-table values are updated using the following equation [41]:

$$Q_{new}(s_i, a_i) \leftarrow Q_{old}(s_i, a_i) + \beta.(Reward_i + \gamma.max_a Q(s_{i+1}, a) - Q_{old}(s_i, a_i)) \quad (8)$$

where $Q(s_i, a_i)$ is the value of state s_i by performing action a_i, β is learning rate, γ is the discount factor, $max_a Q(s_{i+1}, a)$ is the maximum value that can be obtained from state s_{i+1}, and $Reward_i$ is the reward received by performing action a_i in state s_i. According to Equation (1), to simultaneously consider the accuracy and time cost, the $Reward_i$ is determined as follows:

$$Reward_i = Accuracy_i - \alpha.TimeCost_i \tag{9}$$

where $Accuracy_i$ is achieved in final action based on whether the classification result is correct or not. If the class assigned to each instance in pre-finish states is correct, $Accuracy_i$ is set with a positive constant value (we used 10) and otherwise with a negative constant value (we used -10). $TimeCost_i$ for each action indicates the extra time spent to get the output in the CNN up to this point. The α parameter is a factor that determines the balance between time and accuracy. If we look for a faster result, this value is increased, so the results that are generated over a longer time will be more penalized.

By changing the parameter α, the model is trained for different time constraints. So, for various time budgets, we set the α to various values and we will have different Q tables.

According to the times $t_1, t_2, \ldots, t_7$ shown for different parts of the network in Figure 3, Equation (4) can be simplified for our model as follows:

$$T(x_i|t') \simeq \sum_{l=1}^{7} P_l(x_i|t').t_l \tag{10}$$

where $T(x_i|t')$ is the CNN processing time for input image x_i given time budget t', l indicates the l^{th} part of the CNN, and $P_l(x_i|t')$ is the phrase that determines whether each part is processed or not. By penalizing longer times using Equation (9), more layers will be neglected and processing time $T(x_i|t')$ will be reduced.

4 EXPERIMENT RESULTS AND ANALYSIS

4.1 Dataset Description

We created a dataset containing 18 000 different images to train and test the model. We wanted the database to have diverse and challenging images, so we collected many samples from public social networks or image sharing sites. Social network images are often produced by non-professional users, and many of them do not have proper lighting or quality. Dataset images were selected to include both simple and hard images. In the adult class, 38 % are simple images, but the rest of the images have various challenges. About 22 % of the images have poor illumination, in 40 % of the images the important parts of the image are covered with clothes or with text on the image, 20 % of the images are not naked and can only be detected by body positions. The dataset can be downloaded from [40].

Figure 5 shows some samples from this dataset. The upper row shows normal images; the three images on the left are simple, but the two images on the right are more difficult to recognize because of their large area of skin and high similarity to adult images. The lower row shows adult images; the three images on the left side include the naked bodies and are easy to recognize. But the two images on the right side are difficult to recognize, these two images are taken outdoors, and the peoples in them are not completely naked, but they are adult images because of showing certain parts of the body.

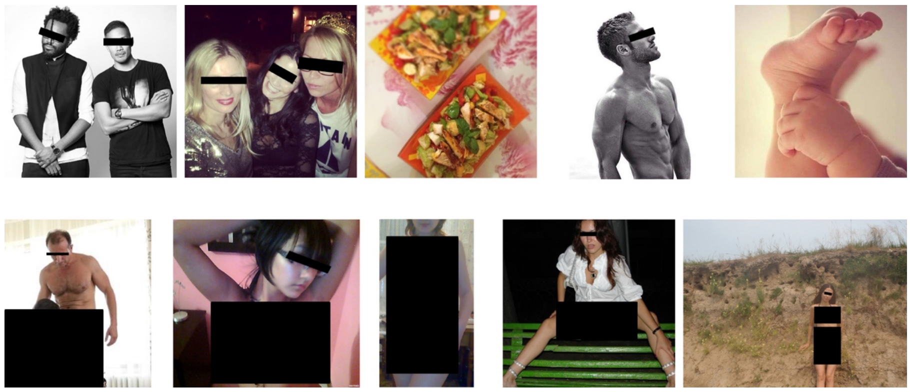


Figure 5. Sample images of our dataset: The first row shows the images of the normal class, and the second row shows the adult class

The entire dataset is divided into three parts. The neural network is trained using the first part of the dataset, the second part of the dataset is used to train the selector, And finally, the third part (test data) is used to report the accuracy of the network.

4.2 Experimental Setup

To train and test the deep neural network and other parts of the model, a device with 2.7 GHz Intel Core i5 processor and 8 GB RAM has been used. MatConvNet software package [38] has been used for training and testing the CNN.

Our CNN is trained using data from the first part of the dataset, then the results are reported on the images of the third part (test data). The values obtained for the various outputs of the CNN are shown in Table 3. The accuracy of the first output (softmax1) is about 77 % and the accuracy of the last output (softmax4) is about 93 %. At the output of the last layer, a 16 % increase in accuracy has been achieved, but the processing time has increased almost sevenfold. The times related to different parts of the CNN are shown with $t_1, t_2, \ldots, t_7$ according to Figure 3.

The reported processing time of each output is the average processing time for all images. And the accuracy is presented according to the following formula using the average accuracy of different classes.

$$Accuracy = \frac{\frac{TP}{totalNumberP} + \frac{TN}{totalNumberN}}{2} * 100 \tag{11}$$

where TP is the number of correctly classified images of the adult class and *totalNumberP* is the total number of images of the adult class, similarly, TN is the number of correctly classified images of the normal class and *totalNumberN* is the total number of images of the normal class. Using this formula, the accuracy of the classification does not change according to the imbalance between the number of samples in each class.

Output name	Accuracy (percent)	Time (ms)	Time according to Figure 3
Softmax1	77.13	4.65	$t_1 + t_3$
Softmax2	82.1	12.03	$t_1 + t_2 + t_5$
Softmax3	85.07	17.27	$t_1 + t_2 + t_4 + t_7$
Softmax4	92.53	33.36	$t_1 + t_2 + t_4 + t_6$

Table 3. Accuracy and the processing time of the outputs of the proposed method's CNN

4.3 Quantizing the Confidence into Four Intervals

As explained in the previous sections, to determine the different states in the selector, it is necessary to quantize the numerical values of the softmax output. We also want to define different states for different classes. Of course, the output nodes of the last layer of the neural network have the same number as classes, so in our case, it has two output nodes. Since the sum of the values of these two outputs is equal to one, the specified classes can be extracted using one of the output nodes. Assuming we display the output range of the normal output node with $[0, 1]$, if the output value is in the range $(0.5, 1]$, we consider it as the normal class, and if it is in the range $[0, 0.5]$, we consider it as the adult class.

In this section, the output range is quantized into four intervals. Therefore, the quantization is considered as follows:

$$R1 = [0, a], R2 = (a, 0.5], R3 = (0.5, a'], R4 = (a', 1] \tag{12}$$

where $R1, R2, R3, R4$ are different intervals. To make this intervals balanced, we set $a' = 1 - a$. Thus, if the output is in the range of $R1$ or $R4$, the confidence of the output is high, otherwise, the confidence is low. For example, if the output for the normal class is in the range $R4 = (a', 1]$, it indicates that the normal class is selected with high confidence, and if it is in the range $R3 = (0.5, a']$, the normal class is selected with low confidence.

Reinforcement learning models are trained using the second part of the database (as a training set for the selector) and different models are obtained for different values of a. The results for these models on the train set as well as the test set are shown in Figure 6.

In each model, by changing the α parameter in Equation (12), the resulting accuracies and their corresponding processing times are obtained.

According to Figure 6, the corresponding line to the all presented models is above the line of the base network, so the given models performed better at the same time budget. Models with a greater area under the curve are generally more accurate. For a better comparison, we have shown the area under the curve in the accuracy vs. time budget chart (AUC) for different models in Table 4. According to the table, the best result is 279.5 that is obtained on the test set for the parameter $a = 0.1$.

Model	Area Under the Curve
base net	201.3
$a = 0.05$	268.5
$a = 0.1$	**279.5**
$a = 0.2$	274.1
$a = 0.3$	260.6
$a = 0.4$	237.8

Table 4. Area under the curve for models with four quantization intervals with different parameter of a

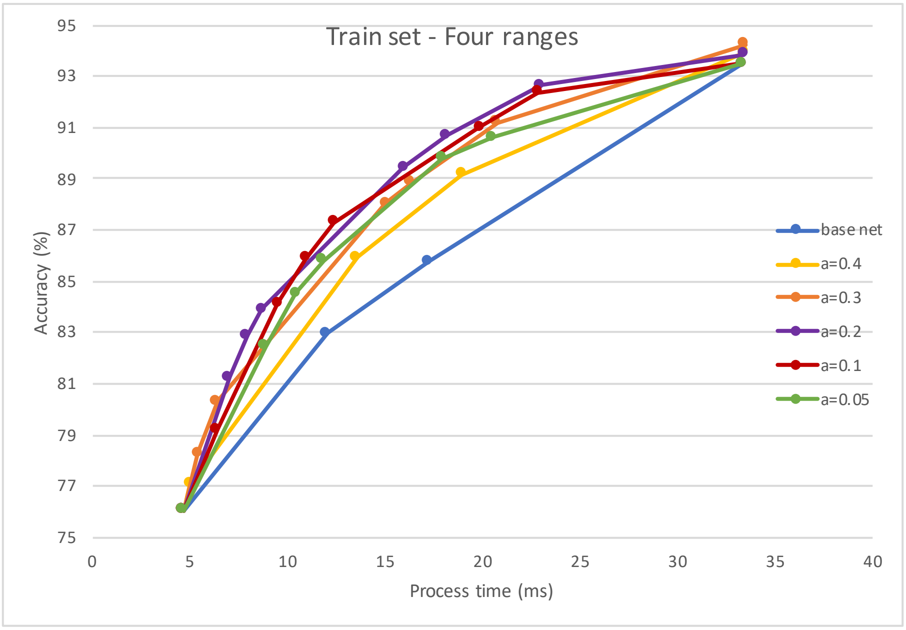
When only the accuracy is important and the penalty for image processing time is low, the selector uses all network outputs to select the appropriate class. Since most of the processing path is common between the outputs of the base network, using all outputs does not significantly increase processing time. Although the use of all outputs increased the accuracy on the training data, there is no significant increase in the accuracy on the test data. This is because most of the image processing is done in the convolution layers, our three branches consist only of fully connected layers.

4.4 Different Number of Quantization Intervals

In this section, we have increased the number of quantization intervals and measured its effect on accuracy. Below our different quantization intervals are shown.

$$3part\ model = \{R1 = [0, 0.3], R2 = (0.3, 0.7], R3 = (0.7, 1]\}, \tag{13}$$

$$4part\ model = \{R1 = [0, 0.2], R2 = (0.2, 0.5], R3 = (0.5, 0.8], R4 = (0.8, 1]\}, \tag{14}$$



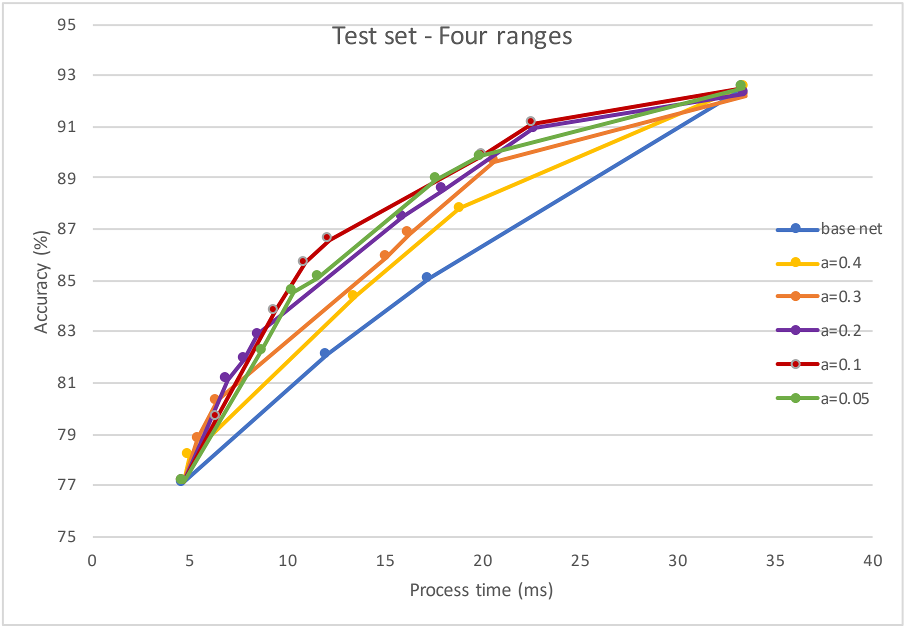


Figure 6. Accuracy versus time budget for models with four quantization intervals: The blue line shows the connection between four points related to the resulting points of the four outputs of the base network. For different values of a, the results are shown by a different color. The top chart shows the results on the train set, and the bottom chart shows the results on the test set.

$$\begin{aligned}\textit{6part model} = \{&R1 = [0, 0.1], R2 = (0.1, 0.2], R3 = (0.2, 0.5],\\ &R4 = (0.5, 0.8], R4 = (0.8, 0.9], R4 = (0.9, 1]\},\end{aligned} \tag{15}$$

$$\begin{aligned}\textit{8part model} = \{&R1 = [0, 0.1], R2 = (0.1, 0.2], R3 = (0.2, 0.3],\\ &R4 = (0.3, 0.5], R5 = (0.5, 0.7], R6 = (0.7, 0.8],\\ &R7 = (0.8, 0.9], R8 = (0.9, 1]\},\end{aligned} \tag{16}$$

$$\begin{aligned}\textit{10part model} = \{&R1 = [0, 0.1], R2 = (0.1, 0.2], R3 = (0.2, 0.3], R4 = (0.3, 0.4],\\ &R5 = (0.4, 0.5], R6 = (0.5, 0.6], R7 = (0.6, 0.7],\\ &R8 = (0.7, 0.8], R9 = (0.8, 0.9], R10 = (0.9, 1]\}\end{aligned} \tag{17}$$

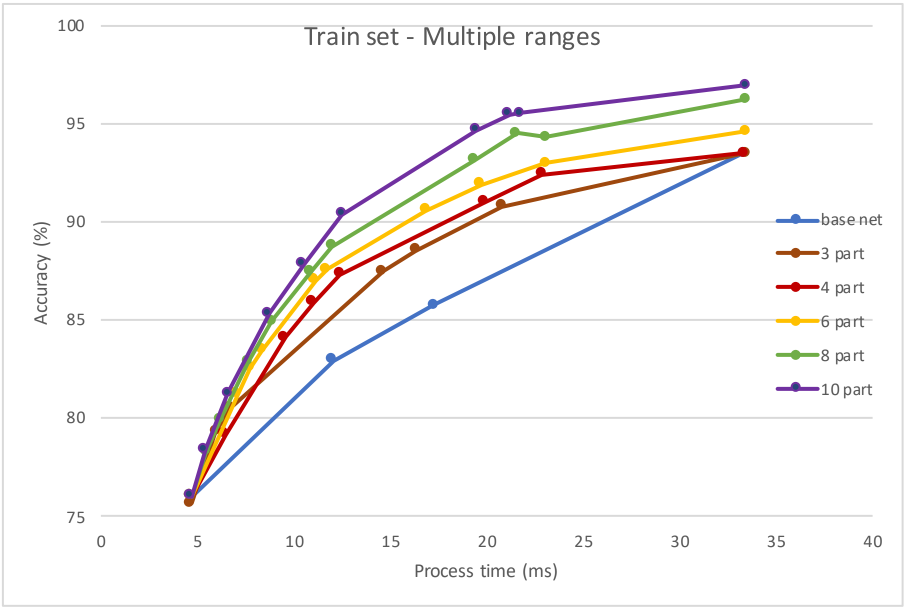
where 3*part model*, 4*part model*, 6*part model*, 8*part model*, 10*part model* are five models that are trained with different quantization intervals. For each of these models, different divider numbers have been tested, and the value with the highest accuracy has been placed in the above equation as the selected model. The result on the test and train set for all models is shown in Figure 7. We notice that most of the points of the proposed models are above the resulting line of the base network. So our models perform better in these conditions.

In the case of 3 quantization interval, in about five milliseconds, the model's accuracy is lower than the first output of the CNN, because in this model only when the output confidence is above 70 %, the class is specified; otherwise, the output value is in the range $R2$, which does not show a specific class and only indicates that the classifier is not confident about its output. As the number of quantization intervals increases, the accuracy of the training data increases, but after increasing the intervals to more than four parts, due to the overfitting of the model to the training data, the accuracy of the test data decreases. As in the previous section, it was found that using all the outputs of the neural network in the selector does not significantly increase the accuracy on the test data, and this shows that the output of the last layer of the base neural network contains the information of the other outputs.

For a more accurate comparison of the models, we also have calculated the area under the curve of accuracy vs. time budget chart (AUC), as shown in Table 5. According to this table, the best number of quantization intervals for the model is four part quantization. The worst model is the 10 part, but even this model has performed better than the base network.

4.5 Final Model Evaluation

To verify that the final selected model performs better than the base network for all different time budgets, we calculated the precision, recall, and F1-score values for a number of different time budgets, and showed them in Table 6. In the least and most time-consuming cases, the results of our model are equal to the base network. But



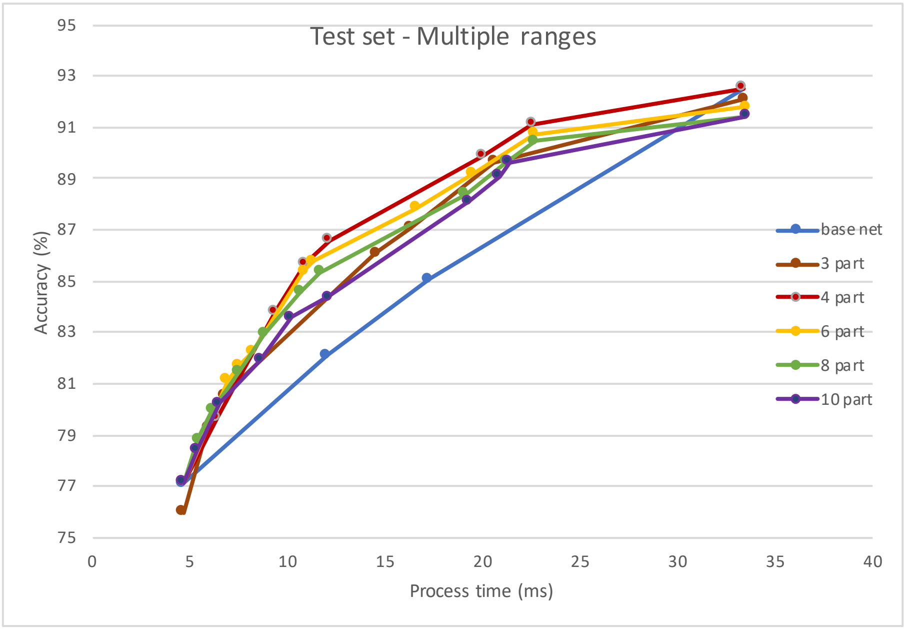


Figure 7. Accuracy versus time budget for models with different quantization intervals: The blue line shows the connection between four points related to the resulting points of the four outputs of the base network. Models by changing the quantization intervals into 3, 4, 6, 8, and 10 intervals are shown in the figure with different colors. The top chart shows the results on the train set, and the bottom chart shows the results on the test set.

Model	Area Under the Curve
base net	201.3
3 part	252.2
4 part	**279.5**
6 part	269.1
8 part	258.9
10 part	244.8

Table 5. Area under the curve for models with different quantization intervals

for all intermediate time budgets, the results are improved. For example, for 12.03 milliseconds for the base network, the precision, recall, and F1-score are 0.8141, 0.832, and 0.8229, respectively, while for the proposed model, in less time, i.e. 8.76 milliseconds, the performance is better, and the precision, recall, and F1-score are 0.819, 0.8406, and 0.8297, respectively. Therefore, as explained in the previous sections, the combination of network outputs as performed by the proposed model is able to improve the final result, in addition to the ability to produce output at the given time budgets.

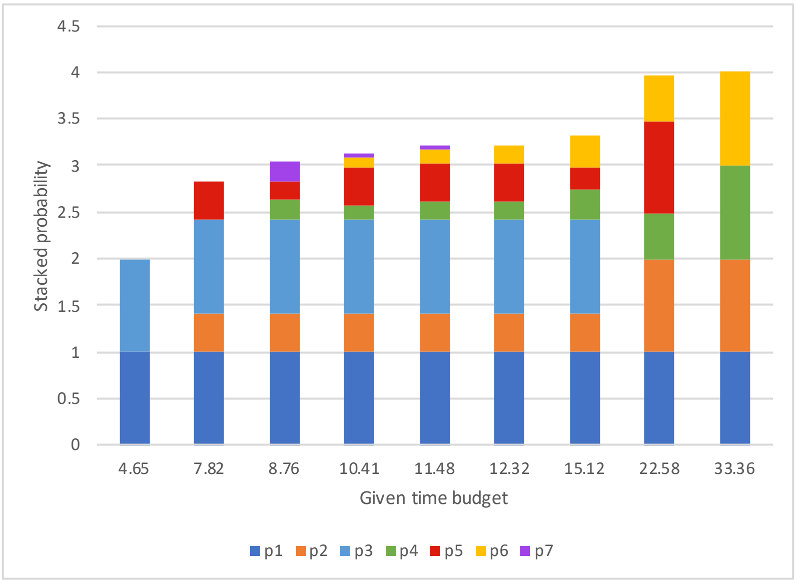
Model	Process Time (ms)	Precision	Recall	F1-score
base net	4.65	0.7602	0.7927	0.7761
	12.03	0.8141	0.832	0.8229
	17.27	0.8465	0.8567	0.8516
	33.36	0.9288	0.9213	0.925
our 4 part model	4.65	0.7603	0.793	0.7763
	8.76	0.819	0.8406	0.8297
	10.41	0.8367	0.849	0.8428
	12.32	0.8724	0.8506	0.8614
	16.55	0.8782	0.8876	0.8829
	22.58	0.9127	0.9096	0.9111
	33.36	0.9287	0.9213	0.925

Table 6. F1-score comparison for 4 part model and base model

Reduction of processing time is achieved by reducing the values of $P_l(x_i|t')$ for different parts of the CNN, in Equation (10). The first chart in Figure 8 shows the average of P_l values for different time budgets t' for all images in the test set. The second chart shows the processing time for different time budgets. P_l and t_l corresponds to a part of the CNN in Figure 3.

In Figure 8, P_3, P_5, P_7, P_6 correspond to the parts leading to the output softmax1, softmax2, softmax3, and softmax4 of the CNN, respectively. In the following, we will explain two charts of the Figure 8.

In the top chart, when the time budget is very low, only softmax1 output is used (i.e. $p1 = 1$ and $p3 = 1$). As the budget increases, the probability of using layers with higher computational cost increases. Therefore, the hypothesis of re-



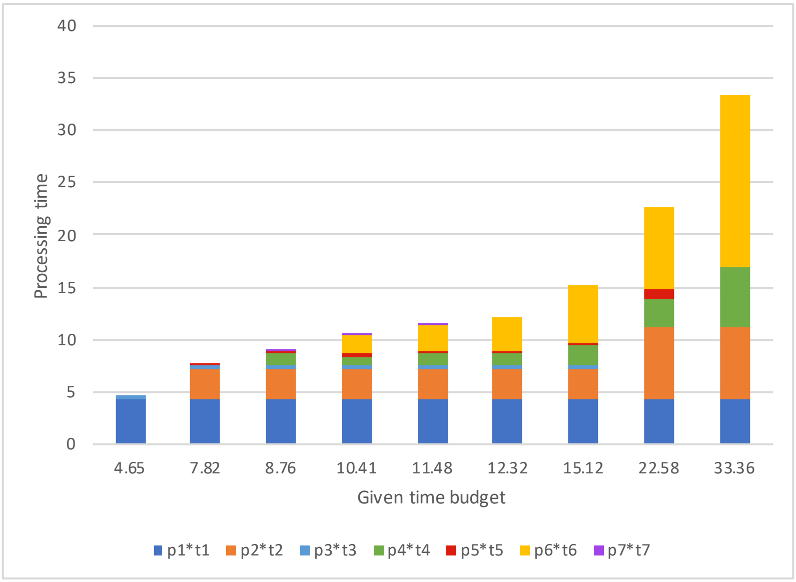


Figure 8. The top chart shows the stacked probability of P_l given different time budgets. The bottom chart shows the stacked time of each part of the CNN (i.e. $P_l * t_l$) given different time budgets.

ducing time complexity by reducing the use of layers (presented in Equation (6)) is verified.

When time budgets are less than 15.12 milliseconds, the first output (P_3) is used, but the sum of probability of using the other outputs ($P_5 + P_7 + P_6$) is less than one. That is, some images are simple and can be correctly classified using only the first output of the CNN.

The bottom chart shows that the outputs P_3, P_5, and $P7$ consume very little processing time and most of the processing is related to the middle layers. When the maximum time budget is given, the selector does not use outputs from the earlier parts of the network and only uses the last output P_6. Because the outputs are not completely independent, the latest output contains information about previous outputs, and the use of earlier outputs does not improve the result.

By increasing the time budget, the selector has used the layers with more time cost, which are also more accurate. But this use is not linear and the selector tries to use the best combination for different given budgets. For example, the selector has used softmax3 (correspond to P_7) very little, and by increasing the time budget, the selector has increased the share of softmax4 (correspond to P_6).

5 CONCLUSIONS AND FUTURE WORKS

This paper presents a method to adaptively resolve the trade-off between accuracy and time budget for adult image classification. We used a dynamic classifier selection technique. First, we created a CNN with three outputs from the middle layers, then we trained a selector using the Q-learning method to select and combine the best CNN outputs for each image based on the available time budget. The results confirm that our selector can increase the accuracy of the classifier by using the appropriate combination of network outputs. The final model also has the ability to generate results for different time budgets.

In the future, we intend to automize the quantization of the outputs of the softmax layers using a linear classifier. So, the error of manually quantizing the ranges can be eliminated. We also plan to extend this method for classifying adult videos. By considering the correlation of consecutive frames and using the appropriate selector, we could reduce the processing time of the entire video.

REFERENCES

[1] Community Guidelines, Instagram Help Center. Available at: `https://help.instagram.com/477434105621119`, July 17, 2020.

[2] Community Standards, Facebook. Available at: `https://www.facebook.com/communitystandards/objectionable_content`, July 17, 2020.

[3] Wang, L.—Zhang, J.—Tian, Q.—Li, C.—Zhuo, L.: Porn Streamer Recognition in Live Video Streaming via Attention-Gated Multimodal Deep Features. IEEE

Transactions on Circuits and Systems for Video Technology, Vol. 30, 2020, No. 12, pp. 4876–4886, doi: 10.1109/TCSVT.2019.2958871.

[4] Lykousas, N.—Gómez, V.—Patsakis, C.: Adult Content in Social Live Streaming Services: Characterizing Deviant Users and Relationships. 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2018, pp. 375–382, doi: 10.1109/ASONAM.2018.8508246.

[5] Liu, H.—Simonyan, K.—Yang, Y.: DARTS: Differentiable Architecture Search. 7th International Conference on Learning Representations (ICLR 2019), 2019, pp. 1–13.

[6] Simonyan, K.—Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. 3rd International Conference on Learning Representations (ICLR 2015). ArXiv Preprint ArXiv:1409.1556v6, 2015.

[7] He, K.—Zhang, X.—Ren, S.—Sun, J.: Deep Residual Learning for Image Recognition. Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[8] Leroux, S.—Bohez, S.—De Coninck, E.—Verbelen, T.—Vankeirsbilck, B.—Simoens, P.—Dhoedt, B.: The Cascading Neural Network: Building the Internet of Smart Things. Knowledge and Information Systems, Vol. 52, 2017, No. 3, pp. 791–814, doi: 10.1007/s10115-017-1029-1.

[9] Karayev, S.—Baumgartner, T.—Fritz, M.—Darrell, T.: Timely Object Recognition. In: Pereira, F., Burges, C. J. C., Bottou, L., Weinberger, K. Q. (Eds.): Advances in Neural Information Processing Systems 25 (NIPS 2012), 2012, pp. 890–898, `http://papers.nips.cc/paper/4712-timely-object-recognition`.

[10] Iandola, F. N.—Han, S.—Moskewicz, M. W.—Ashraf, K.—Dally, W. J.—Keutzer, K.: SqueezeNet: AlexNet-Level Accuracy with 50× Fewer Parameters and < 0.5 MB Model Size. 2016, pp. 1–13, `http://arxiv.org/abs/1602.07360`.

[11] Howard, A. G.—Zhu, M.—Chen, B.—Kalenichenko, D.—Wang, W.—Weyand, T.—Andreetto, M.—Adam, H.: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. 2017, `http://arxiv.org/abs/1704.04861`.

[12] Zhang, X.—Zhou, X.—Lin, M.—Sun, J.: ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 6848–6856, doi: 10.1109/CVPR.2018.00716.

[13] Ma, N.—Zhang, X.—Zheng, H.-T.—Sun, J.: ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (Eds.): Computer Vision – ECCV 2018. Springer, Cham, Lecture Notes in Computer Science, Vol. 11218, 2018, pp. 122–138, doi: 10.1007/978-3-030-01264-9_8.

[14] Berestizshevsky, K.—Even, G.: Dynamically Sacrificing Accuracy for Reduced Computation: Cascaded Inference Based on Softmax Confidence. In: Tetko, I. V., Kůrková, V., Karpov, P., Theis, F. (Eds.): Artificial Neural Networks and Machine Learning – ICANN 2019: Deep Learning. Springer, Cham, Lecture Notes in Computer Science, Vol. 11728, 2019, pp. 306–320, 2019, doi: 10.1007/978-3-030-30484-3_26.

[15] Liu, L.—Deng, J.: Dynamic Deep Neural Networks: Optimizing Accuracy-Efficiency Trade-Offs by Selective Execution. 32nd AAAI Conference on Artificial Intelligence (AAAI-18), 2018, pp. 3675–3682.

[16] Lin, J.—Rao, Y.—Lu, J.—Zhou, J.: Runtime Neural Pruning. In: Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 30 (NIPS 2017), 2017, pp. 2182–2192.

[17] Rao, Y.—Lu, J.—Lin, J.—Zhou, J.: Runtime Network Routing for Efficient Image Classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 41, 2019, No. 10, pp. 2291–2304, doi: 10.1109/TPAMI.2018.2878258.

[18] Zheng, H.—Daoudi, M.—Jedynak, B.: Blocking Adult Images Based on Statistical Skin Detection. In Electronic Letters on Computer Vision and Image Analysis, Vol. 4, 2004, No. 2, 14 pp., doi: 10.5565/rev/elcvia.78.

[19] Zheng, H.—Liu, H.—Daoudi, M.: Blocking Objectionable Images: Adult Images and Harmful Symbols. 2004 IEEE International Conference on Multimedia and Expo (ICME), Vol. 2, 2004, pp. 1223–1226, doi: 10.1109/icme.2004.1394442.

[20] Lee, J. S.—Kuo, Y. M.—Chung, P. C.: The Adult Image Identification Based on Online Sampling. Proceedings of the 2006 IEEE International Joint Conference on Neural Network, 2006, pp. 2566–2571, doi: 10.1109/IJCNN.2006.247111.

[21] Zheng, Q. F.—Zhang, M. J.—Wang, W. Q.: A Hybrid Approach to Detect Adult Web Images. In: Aizawa, K., Nakamura, Y., Satoh, S. (Eds.): Advances in Multimedia Information Processing – PCM 2004. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3332, 2004, pp. 609–616, doi: 10.1007/978-3-540-30542-2_75.

[22] Zhu, H.—Zhou, S.—Wang, J.—Yin, Z.: An Algorithm of Pornographic Image Detection. Proceedings of the 4th International Conference on Image and Graphics (ICIG 2007), 2007, pp. 801–804, doi: 10.1109/ICIG.2007.29.

[23] Zheng, Q. F.—Zeng, W.—Wang, W. Q.—Gao, W.: Shape-Based Adult Image Detection. International Journal of Image and Graphics, Vol. 6, 2006, No. 1, pp. 115–124, doi: 10.1142/S0219467806002082.

[24] Lopes, A. P. B.—de Avila, S. E. F.—Peixoto, A. N. A.—Oliveira, R. S.—De A. Araújo, A.: A Bag-of-Features Approach Based on Hue-SIFT Descriptor for Nude Detection. 2009 17th European Signal Processing Conference (EUSIPCO), Glasgow, UK, 2009, pp. 1552–1556, `https://ieeexplore.ieee.org/abstract/document/7077625/`.

[25] Deselaers, T.—Pimenidis, L.—Ney, H.: Bag-of-Visual-Words Models for Adult Image Classification and Filtering. 2008 19th International Conference on Pattern Recognition, 2008, pp. 1–4, doi: 10.1109/ICPR.2008.4761366.

[26] Chatfield, K.—Simonyan, K.—Vedaldi, A.—Zisserman, A.: Return of the Devil in the Details: Delving Deep into Convolutional Nets. Proceedings of the British Machine Vision Conference 2014, 2014, pp. 6.1–6.12, doi: 10.5244/C.28.6.

[27] Moustafa, M.: Applying Deep Learning to Classify Pornographic Images and Videos. Proceedings of the 7th Pacific-Rim Symposium on Image and Video Technology (PSIVT 2015), Auckland, New Zealand, 2015, `http://arxiv.org/abs/1511.08899`.

[28] Krizhevsky, A.—Sutskever, I.—Hinton, G. E.: Imagenet Classification with Deep Convolutional Neural Networks. In: Pereira, F., Burges, C. J. C., Bottou, L., Weinberger, K. Q. (Eds.): Advances in Neural Information Processing Systems 25 (NIPS 2012), 2012, pp. 1097–1105, `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks`.

[29] Szegedy, C.—Liu, W.—Jia, Y.—Sermanet, P.—Reed, S.—Anguelov, D.—Erhan, D.—Vanhoucke, V.—Rabinovich, A.: Going Deeper with Convolutions. Proceedings of the 2015 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

[30] Nian, F.—Li, T.—Wang, Y.—Xu, M.—Wu, J.: Pornographic Image Detection Utilizing Deep Convolutional Neural Networks. Neurocomputing, Vol. 210, 2016, pp. 283–293, doi: 10.1016/j.neucom.2015.09.135.

[31] Shen, R.—Zou, F.—Song, J.—Yan, K.—Zhou, K.: EFUI: An Ensemble Framework Using Uncertain Inference for Pornographic Image Recognition. Neurocomputing, Vol. 322, 2018, pp. 166–176, doi: 10.1016/j.neucom.2018.08.080.

[32] Vitorino, P.—Avila, S.—Perez, M.—Rocha, A.: Leveraging Deep Neural Networks to Fight Child Pornography in the Age of Social Media. Journal of Visual Communication and Image Representation, Vol. 50, 2018, pp. 303–313, doi: 10.1016/j.jvcir.2017.12.005.

[33] Deng, J.—Dong, W.—Socher, R.—Li, L. J.—Li, K.—Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[34] Cheng, F.—Wang, S. L.—Wang, X. Z.—Liew, A. W. C.—Liu, G. S.: A Global and Local Context Integration DCNN for Adult Image Classification. Pattern Recognition, Vol. 96, 2019, Art. No. 106983, doi: 10.1016/j.patcog.2019.106983.

[35] Bolukbasi, T.—Wang, J.—Dekel, O.—Saligrama, V.: Adaptive Neural Networks for Efficient Inference. 34th International Conference on Machine Learning (ICML 2017), PMLR, Vol. 70, 2017, pp. 527–536.

[36] Bengio, E.—Bacon, P.-L.—Pineau, J.—Precup, D.: Conditional Computation in Neural Networks for Faster Models. 2015, `http://arxiv.org/abs/1511.06297`.

[37] Odena, A.—Lawson, D.—Olah, C.: Changing Model Behavior at Test-Time Using Reinforcement Learning. Proceedings of the 5th International Conference on Learning Representations (ICLR 2017), Workshop Track, 2017, pp. 1–6, `https://arxiv.org/pdf/1702.07780.pdf`.

[38] Vedaldi, A.—Lenc, K.: MatConvNet: Convolutional Neural Networks for MATLAB. Proceedings of the 23rd ACM International Conference on Multimedia (MM 2015), 2015, pp. 689–692, doi: 10.1145/2733373.2807412.

[39] Cruz, R. M. O.—Sabourin, R.—Cavalcanti, G. D. C.: Dynamic Classifier Selection: Recent Advances and Perspectives. Information Fusion, Vol. 41, 2018, pp. 195–216, doi: 10.1016/j.inffus.2017.09.010.

[40] Noktedan, A.: Adult Content Dataset. Figshare, 20-Dec-2020, doi: 10.6084/m9.figshare.13456484.v1.

[41] WATKINS, C. J. C. H.—DAYAN, P.: Q-Learning. Machine Learning, Vol. 8, 1992, pp. 279–292, doi: 10.1007/BF00992698.

Mohammad Reza MAZINANI is Ph.D. candidate in computer engineering, Malek Ashtar University of Technology, Iran. He received his M.Sc. degree in artificial intelligence from the Computer Engineering Department, Sharif University of Technology, Iran. His research interests are deep learning, cost-sensitive computing, image and video processing, and pattern recognition.

Seyyed Mojtaba HOSEINI received his B.Sc. degree in electronic engineering from Malek Ashtar University of Technology in 1991. He also received his M.Sc. and Ph.D. degrees in computer architecture engineering from Amirkabir University of Technology in 1995 and 2011, respectively. His research interest is in wireless sensor networks, with an emphasis on target coverage and tracking applications, image and signal processing, and evolutionary computing.

Kourosh DADASHTABAR AHMADI is Chairman of Computer and Artificial Intelligence Research Institute, Head of AI group, and Assistant Professor in Malek Ashtar University of Technology, Iran. He received his M.Sc. in information technology from Tarbiat Modares University in 2007, and Ph.D. in artificial intelligence from Malek Ashtar University of Technology, Iran in 2015. His research interests are information fusion, cyber security, image processing, deep learning, and reinforcement learning.