Tartu University

Faculty of Science and Technology

Institute of Technology

Dogus Karabulut

Neural Networks Based Automatic Content Moderation on Social Media

Master's thesis (30 ECTS)

Robotics and Computer Engineering

Supervisors:

Dr. Cagri Ozcinar

Prof. Gholamreza Anbarjafari

Tartu 2020

Resümee/Abstract

Neural Networks Based Automatic Content Moderation on Social Media

Millions of users produce and consume billions of content on social media. Therefore, human-reviewed content moderation is not achievable in such volume. Automating content moderation is a scalable solution for social media platforms. In this thesis work, we propose a neural networks based automatic content moderation pipeline. Our solution consists of two main parts: the first part that classifies the content into granular content classes and a second part that automatically obfuscates the part of the image that might be inappropriate for the target audience. The proposed solution is cost-efficient in terms of human labour. Our classification network is trained with automatically labelled data using noise-robust techniques. Our automatic obfuscation algorithm uses the information obtained from the classification network and does not require additional annotation or supplementary training. This obfuscation algorithm presents a novel-use case to the state-of-the-art.

CERCS: P170 Computer science, numerical analysis, systems, control; T125 Automation, robotics, control engineering; T111 Imaging, image processing

Keywords: —- Computer vision, Image recognition, Pattern recognition, Convolutional neural networks, Deep learning, Automated content moderation

Neurovõrgul baseeruv automaatne sisu modereerimise lahendus sosiaalvõrgustikele

Miljonid sotsiaalmeediakasutajad loovad ja tarbivad miljardeid ühikuid sisu. Sisu rohkuse tõttu pole inimeste poolt tehtav sisu modereerimine enam saavutatav. Sisu modereerimise automatiseerimine on sotsiaalmeediaplatvormidele skaleeritav lahendus. Selles töös pakume neurovõrgul baseeruvat automaatset sisumoderatsiooni konveierit. Meie lahendus koosneb kahest osast: esimene neist klassifitseerib sisu granuleeritud sisuklassidesse, teine eraldab automaatselt sisu, mis võib sihtgrupile sobimatuks osutuda. Pakutav lahendus on inimressursi mõttes kuluefektiivne. Meie klassifitseerimisvõrgustik on treenitud automaatselt märgistatud infoga, kasutades müratõhusaid tehnikaid. Meie automaatne eraldusalgoritm kasutab klassifikatsioonivõrgustikust saadud infot ja ei vaja mingisugust annotatsiooni ega lisatreeningut. Eraldusalgoritm on uusim ja tipptasemel.

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria) ; T125 Automatiseerimine, robootika, juhtimistehnika; T111 Pilditehnika

Märksõnad: — Arvutinägemine, pildituvastus, mustrituvastus, konvolutsioonilised närvivõrgud, sügav õppimine, automatiseeritud sisu modereerimine

Contents

List of Figures

List of Tables

Abbreviations, constants, definitions

CE - Cross Entrop

CNN - Convolutional Neural Networks

FC - Fully Connected layer

LS - Label Smoothing

NSFW - Not safe for work

RCE - Reverse Cross Entropy

ReLU - Rectified linear activation unit

ResNet - Residual Neural Network

SCE - Symmetric Cross Entropy

SVM - Support Vector Machine

1 Introduction

With the dramatic increase of accessibility of the internet, social media websites have grown into massive content repositories on a diverse spectrum of subjects. Millions of users can use these platforms to share and explore almost anything. This has caused a revolution in the way we produce and consume content. Information is no longer produced by one person as in the past. Now, millions of people produce contents that are later consumed by many others. These vast quantities of users and data have changed the need for content management considering that not all the contents on social media are appropriate for all users, especially children, or suitable for access from particular locations such as workplaces.

1.1 Problem definition

Human-reviewed content moderation of billions of images is not viable in many aspects. Humans cannot achieve an acceptable speed to analyse real-time images, and a notable lag between uploading photo and publishing the post causes a less attractive user-experience. Furthermore, humans can cause inconsistencies in a subjective task such as content classification. Following the same graphic content policy can be challenging with multiple people. Additionally, long term exposure to disturbing content can cause mental health issues in human annotators [6].

Automating content moderation is a scalable solution for social media websites. However, using algorithms for analysing uploaded content has several drawbacks. Unlike humans, algorithms cannot instinctively decide whether the content is appropriate or not. Furthermore, the definition of inappropriate content is a grey area when it comes to suggestive nudity. While some social media websites allow such content, others might restrict it.

1.2 Thesis objective

In this thesis work, we propose a neural networks based automatic content moderation pipeline. Our solution consists of two main parts: the first part that classifies the content into granular content classes and a second part that obfuscates the part of the image that might be inappropriate for the general audience.

Our solution is cost-efficient in terms of human labour. We experimented with novel learning techniques with labelling noise for training content classification network on automatically labelled real-life data. Correspondingly, our classification network demonstrates well-performing results with moderate labelling noise.

Moreover, this work presents a novel use-case to state of the art. Our proposed obfuscation mask generation pipeline does not need separate training or annotations but instead exploits already learnt information from the classification network.

2 Background

2.1 Machine Learning

The field of machine learning studies systems by learning their data. This field focuses on the development of computer algorithms that can automatically extract data and use this information to build models to take actions without explicitly programmed by constraints [7]. This definition is usually expressed as [8]:

"A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P** if its performance at tasks in **T**, as measured by **P**, improves with experience **E**."

The task of the machine learning algorithm is dedicated to solve the problems. Over the scope of this thesis work, our task **T** will be mainly the classification of data points into classes. The experience **E** is represented by a dataset, where data points are collected. The performance measure is a quantitative measure of how well the machine learning model performs. Usually, the performance metric **P** in classification tasks is accuracy, which can be described as the ratio of correct classified data points with respect to all data points.

2.1.1 Learning Algorithms

Learning algorithms produce machine learning models using datasets, and they can be categorised as either supervised or unsupervised.

Supervised Learning

In supervised learning, each data sample has an associated target variable. In a classification task, the target variable is the class of the data sample. The model tries to predict the target value of an unseen input data sample using the provided information of training dataset. The goal is to achieve a good performance on unseen data samples.

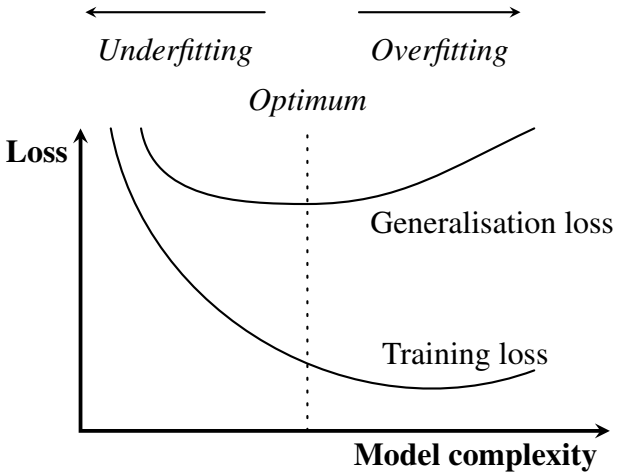


Figure 2.1: The graph depicts typical training and generalisation error curves with respect to the capacity of the model. Optimum model capacity is the point where the model reached the global minima of generalisation error. The left side of the optimum capacity is an underfitting model and on the right side of the optimum capacity is an overfitting model. The generalisation error typically has a U-shaped curve.

Unsupervised Learning

Opposed to supervised learning, target values are not provided for data points in unsupervised learning. The machine learning model can learn from the data without using data with labels. Hence it is more efficient in terms of resources, and this is a significant advantage over supervised learning.

2.1.2 Generalisation

Generalisation is the ability of the machine learning model to perform well on previously unseen input. To measure generalisation metrics, a dataset is divided into three subsets, which are called training set, validation set and test set.

The generalisation error is measured validation set. This subset is solely dedicated to this purpose and data points in validation set do not participate in learning. If a model does not perform well on unseen data, this might be caused by either underfitting or overfitting. The ultimate objective is to find the model with the lowest generalisation error. The generalisation error and training loss curve typically has a U-shaped curve as depicted in Figure 2.1.

Underfitting

A supervised learning model can be underfitting if its error on the training dataset is undesirably high. In this case, the model does not have enough capacity, and the number of trainable

parameters is insufficient.

Overfitting

If the model has too much capacity, then overfitting might be the underlying reason for high error on the test set. If the model is overfitting, then the training error is low, and the test error is undesirably high. Models with a high number of trainable parameters are more prone to overfit to the training set.

2.1.3 Model Selection

During training, several models are tested to choose the best model. If models are tested on a training set, the chosen model with the smallest error might be overfitted to a training set. It is an ideal practice to choose the model with the smallest generalisation error. For this purpose, a validation set is used to choose the best model. However, this practice causes a bias in predicted performance because the model has chosen in a way to minimise the error and maximise the performance in a validation set. Thus, in order to get a good approximation on how well the model performs on unseen data, the model should be evaluated on a test set.

2.2 Neural Networks

An artificial neural network (commonly referred to as a neural network) is a model of stacked computational layers that learns abstract features of training data needed to accomplish the task. Each layer consists of a different number of nodes (also referred to as perceptrons). Perceptrons are an abstract representation of biological neurons. Each perceptron is connected to each other with corresponding weights.

There are two main types of networks distinguishable by the type of interconnection: feedforward and recurrent. A network is called feedforward neural network if the connection of nodes goes only in the direction from the input layer to the output layer. If a layer has any connections to the same layer or previous layers, then this type of networks is called recurrent neural networks. For this thesis work, only feedforward neural networks are studied.

2.2.1 Model of a Perceptron

A perceptron consists of an activation function and a propagation rule. A propagation rule defines the single input of the perceptron by mapping its inputs into a single input value. An activation function is then applied on this single input value in order to obtain the output value of the perceptron. The output is also commonly referred to as activation. Please see Figure 2.2 depicting a single perceptron where y is the output of the unit, z is the actual single input value obtained by mapping $x_1, \ldots, x_N$ which denote inputs from other units within the network, and b that denoted the bias as an external input to the perceptron using the propagation rule, and f is the activation function.

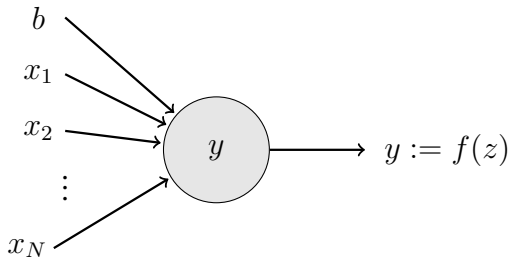


Figure 2.2: A single perceptron with N inputs and one output.

2.2.2 Feedforward Neural Networks

A feedforward neural network is a multilayered network of perceptrons where perceptrons are connected in series. Layers of a feedforward neural network are distinguished by specific names. The first layer that accepts the input of the whole network is called an input layer. Likewise, the last layer that forms the output of the whole network is called the output layer. All layers between the input layer and the output layer are called hidden layers.

The number of layers describes the depth of a feedforward neural network, and the number of units in a layer describes the width of that layer. In Figure 2.3, an example feedforward neural network of a depth D is depicted. The width of the i^{th} hidden layer is $m^{(i)}$, the width of the input layer is N, and the width of the output layers is M. The bias unit of the i^{th} hidden layer is $b^{(i)}$, and it is not counted to the width of that layer.

The units in every layer are fully connected (FC); thus, these layers are called "fully connected layer". In an FC layer, every unit of i^{th} hidden layer get the outputs of all units from the $i - 1^{th}$ hidden layer as input. If the width of the layers is increased, the number of links and weights are increased in high numbers. If i^{th} hidden layer has $(m^{(i)} + 1)$ units and if $i + 1^{th}$

hidden layer has $(m^{(i+1)}+1)$ units, then the number of weights between i^{th} and $i+1^{th}$ hidden layers is $|W_i| = (m^{(i)}+1)\cdot(m^{(i+1)}+1)$. Considering this, it is common for state-of-the-art neural networks have millions of parameters to be trained.

The example feedforward neural network in Figure 2.3 has $(D-1)$ FC layers. For d^{th} hidden layer, the output of unit y_i^d can be calculated as

$$y_i^d = f(z_i^d) \text{ where } z_i^d = \sum_{k=1}^{m^{(d)}} w_{ik}^{(d)} \cdot y_i^{(d-1)} + w_{i0}^{(d)} \cdot b^{(d)} \tag{2.1}$$

where $w_{ik}^{(d)}$ is the weight of the connection between y_i^d and $y_k^{(d+1)}$, $b^{(d)}$ is the bias unit in the d^{th} hidden layer.

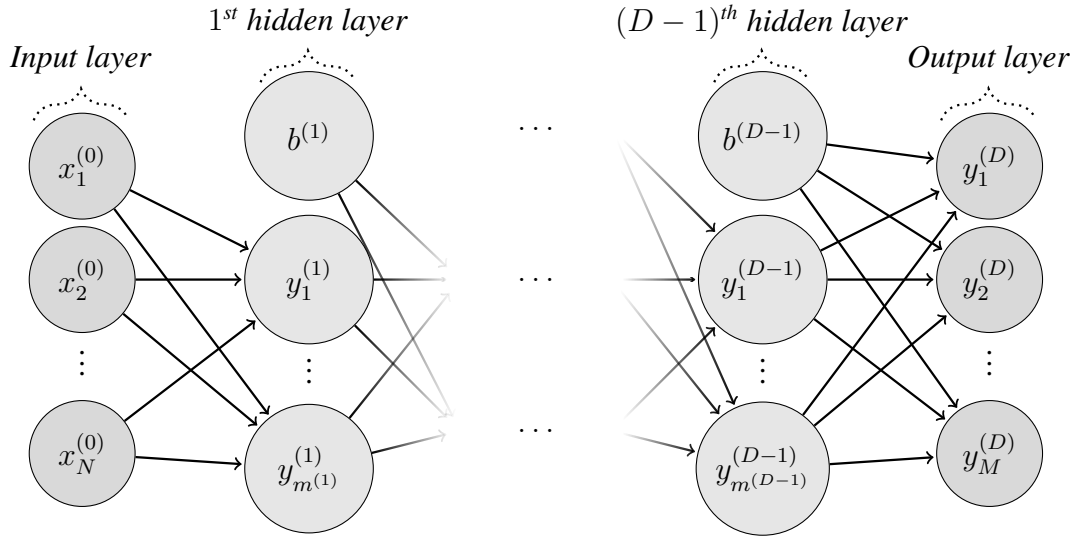


Figure 2.3: An example feedforward neural network [1]

2.2.3 Activation Functions

The activation function is the part that makes the network to solve nonlinear problems. Without an activation function, the neural network could only solve linear problems. Thus, the activation function should always be nonlinear. As explained in **2.2.5**, the gradient of the output should be computed for the backpropagation algorithm; therefore, the activation function must be continuously differentiable.

In early research, sigmoid and tanh functions were typically used as an activation function. Both of these functions have a slow convergence because, for extreme values of the input, the gradient approaches to zero. Moreover, if the network depth is high, vanishing or exploding gradients might occur. Exploding gradients can occur if the weights are large and the output

is really small, where the derivate of the output reaches the maximum value. Vanishing gradients can occur during backpropagation in early layers due to multiplication of small values. In today's research, new activation functions like ReLU are commonly used. In this section, activation functions that are used in this thesis work is presented.

ReLU - Rectifiere Linear Unit

A unit using the rectifier is called a rectified linear unit (so-called ReLU) [9]. This activation function is one of the most popular activation functions used for computer vision deep learning applications. The function is given in Equation 2.2 and it is illustrated in Figure 2.4.

$$\text{ReLU}(x) = (x)^+ = \max(0, x) \tag{2.2}$$

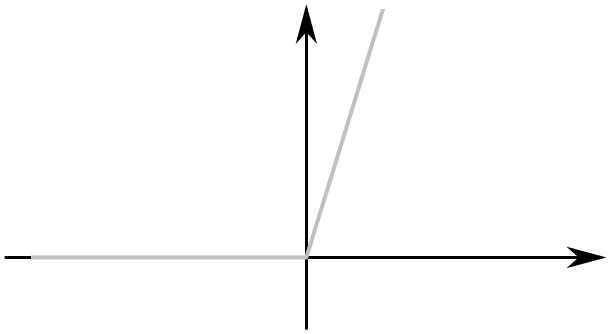


Figure 2.4: Diagram for ReLU activation function

Softmax and LogSoftmax

The Softmax function takes an input of a vector and normalises it into probabilities proportional to the exponentials of the inputs. It is commonly used to map non-normalized neural network output to a probability distribution over predicted outputs. Softmax and LogSoftmax can be expressed as in Equation 2.3 and Equation 2.4 and illustrated in Figure 2.5 and in Figure 2.6 respectively.

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{2.3}$$

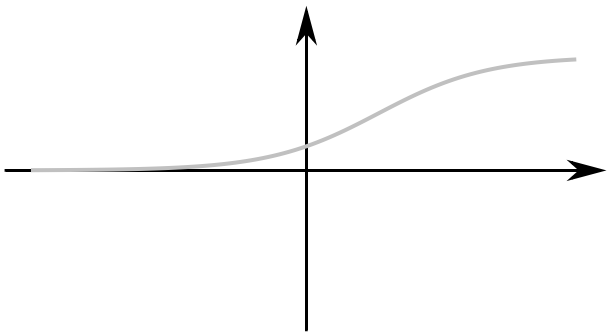


Figure 2.5: Diagram for Softmax activation function

$$\text{LogSoftmax}(x_i) = \log\left(\frac{\exp(x_i)}{\sum_j \exp(x_j)}\right) \tag{2.4}$$

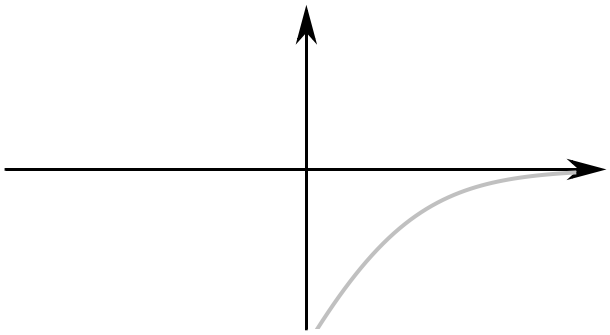


Figure 2.6: Diagram for LogSoftmax activation function

2.2.4 Loss Functions

A loss function is used to map the value of the model outputs to a single value that describes the discrepancy between the output and the target value. The goal of training of the neural network is to minimize the loss. The loss function choice depends on the learning problem.

Cross Entropy Loss

Cross Entropy loss is a common loss function for multi-class classification problems as in this thesis work. Cross-entropy calculates a loss between two distributions using the following

equation:

$$L_{CE} = H(\mathbf{q}, \mathbf{p}) = -\sum_{i=0}^{n} q_i \cdot \ln p_i \tag{2.5}$$

where **p** is the predicted output distribution and **q** is the target distribution.

Symmetric Cross Entropy Loss

Symmetric Cross Entropy loss was proposed in [10]. The paper [10] exhibited that neural networks train using Cross Entropy tend to overfit to noisy labels in some easy classes while significantly underfitting to hard classes. The proposed solution, Symmetric Cross Entropy, has an additional noise-robust counterpart term called Reverse Cross Entropy in order to address both overfitting and underfitting problems of neural networks on data with noisy labels [10]. In the scope of this thesis work, this loss is incorporated into the proposed solution as an experiment.

Symmetric Cross Entropy loss can be defined as:

$$L_{SCE} = \alpha \cdot L_{CE} + \beta \cdot L_{RCE} = \alpha \cdot H(\mathbf{q}, \mathbf{p}) + \beta \cdot H(\mathbf{p}, \mathbf{q}) \tag{2.6}$$

where L_{RCE} is Reverse Cross Entropy, α and β are decoupled hyperparameters.

Cross Entropy Loss with Label Smoothing

Label smoothing has been commonly used in order to improve the accuracy, generalisation, and learning speed of neural networks focusing on various tasks, including image classification, language translation and speech recognition. This technique was initially proposed as a strategy to significantly improve the performance of Inception-V2 architecture on the ImageNet dataset [11]. This technique uses soft targets that are a weighted average of the hard targets in order to prevent the network from becoming over-confident. The interpretation of label smoothing regularisation considering cross entropy can be expressed as:

$$L_{CELS} = H(\mathbf{q'}, \mathbf{p}) = (1 - \epsilon) \cdot H(\mathbf{q}, \mathbf{p}) + \epsilon \cdot H(\mathbf{u}, \mathbf{p}) \tag{2.7}$$

where the second loss is used as a penalty for the deviation of predicted label distribution **p** from the prior **u**, with the relative weight $\frac{(1-\epsilon)}{\epsilon}$ and ϵ is smoothing parameter.

2.2.5 Optimisation Procedure

Backpropagation Algorithm

Algorithm 1 Backpropagation learning algorithm

for image in dataset **do**
  FORWARD PROPAGATION
    Starting from the input layer, perform a forward pass through the network using Equation 2.1 to calculate the outputs of neurons at each layer
  BACKWARD PROPAGATION
    Calculate the derivatives of the error function with respect to the network outputs
    for layer in layers **do**
      Calculate the derivatives of the error function with respect to the inputs of the neurons in the upper layer
      Calculate the derivatives of the error function with respect to the weights between the layer ahead and the layer before
      Compute the derivatives of the error function with respect to the outputs of the layer before
    end for
    Updates the weights
end for

Optimisers

Adam: Adam [12] is an optimisation algorithm that can be used to update network weights iteratively based on training data. It has been designed specifically for training deep neural networks. Adam is an adaptive learning rate method, which means it computes the learning rates of different parameters individually.

2.2.6 Regularisation

Regularisation methods control the overfitting behaviour of the network and reduce the generalisation error. While some techniques are commonly utilised in machine learning, others are used only for neural network approaches. Large models, if adequately regularised, tend to be the best fitting models [13]. In this section, the regularisation methods used in this thesis are presented.

Early Stopping

While training large models, both the training and validation error decreases over time. However, at some point, the validation error starts increasing. This process is depicted in Figure 2.1.

This is an indication that the model is beginning to overfit to the training set, and the generalisation capacity is reducing. Early Stopping is a method to stop the training at this point for the sake of generalisation. The training is evaluated on the validation set, and if the validation metric starts improving for a predetermined number of epochs, then the practice gets stopped. This approach ensures returning the best validation model.

Dropout

Dropout [14] is a regularisation method used in neural networks. The key idea is that the connections between neurons are randomly dropped with a certain probability in every iteration of the training. For testing, the dropout is not available, and every connection is preserved. The main drawback of dropout is slowing the convergence and increasing the training time because not all weights are trained in every iteration.

Batch Normalisation

As the weights on every layer changes during the training, changes in previous layers cause a shift in the input distribution for the consecutive layers. This phenomenon is called internal covariate shift [15]. In order to counteract this change, a technique called batch normalisation is introduced in [15]. In this technique, every input mini-batch is normalised on the input of the previous layer.

2.3 Convolutional Neural Networks

A CNN consists of multiple convolutional layers and pooling layers typically connected to a fully connected layer block at the end. Convolution layers are used to extract useful features from the input image, and they output multiple feature maps. A pooling layer is then usually applied to reduce the spatial size of these feature maps. An example CNN is shown in Figure 2.7 where convolutional and pooling layers are followed by two fully connected layers.

It is quite common to apply convolution operation on images in image processing in order to manipulate an image or extract features from an image. The difference between the convolution operation in a CNN and in traditional computer vision is that instead of using hard-coded filters, CNN learns these filters that optimise it for the ultimate goal of the network. In the following of this section, CNN components used in this thesis work are presented.

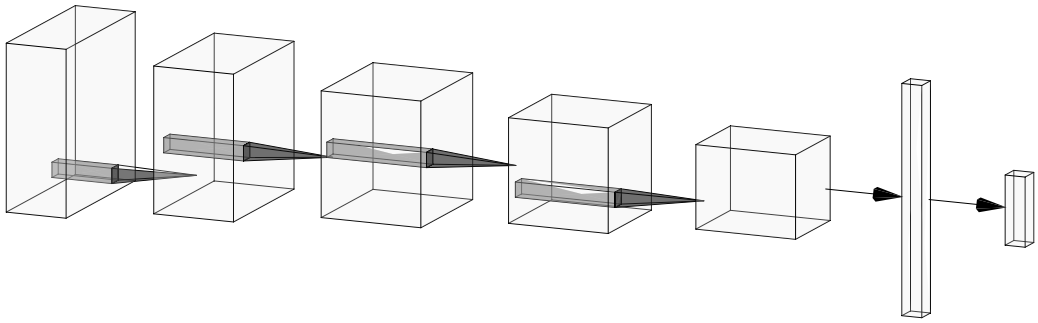


Figure 2.7: A diagram depicting an example convolutional neural network [2]

2.3.1 Convolutional Layer

The main reason for using convolutional layers in neural networks for image processing is that they can exploit the spatial information of the input and learn how to transform it to most informative form possible. The behaviour of a convolutional layer can practically be controlled using some set of parameters in order to adapt their usage into different problems. These parameters are listed in this section.

Number of filters

The number of filters is the number of dimensions in the output of the later. Each filter produces a distinct feature map; therefore, the number of dimensions is the same as the number of filters. Considering the size of feature maps decrease along the network while moving toward the output; increasing the number of filters might help to decrease the amount of information loss in the network. Increasing the number of filters increases the number of total trainable parameters and network capacity.

Filter size

In a convolutional layer, not all units of the input are connected with the output unit. The number of units in a layer connected to the output unit is controlled by filter size. Filter size is the area of the filter sliding along a layer. For deep neural networks, the optimal filter size is usually determined experimentally.

Padding

The convolution operation is not defined along the borders of the input as the filter cannot overlap with any input value along the edges. As a solution to this problem, the input can be

padded with zeros along the borders. Padding can be used to control the output size as the number of input units that convolution can be performed directly affects the size of the output.

Stride

Stride is the number of units the filter will skip while moving along the image. It can be considered as a step size. Stride can be used to control the output size of feature maps.

2.3.2 Pooling Layer

Typically a pooling layer is applied consecutively after some convolutional layers. It is used to reduce the spatial size of input feature maps by downsampling. This approach helps to reduce the total number of trainable parameters and computation time. Pooling layers do not have any activation function or trainable weights. There are mainly two common kinds of pooling layer types: Average Pooling, Max Pooling.

The average pooling computes the average of the pixel values in the receptive field of the filter while the max pooling computes the maximum value.

2.3.3 Popular CNN Architectures

LeNet

The LeNet [16] architecture was first introduced in 1998. It is a convolutional network that consists of 7 layers and is used to classify hand-written digits by several banks. It is the first successful CNN application.

AlexNet

AlexNet [17] was the first work that popularized CNNs for computer vision after taking the first place and outperforming all other approaches in the ImageNet challenge of 2012. AlexNet had a substantially similar architecture as LeNet, yet it was deeper with stacked convolutional layers, each of which had more filters.

GoogLeNet

GoogLeNet [18] (also known as Inception v1) was the winner of ILSVRC 2014 competition. This architecture included an Inception Module that uses convolutional layers of different sizes

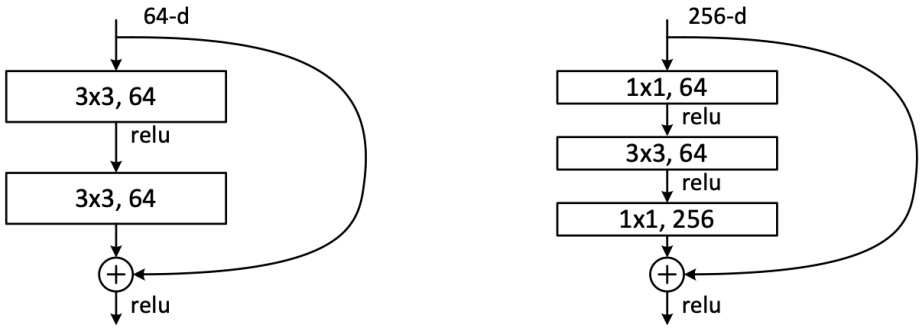


Figure 2.8: The building blocks of ResNet. *Left:* The blocks used in 56x56 feature maps *Right:* The blocks used in ResNet50/101/152 [3]

and concatenates their output feature maps allowing the model to decide which filter size is optimal for capturing the features. Moreover, the network used batch normalisation, image distortions and RMSprop as an optimiser.

VGGNet

VGGNet [19] took second place in ILSVRC 2014. The network showed how vital the network depth is. It had a uniform architecture consisting of 16 trainable layers and employed only 3x3 convolutions and 2x2 pooling. However, the network was costly, considering its 140 millions of trainable parameters.

ResNet

ResNet [3] was the winner of the ILSVRC 2015 competition. The network utilised special residual blocks that present significant improvement for optimisation of the cost function and featured heavy use of batch normalisation.

ResNet was initially proposed with five versions which respectively containing 18, 34, 50, 101, and 152 layers. The network architecture is presented in Table 2.1. The brackets represent the building blocks depicted in Figure 2.8.

SENet

SENet [20] won the ILSVRC 2017 competition. It exploited a mechanism of an adaptive weighting of intermediate feature maps in the hidden layers of network in order to allow the network to learn the best features available from the input.

Table 2.1: Detailed architectures of ResNet variations. The building blocks in brackets are presented in Figure 2.8. [3]

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

2.3.4 Transfer Learning

Transfer learning is the practice of using a pre-trained model on a new task and in a new domain [21]. The training domain is called the source domain; the new domain is called the target domain. Likewise, the new task is called the target task, and the task in the source domain is called the source task. For instance, a handwritten letter classifier can be used to classify digits. For this thesis work, we utilised convolutional neural networks trained on ImageNet to classify images shared on social media for content moderation.

There are two typical transfer learning approaches: feature extraction approach and weight initialisation approach. Feature extraction is when the output of some layer in the pre-trained network is used as features for a classifier. Most of the time, the last convolutional layer is used for this purpose. The weight initialisation approach uses weights of some early layers and then trains the whole network fro the new task. This approach is also called fine-tuning as it tunes the network for the new task and domain.

2.3.5 Gradient-weighted Class Activation Mapping

Even though spatial information is usually lost in fully-connected layers, convolutional layers typically preserve this information. The neurons in these layers highlight class-specific features in the image. Gradient-weighted Class Activation Mapping is a technique that exploits the gradients information in these layers to determine the importance of each neuron for the output layer decision [4]. The class-discriminative localisation map L_c of an image of width u, and

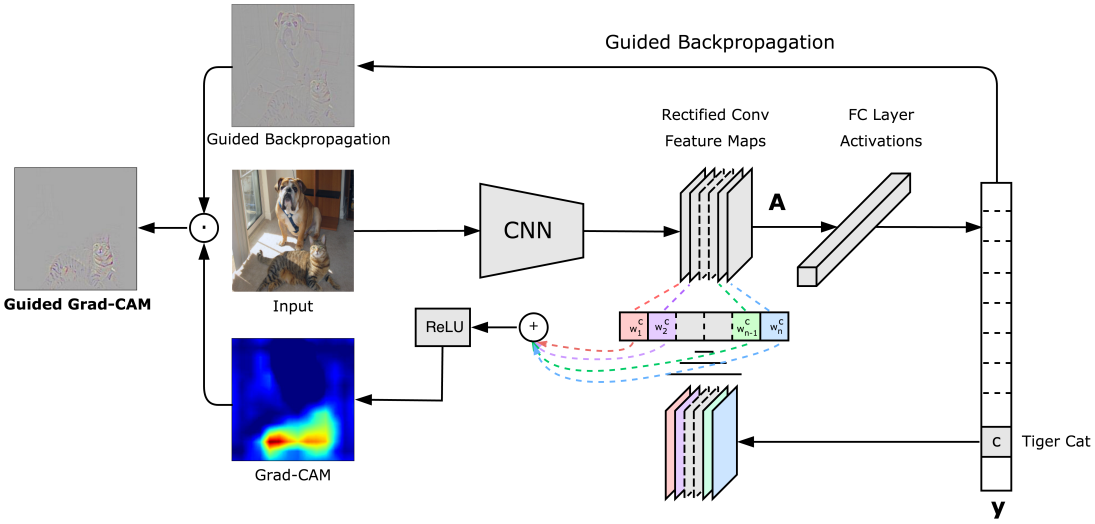


Figure 2.9: Overview of gradient-weighted class activation mapping [4]

height v that belongs to class c can be calculated as follows:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\delta y^c}{\delta A_{ij}^k} \tag{2.8}$$

$$L_c = ReLU(\sum_k \alpha_k^c A^k) \tag{2.9}$$

where A^k is the activation of the feature map of the convolutional layer, y^c is the class score logit, and α_k^c is the neuron importance weights.

2.4 Related Work

The recognition of sexually explicit images can be categorised into four main approaches: methods based on colour information, methods based on shape information, methods based on local features, methods based on deep learning.

Colour-based methods typically analyse the images for low-level image attributes like skin colour and classify pixel regions as skin or non-skin. The detection approach can be as simple as manually checking whether the pixel colour is within a pre-defined skin colour range [22]. Furthermore, it can also exploit colour distribution information using simple histogram learning techniques [23] or parametric colour distribution functions [24]. After extracting skin colour features of the image, a sexually explicit content can be detected by either thresholding the computed histogram or training a classifier on the extracted features [25]. Another approach

[26] exploiting skin region identification adopts a combined method of HSV colour modelling together with YIQ and YUV models. The proposed work extracts colour information using a white balance algorithm and grey level comatrix. This work also utilises structural data to remove noise coming from the background. Further discussion on various skin modelling and classification approaches using colour information is carried in [27]. This survey focuses on colour spaces used for skin recognition.

Secondly, shape-based methods usually use the shape information or compactness of colour in the skin areas. These methods analyse the structural information of skin colour regions. For instance, some utilise contours of skin colour regions as a feature for the classification [22]. Others use shape features of skin regions obtained using shape descriptors and feed them into a classifier to detect whether the detected skin regions represent nudity or not [28] or propose using geometric constraints to model the geometry of the human body [29]. Moreover, most of the works utilise colour and shape information together to boost their performances.

Moreover, the success of using local features for classification in other image recognition problems led researchers into the third approach. For instance, sexually explicit images can be recognised using Support Vector Machine (SVM) on the features of Scale Invariant Feature Transform together with the bag of words [30].

Last but not least, deep learning based approaches are the most recent and most advanced image content recognition method. An example work is presented in [31], which makes use of residual learning by increasing the depth and consequently, detection accuracy. Another popular work was developed by Yahoo! [32] to detect Not-Safe-for-Work content, which outputs a probability score for explicit images. A similar approach to ours is used in [33] to classify images as explicit/non-explicit using transfer learning. Some more recent works also focus on explicit content recognition other media such as videos either sequentially feeding several frames into a classification network based on LTSMs or concatenating several frames together beforehand for 3DCNNs [34, 35].

3 Dataset

3.1 Data Source

At the time of writing, there are two major broadly-used social media platforms that allow sharing sexually explicit content to some extent: Twitter and Reddit. Other major social media platforms such as Facebook, Instagram, Pinterest, and Tumblr have terms of services that are excessively restrictive against explicit sexual content, and therefore not a suitable option for data mining in the scope of this thesis work. Due to a more structured way of sharing content compared to Twitter, Reddit was used to mine data for the dataset.

Content is shared on Reddit into subcommunities, so-called subreddits, created by users. A subreddit consists of posts shared by users that may contain texts, images, videos, or links. Users can upvote to reward interesting posts other users' shared. Each subreddit focus on one topic or theme and has a content-sharing policy and moderation in order to improve content quality. Therefore, images posted in a subreddit principally can be identified with the subreddit topic. On the other hand, they might encompass a wide variety in terms of image features since they are posted and created by different people. Thus, Reddit constitutes a good source of images labelled with a topic or theme. For instance, the subreddit `r/dogpictures` is dedicated to images of dogs, as the name suggests, and it does not allow sharing images of certain file formats and any images that are not related to such. Moreover, Reddit also allows posting sexually explicit content. Posts that include such content are required to have an NSFW (not safe for work) tag so that these post can be hidden from users who have not explicitly given consent to see them.

Notwithstanding enforced moderation and content policies, there can still be posts that are not the best fit for their subreddits in terms of image content. Some images can be related in a roundabout way to the theme of the subreddit and can be tolerated. Another reason for such inconsistencies can be that moderators who enforce the content policies can often be over-

whelmed by the number of posts and might miss some outlier posts. Due to these reasons, our collected dataset might have label noise to some extent, depending on the subreddit the image is sourced.

3.2 Dataset Description

The dataset consists of 5 classes: *Explicit Nudity, Suggestive Nudity, Neutral, Sexually Explicit Artwork, Neutral Artwork*. Each single one of them is curated among images scraped from hand-selected subreddits with exclusively photo submissions. Subreddits are chosen in a way that its theme fits into the class description. Since the data mining process is automised, the quality of this dataset is limited by the quality and homogeneity of the subreddit itself. In order to ensure quality, subreddits that are not effectively moderated or have too granular content range are not taken into consideration. Furthermore, posts are filtered out by popularity. The number of upvotes given to the post by other users is used as a metric for the popularity of the post. Due to the nature of the data, the names of the subreddits used in this dataset includes explicit words that might not be suitable for a general audience, and due to this reason, they were not revealed in this section.

The images in the dataset were downloaded for a duration of several months among daily posts that received at least a particular amount of popularity. Each subreddit have a preset threshold for this metric considering active user count and the number of daily posts. In order to minimise potential duplicates in the dataset, subreddits that people usually cross-post the same images are not taken into consideration. As an extra step to have a fair evaluation, testing images are downloaded in a distinct time period and manually labelled.

Classes

- *Explicit Nudity:* This class consists of content primarily includes photos depicting real-life human genitals or buttocks, female-presenting nipples, sexual act, sexual bodily fluids, complete nudity. Selected examples from this class is presented in Figure 3.1.a.
- *Suggestive Nudity:* Some possible contents that this class encompass are photos of male-presenting nipples, partial coverage of buttocks, humans in swimwear, underwear, clothes with cleavage or revealing clothes. Examples of images from this class is depicted in Figure 3.1.b.
- *Sexually Explicit Artwork:* This class consists of artwork or photos of an artwork that

illustrates full or partial nudity, sexual intercourse or other sexual acts of humans or depictions of animals with human-like features. Several examples of images from this class is presented in Figure 3.1.c.

- *Neutral Artwork:* This class consists of artworks or photos of artworks that do not suggest or present any sexuality and/or nudity and those that should not be in Sexually Explicit Artwork class. Some examples from this class can be found in Figure 3.1.d.

- *Neutral:* Any photos that do not suggest or present any sexuality and/or nudity is included in this class. Some examples from this class is presented in Figure 3.1.e.

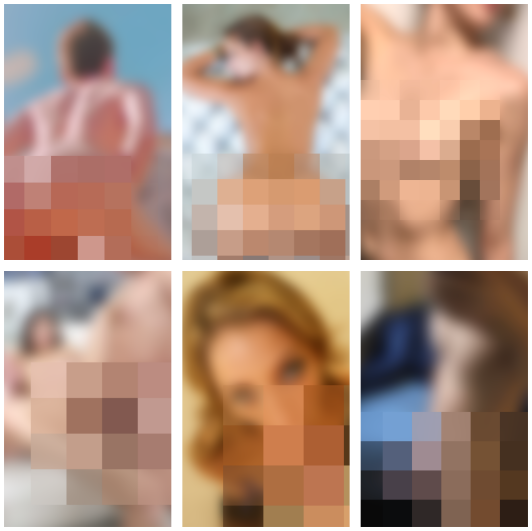
Statistics

Number of samples in the dataset is presented in Table 3.1. The dataset consists of 106479 images in total, 8000 of which is dedicated for testing. Testing set is manually labelled and downloaded in a distinct time period to prevent duplicate images appearing during both training and testing the model.

Table 3.1: Number of samples in the dataset

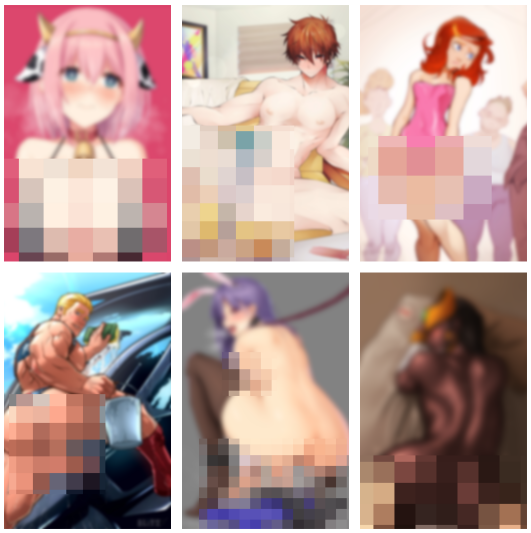
Class	**Number of samples (Training, Validation)**	**Number of samples (Testing)**
Explicit Nudity	20221	2000
Suggestive Nudity	17262	2000
Neutral	26245	2000
Sexually Explicit Artwork	17074	2000
Neutral Artwork	17677	2000

In order to obtain a generalisable model, subreddits that are the source of images are chosen with a considerable variety in terms of some factors including (but not limited to) gender, skin colour, sexual orientation and/or identity, performed sexual acts, and image quality. The number of samples in the training set are composed considering the required variety to enable this generalisation. For instance, Neutral class has a higher number of samples because this category can include any non-sexual photos such as photos of babies, nature, pets, or any random objects.



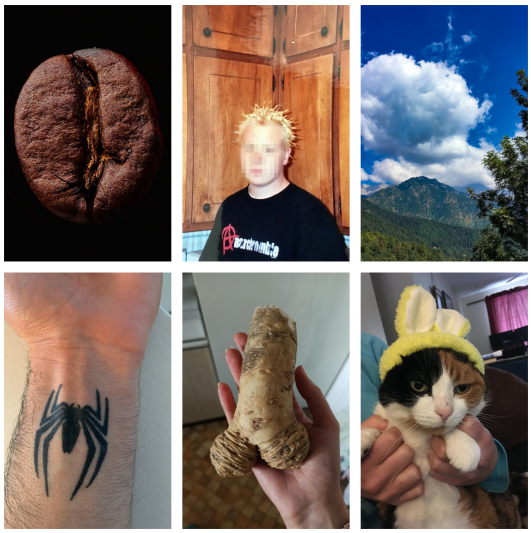
(a) *Explicit Nudity*

(b) *Suggestive Nudity*



(c) *Sexually Explicit Artwork*

(d) *Neutral Artwork*



(e) *Neutral*

Figure 3.1: Selected examples from the dataset

4 Proposed Solution

Our proposed solution consists of two parts:

1. A convolutional neural network to analyse the content
2. An auto obfuscation algorithm for censoring restricted content.

The first part is used to classify uploaded images to a social media platform, and analyse whether the image complies with content policies. We classify an image into five categories: Explicit Nudity, Suggestive Nudity, Neutral, Sexually Explicit Artwork, Neutral Artwork. These classes were expected to be granular in order to tailor this solution to different platforms depending on their content appetite. This approach is a necessity considering the definition of appropriate content is not binary.

If the content contains explicit nudity, we propose an algorithm to cover regions containing nudity automatically. This algorithm is a novel use case to the state of the art which eliminates the necessity of additional training and annotations to perform this task. Obfuscating images automatically is an essential addition to content moderation that presents several additional options whenever an upload is predicted to be not complied with content policies. If it is tolerated by the community, an obfuscated image can be published in a post. Otherwise, if this flag requires a manual moderation, an image might be edited first before presenting it to human moderators, as long term exposure to such content might cause mental health issues in [6]. If an image is not suitable for such modification, the content can be removed automatically.

4.1 Proposed Solution for Possible Scenarios

Depending on the input image content, there are two main scenarios. Scenario 1 is the case where the input image is acceptable by the social media platform, and consequently, no further processing is needed. In this case, our pipeline only consists of content detection network, and no obfuscation is necessary. Scenario 2 corresponds to the case where the input image is

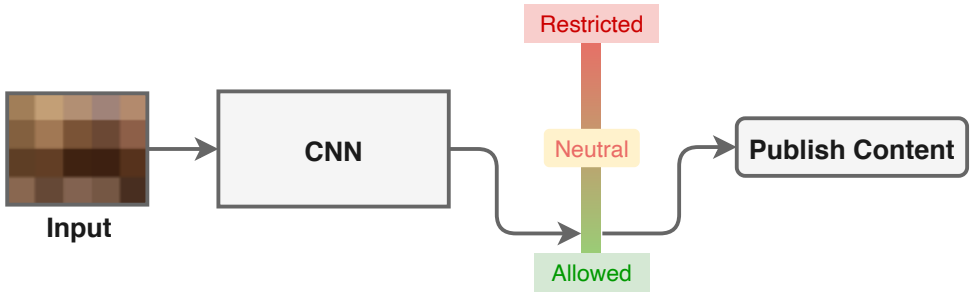
improper for the social media audience, and therefore the image needs to be either obfuscated or removed. In such a situation, the processing continues with automatic obfuscation. Two main aforementioned scenarios are depicted in Figure 4.1.

Scenario 1:

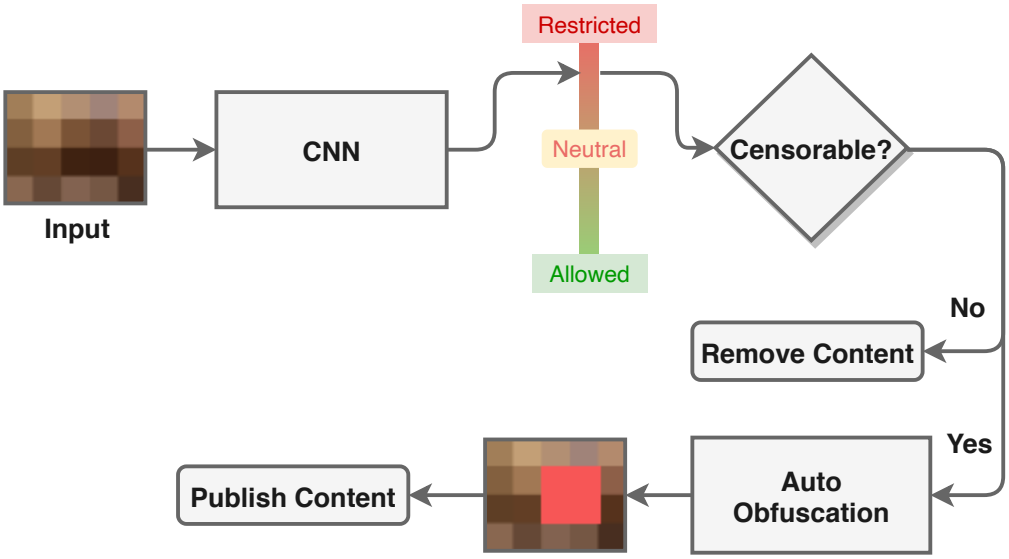
1. The user uploads an image.
2. The input is classified as an allowed content class by the CNN.
3. The content is published.

Scenario 2:

1. The user uploads an image.
2. The input is classified as a restricted content class by the CNN.
3. If the content is censorable, an algorithm automatically obfuscated the image, and the modified content is published. If the content is not censorable, the upload is removed.



(a) The proposed solution scenario for an allowed content



(b) The proposed solution scenario for a restricted content

Figure 4.1: The proposed solution scenarios

4.2 Proposed Approaches for Content Classification

In this section, we will present the proposed model architectures and the experiment configurations for solving the problem. These experiments are then evaluated in the next section. Over the course of this thesis work, we searched for a good CNN for content moderation.

The primary concept was exploiting the information from existing models trained for huge classification problems such as ImageNet [36] challenge. This is a popular strategy that is called transfer learning. Details about Transfer Learning is discussed in Section 2.3.4. For the transfer learning, we used three models pre-trained on ImageNet [36], which are ResNet-18, ResNet-50 and ResNet-152 [3]. ImageNet dataset includes around 1.2 million images in 1000 classes. These classes do not directly overlap with our classes. Though, the scope of content moderation quite broad and *Neutral* class might include (but not limited to) anything from these 1000 classes.

For the rest of this section, we will present numerous experiments that investigate various techniques or configurations to find an optimal augmentation policy, loss function, network architecture, and a number of layers. Even though some modifications and configurations might seem arbitrary for the early experiments, the reasoning of these choices will be validated in the next experiments.

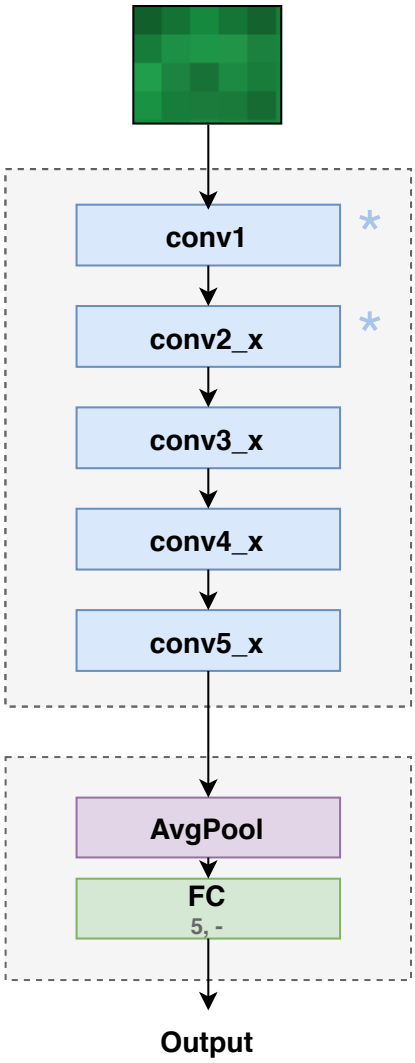


Figure 4.2: The proposed architecture for **Experiment 3**.

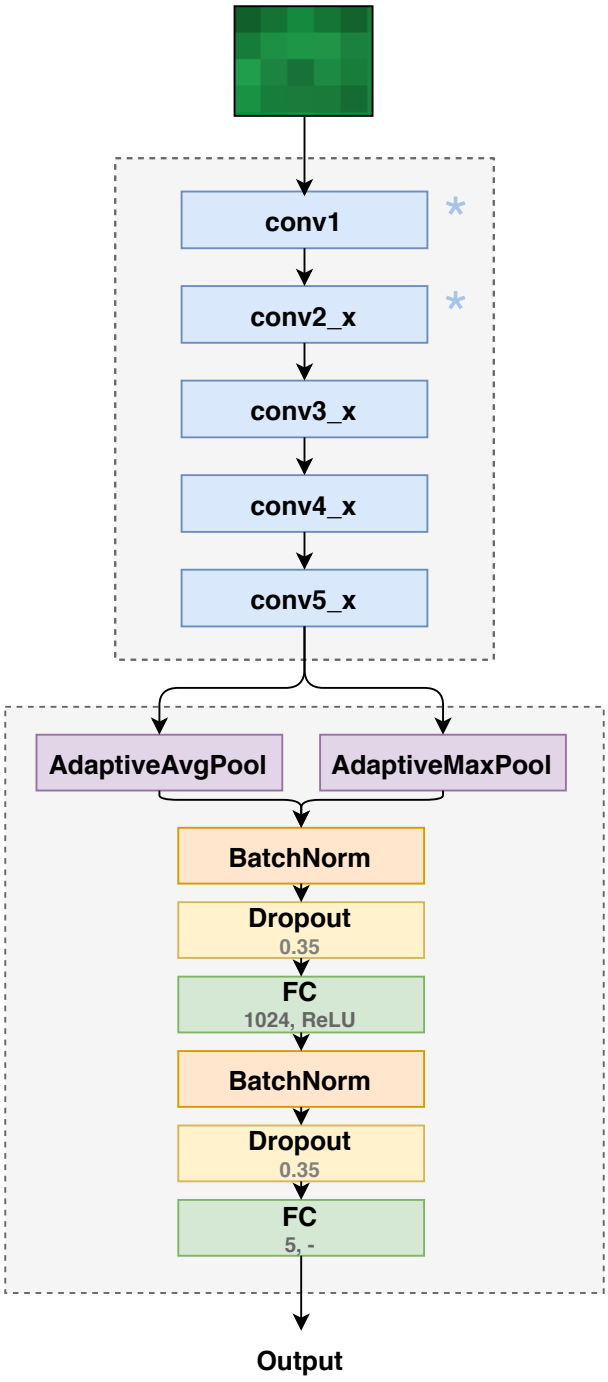
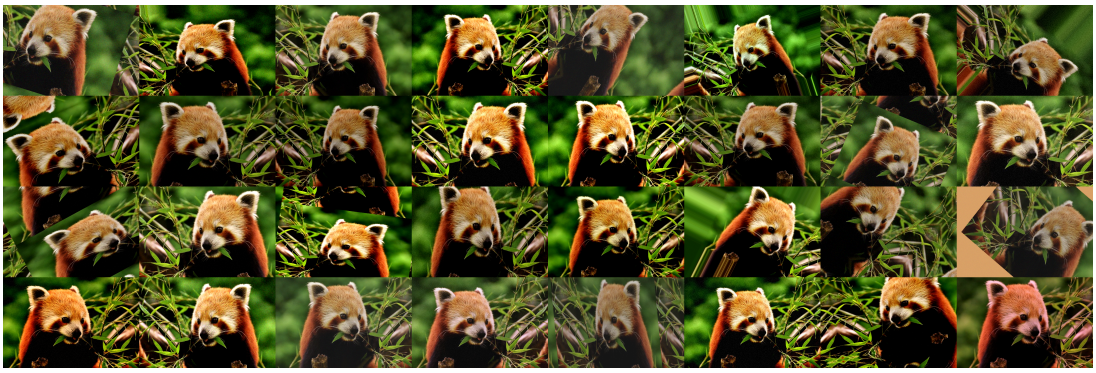


Figure 4.3: The proposed architecture for **Experiment 1, 2, 3, 4**.

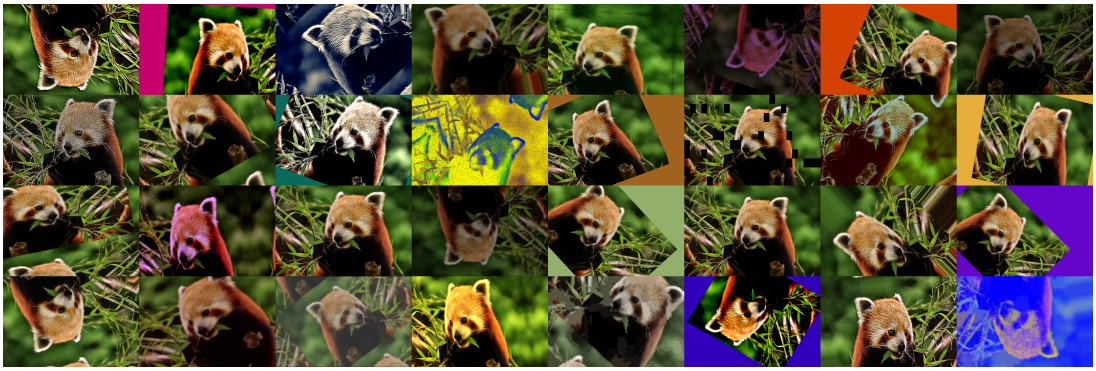
4.2.1 Experiment 1: Augmentation strategies

We experimented with different augmentation strategies for the classification network. The architecture used for this experiment is presented in Figure 4.3. Detailed information regarding the building block in this architecture is provided in Section 2.3.3. The classification head of the model is replaced by a concatenation of max and average pooling followed by two fully connected layers regularised by batch normalisation and dropout. First three convolutional blocks are frozen (i.e. their weights are not updated during the training), and the rest of the blocks are trained with varying learning rates. The training configuration is the same as Model 2.1 listed in Table 4.2. We use the cross-entropy loss function as described in Section 2.2.4. As the optimisation method, Adam [12] is used with a batch size of 64. If the validation accuracy is not improved, the learning rates are scaled by 0.1. The training job is stopped by the early stopper if the validation accuracy stops improving after 10 epochs.

The first model *Model 1.1* is trained with no augmentation. *Model 1.2* and *Model 1.3* are trained with standard and aggressive augmentation strategies with the configurations presented in Table 4.1. The augmentation strategies are illustrated on an example input image in Figure 4.4.



(a) Visualisation of applied Standard Augmentation



(b) Visualisation of applied Aggressive Augmentation

Figure 4.4: Visualisation of augmentation strategies used in *Experiment 1*

Table 4.1: Augmentation configurations used for training of **Experiment 1**.

Model 1.1: *No Augmentation*
Model 1.2: *Standard Augmentation*
Model 1.3: *Aggressive Augmentation*

	Standard Augmentation		*Aggressive Augmentation*		
Parameter	**Value**	**Probability**	**Value**	**Probability**	
Horizontal Flip	Yes	0.5	Yes	0.5	
Vertical Flip	No	-	Yes	0.2	
Cropping and Padding of the Height/Width	[0,0.1]	1.0	[0-0.1]	1.0	
Gaussian Blur Sigma	[0,0.5]	0.5	[0,0.5]	One at a time	
Average Blur Kernel Size	-	-	[2,7]		
Median Blur Kernel Size	-	-	[3,11]		
Linear Contrast	[0.75,1.25]	1.0	[0,5,2.0]	1.0	
Dropout Pixels	-	-	[0.01,0.1]	One at a time	
Coarse Dropout Pixels	-	-	[0.03,0.15]		
Additive Gaussian Noise	[0,12.75]	1.0	[0,12.75]	1.0	Up to
Sharpened Image Overlay	-	-	[0,1.0]	1.0	
Embossing Effect Image Overlay	-	-	[0,1.0]	1.0	
Grayscale image Overlay	-	-	[0,1.0]	1.0	
Edge Detection Image Overlay	-	-	[0,0.7]	One at a time	
Directed Edge Detection Image Overlay	-	-	[0,0.7]		
Multiply Pixels	[0.8,1.2]	1.0	[0.5,1.5]	1.0	
Add to Pixels	-	-	[-10,10]	1.0	
Scale	[0.8,1.2]	0.5	[0.8,1.2]	0.7	
Translate	[-0.2,0.2]	0.5	[-0.2,0.2]	0.7	
Rotate (degrees)	[-45,45]	0.5	[-45,45]	0.7	
Shear	[-16,16]	0.5	[-16,16]	0.7	

4.2.2 Experiment 2: Noise-robust loss functions

Our dataset is mined from a social media platform based on posted communities and not manually annotated. This approach comes with an unpreventable drawback: labelling noise. Training well-performing neural networks in the presence of noisy labels might be challenging. We experimented with some strategies that might present better performance with labelling noise. We took *Model 2.1* as the baseline and observed potential improvements. Training configurations for this experiment is presented in Table 4.2. For *Model 2.2*, we used a configuration with Symmetrical Cross Entropy loss function whose details are explained in Section 2.2.4. The coefficients for Cross Entropy and the noise robust counterpart Reverse Cross Entropy parts were taken as 0.1 and 1.0 respectively. This approach was presented by Wang et al. in 2019 [10]. Whereas *Model 2.3* has Cross Entropy loss function with label smoothing. This approach might help the model training with mislabeled data and improve the robustness. We used a label smoothing factor of 0.1 for training. This technique was initially proposed as a strategy to improve the performance of Inception-V2 [11].

Table 4.2: Training configuration used for training of **Experiment 2**.

	Parameter		**Value**		
			Model 2.1	**Model 2.2**	**Model 2.3**
Model	Model Architecture		See Figure 4.3	See Figure 4.3	See Figure 4.3
	Trainable Convolutional Blocks		3,4,5	3,4,5	3,4,5
	Input Size		224	224	224
	Batch Size		64	64	64
	Epochs		200	200	200
Loss	Criterion		CE w/o LS	SCE $(\alpha = 0.1, \beta = 1.0)$	CE w/ LS $(\epsilon = 0.1)$
Optimisation	Learning Rate	*cov3_x*	0.00001	0.00001	0.00001
		cov4_x	0.00001	0.00001	0.00001
		cov5_x	0.0001	0.0001	0.0001
		FC	0.001	0.001	0.001
	Optimiser		Adam	Adam	Adam
	β_1		0.9	0.9	0.9
	β_2		0.999	0.999	0.999
	Learning Rate Reduction Factor		0.1	0.1	0.1
	Learning Rate Reduction Patience		5	5	5
	Early Stopping Patience		10	10	10
Preprocessing	Feature Scaling		Yes	Yes	Yes
	Colorspace		BGR	BGR	BGR
Augmentation	-		Aggressive	Aggressive	Aggressive

4.2.3 Experiment 3: Classification-head architectures

If one compares the architectures shown in Figure 4.2 and Figure 4.3, the only difference in principle is the classification head of the network. The network with only a single fully-connected layer is the same as the original ResNet architecture [3] except for the width of the output layer. As the original ResNet model is trained on ImageNet dataset, it had a different number of classes to classify. After a series of experiments on various modifications for the classification head, we came up with the architecture in Figure 4.3.

This approach was based on assumptions that applying both average pooling and max pooling to the feature map of the last layer might help to preserve more useful information than solely using average pooling. This study cannot be considered large enough for ruling out a generalisation, but rather it can be evaluated in our unique use case. To access how useful the modification we made on the classification head, we performed this experiment. The architecture used for *Model 3.1* is shown in Figure 4.2, and the architecture for *Model 3.2* is shown in Figure 4.3. Detailed information regarding the building blocks in this architecture is provided in Section 2.3.3. The layers marked with an asterisk are frozen (i.e. their weights are not updated during the training). For training, the same configurations for *Model 2.3* are used.

In order to access how useful the modification we made on the classification head, we performed this experiment. The architecture used for *Model 3.1* is presented in Figure 4.2, and for *Model 3.2* the architecture in Figure 4.3 is used. Detailed information regarding the building blocks in this architecture is provided in Section 2.3.3. The layers marked with an asterisk are frozen (i.e. their weights are not updated during the training). For training, the same configurations for *Model 2.3* are used.

4.2.4 Experiment 4: Number of ResNet layers

ResNet architectures are named according to the number of layers each building block involves. As it is covered in detail during Section 2.1.2, the capacity of a network has a direct influence on the generalisation performance of the network. The capacity of the network increases as the number of layers (hence the number of trainable parameters) increases. This experiment can validate our choice of network architecture in previous experiments. We use respectively 128-, 50-, and 18-layered ResNet architectures for *Model 4.1*, *Model 4.2* and *Model 4.3*. The training configurations are presented in Table 4.3. Please note that *Model 4.1* corresponds the the same model as of *Model 3.1* in Experiment 3.

Table 4.3: Training configuration used for training of **Experiment 4**.

	Parameter		**Value**		
			Model 4.1	**Model 4.2**	**Model 4.3**
Model	Model Architecture		ResNet-128	ResNet-50	ResNet-18
	Trainable Convolutional Blocks		3,4,5	3,4,5	3,4,5
	Input Size		224	224	224
	Batch Size		64	64	64
	Epochs		200	200	200
Loss	Criterion		CE w/ LS ($\epsilon = 0.1$)	CE w/ LS ($\epsilon = 0.1$)	CE w/ LS ($\epsilon = 0.1$)
Optimisation	Learning Rate	*cov3_x*	0.00001	0.00001	0.00001
		cov4_x	0.00001	0.00001	0.00001
		cov5_x	0.0001	0.0001	0.0001
		FC	0.001	0.001	0.001
	Optimiser		Adam	Adam	Adam
	β_1		0.9	0.9	0.9
	β_2		0.999	0.999	0.999
	Learning Rate Reduction Factor		0.1	0.1	0.1
	Learning Rate Reduction Patience		5	5	5
	Early Stopping Patience		10	10	10
Preprocessing	Feature Scaling		Yes	Yes	Yes
	Colorspace		BGR	BGR	BGR
Augmentation	-		Aggressive	Aggressive	Aggressive

4.3 Obfuscation of Inappropriate Images Automatically

The second part of our proposed solution is an algorithm automatically obfuscates parts of an image containing explicit nudity. We propose a novel use-case that does not need separate training or annotations but instead utilises information obtained from the classification network. We process gradient-weighted class activation maps to produce obfuscation masks. These maps are usually used to diagnose neural networks by visualising the regions to which the network paid attention while making the decision. These maps are obtained by exploiting the gradients in the convolutional layers and highlighting neurons important for the network output. Details regarding this visualisation technique are presented in Section 2.3.5.

The process of producing obfuscation masks is depicted in Figure 4.5. Each step is marked in the figure with the corresponding numbers.

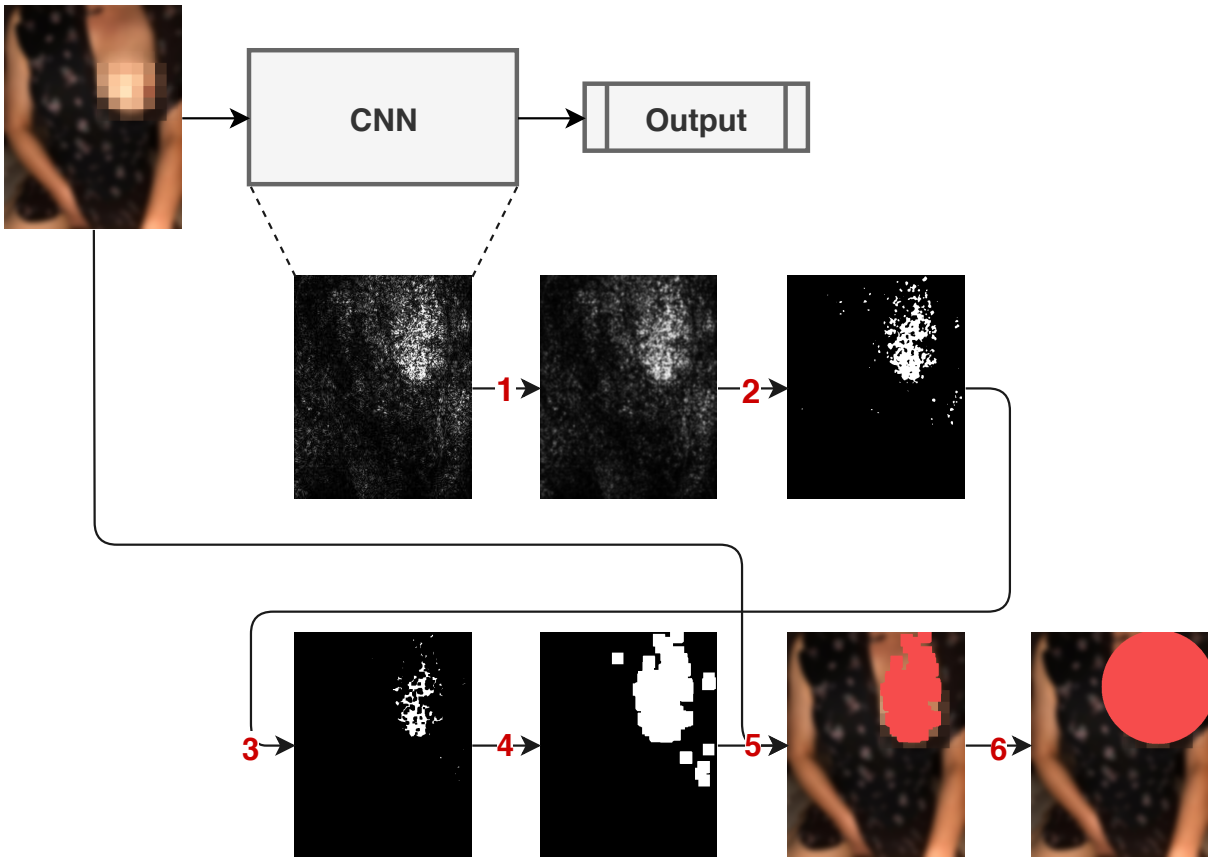


Figure 4.5: The flowchart of the algorithm used for obtaining obfuscation masks

1. A box filter with a kernel size of 5 is applied on the gradient-weighted class activation map. This filter is used for smoothing and image denoising. The noise reduction is noticeable due to reduced sharp changes in intensity levels.

2. Image binarisation is utilised on the blurred map with a threshold of 100 as pixel intensity. The output image has reduced background noise and distortion.

3. To filter out the noise and small grains in the mask, erosion is applied for 2 iterations. This process reduces the peaks while increasing the widths of minimum regions. The output image has reduced positive noises.

4. The mask is dilated using a rectangular structuring element for 10 iterations. This process increases the valleys and increases the width of maximum regions. The output image has reduced negative impulsive noise.

5. Small contours in the image are filtered out, only the largest contour and contours larger than 2/3th of that contour is used for the final mask.

6. This step is optional. In order to obtain regular shapes for obfuscation, we used the smallest circles enclosing each filtered contours.

5 Results and Further Development

In this section, we evaluate the performance of our proposed pipeline. The results represent the performance on unseen testing data. For the testing of the models, manually labelled test-set is used, which we have described in Section 4.2. For testing, the images are preprocessed as the training-set, but no augmentation is applied. For every image, the model outputs a vector for prediction. This vector describes the probability for each class. Thus, the class with the highest probability is accepted as the predicted class.

Performance of the classification network is measured in the scope of experiments presented in Section 5.2. The experiments investigate various techniques or configurations to find an optimal augmentation, loss function, network architecture, and a number of layers. After presenting numerical results, we visualised the learned features of the network to take a more in-depth look into the classification. We used the best-performing model in the test-set for evaluating both classification and automatic obfuscation algorithm. Furthermore, we demonstrated several selected auto-obfuscated images.

For the evaluation of the performance of the pipeline, several metrics that quantify the prediction quality will be used. We use these metrics to make a comparison between different models. The accuracy is the primary metric used for the fair comparison of different models of an experiment. The testing loss is not preferred as we use different loss functions for different models, and therefore a direct comparison is not always possible. Another reason for this choice is that the accuracy metric is also used for regularisation during the training. The model with the highest validation accuracy is chosen as the best model, and the training is terminated if the validation accuracy is not improved for a certain amount of time.

Besides the accuracy metric, we also used Top-2 accuracy that measures how often the predicted class falls in the top 2 values of the output prediction vector. This approach is relevant considering for our use case, the distinguishment between some classes (e.g. explicit nudity, and suggestive nudity) might not be very clear. Assuming that similar classes are more likely

to be mistaken by the network, this metric gives a better understanding of network prediction performance.

5.1 Results for Content Classification

5.1.1 General Evaluation

Comparison of Top-1 and Top-2 accuracies of each model are demonstrated in Figure 5.1. Configurations of these models are presented with detail in Section 5.2. For the content classification network, we set up four experiments in order to find the best configurations for the network. In each experiment, we changed a specific configuration on the best-performing model from the previous experiment. To follow a consistent naming scheme, we renamed the best model of the earlier experiment to compare to those of the proceeding experiment. Even though some modifications and configurations can be arbitrary for the initial experiments, the reasoning of these decisions is verified in the next experiments.

Experiment 1 investigates the performance of different augmentation strategies. As shown in the figure, models trained with augmentation (*Model 1.2, Model 1.3*) were the ones that obtained the most robust results compared to the one (*Model 1.1*) trained without augmentation While the difference between the models trained with aggressive augmentation strategy (*Model 1.3*) and standard augmentation strategy (*Model 1.2*) is almost neglectable, our findings demonstrate that our aggressive augmentation strategy slightly performed better.

As our dataset consists of automatically labelled data, there might be moderate amount of labelling noise. Intending to improve the performance of the model with this dataset, Experiment 2 employs several different loss function modifications that were proposed for robust training with noisy data. Results carried out from this experiment show that using Symmetrical Cross Entropy (Model 2.2) and Cross Entropy with label smoothing (Model 2.3) reveals meaningful improvement over our baseline (*Model 2.1*). The results in Figure 5.1 demonstrated that utilising label smoothing presented an incremental improvement.

Experiment 3 compares our proposed classification head architecture with a single fully connected layer as proposed for the original ResNet architecture. The results demonstrate that our network (*Model 3.1*) outperforms the standard approach (Model 3.2) for our dataset.

Finally, we compared how the number of layers changes the performance of the network. We compared our performance (*Model 4.1*) with ResNet-50 (*Model 4.2*) and ResNet-18 (*Model*

4.3) baseline. A similar pattern of results to those in [3] was obtained. Expectedly, our model based on ResNet-128 is slightly superior to other models with less capacity.

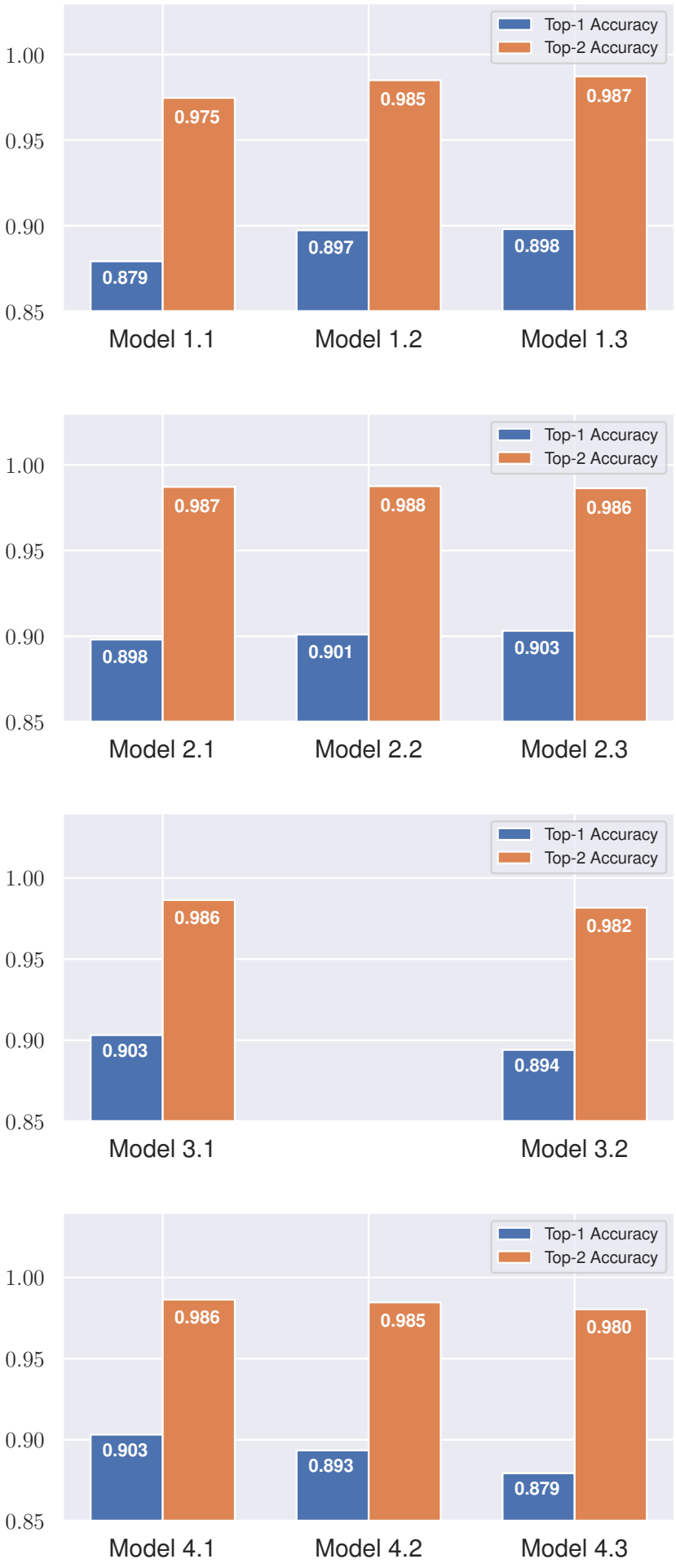


Figure 5.1: Visualisation of testing accuracies from the experiments

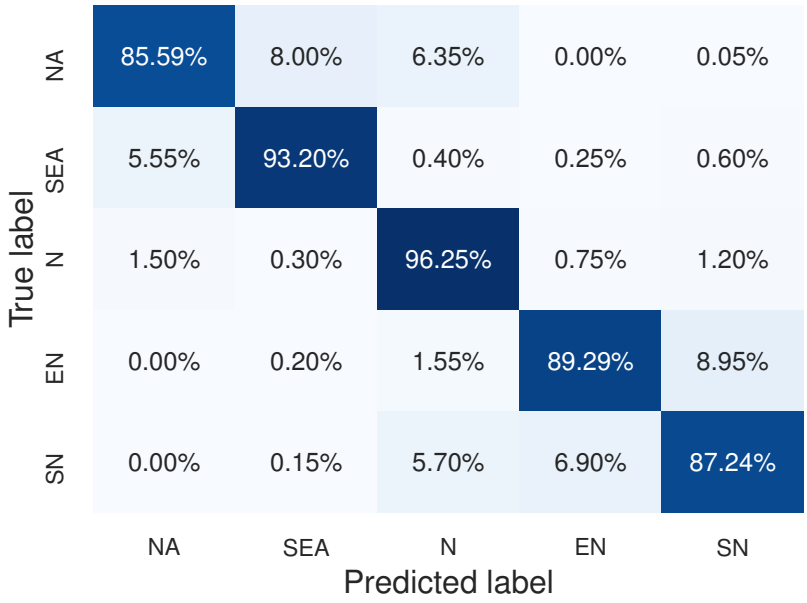
5.1.2 Evaluation of the Best Model

We used the model with the highest accuracy and investigated its prediction performance. The same model is also used for the evalution of the obfuscation algorithm.

In order to evaluate the model, the confusion matrix is presented in Table 5.1. The rows of the confusion matrix describe the true class, and the columns describe the predicted class. The diagonal of the matrix represents the correct predicted samples. The confusion matrix is normalised with respect to total number of true labels.

Looking at the confusion matrix, one might notice that the class pairs often symmetrically confused with each other. Neutral Artwork and Artwork are more likely to be mistakenly labelled with each other rather than other classes. Likewise, Suggestive Nudity and Explicit Nudity are also symmetrically mispredicted. This could be explained by the fact that these classes are not entirely mutually exclusive, and that distinguishing these classes has an intuitive and subjective nature.

Table 5.1: Confusion matrix for content classification



| **NA:** Neutral Artwork | **SEA:** Sexually Explicit Artwork | **N:** Neutral | **EN:** Explicit Nudity | **SN:** Suggestive Nudity |

Moreover, we listed the recall, precision, F1-score values of each class in Table 5.2. The recall defines the hit rate of a class. It is the ratio of the number of the correctly predicted labels divided by the number of all samples for this class in the test set. Besides the precision

characterises the precision of the hit. This is the ratio of the number of the correctly predicted labels divided by the number of predictions for this class. The F1-score is the harmonic mean of the recall and the precision.

Table 5.2: Evaluation metrics of each class

	Precision	**Recall**	**F1-Score**
Neutral Artwork	0.924	0.856	0.889
Sexually Explicit Artwork	0.915	0.932	0.923
Neutral	0.873	0.963	0.916
Explicit Nudity	0.919	0.893	0.906
Suggestive Nudity	0.890	0.872	0.881

5.1.3 Network Visualisation

Neural networks present a superior performance compared to traditional computer vision algorithms. However, they are not decomposable into intuitive components and consequently not easy to interpret [37]. The proposed solution in this thesis work exploits neural networks for automizing content moderation in social media. In this particular use case, interpretability is essential. When a post gets flagged as inappropriate, a human moderator might be clueless about why the network made this decision. The primary goal of interpretability is to identity the failure models [38, 39]. In the scope of this work, the visualisation of the network decisions are presented to clarify how the network makes decision on the content.

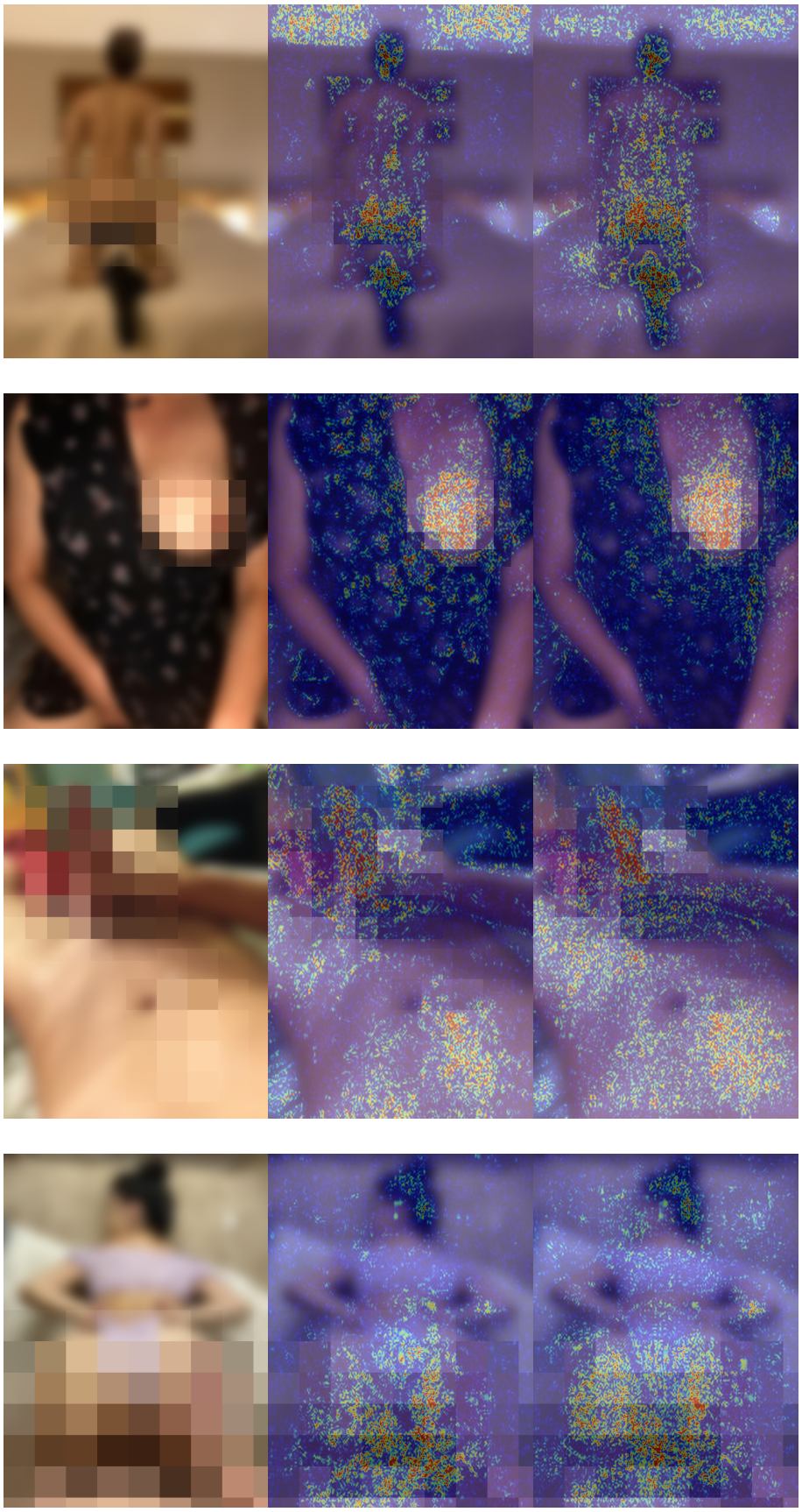
Numerous convolutional neural network visualisation techniques are used to interpret the decision-making behaviour of the network in Figure 5.2, Figure 5.3, Figure 5.4, Figure 5.4, Figure 5.5, and Figure 5.9 for each class. The original input image is presented in column (a) of each row. Column (b) shows the highlighted class-discriminative features by Guided Gradient-weighted Class Activation Map Saliency [4]. A higher resolution class-discriminative visualisation is obtained using Integrated Gradients Map [5] and presented in column (c). Red and yellow regions correspond to high scores for the interested class.

In Figure 5.2, input images are obfuscated, considering the explicit nature of the data. Randomly selected samples illustrate that the network pays attention to the regions that get an image classified as *Explicit Nudity*. Highlighted discriminative features are in line with the description provided for this class in Section 4.2. For the first row in the figure, a region showing uncovered buttocks is highlighted. The highlighted area in the second row corresponds to female-presenting nipples. The third image has two heavily emphasised parts which match to human

genitals and bodily sexual fluids. The last row demonstrates a sexual act, and the corresponding region is highlighted as expected. Considering that all highlighted areas are generalisable features of this class, we can conclude that the model is not biased into non-specific features.

Looking at Figure 5.3, one can see the discriminative features of *Suggestive Nudity* class highlighted. As the description of the class suggests, those parts respectively correspond to uncovered body parts of a person wearing underwear, that of a person in swimwear, male-presenting nipples, and conceivably revealing clothes. Similar to *Explicit Nudity*, the network seems to pay attention to generalisable features of the class.

Figure 5.5 demonstrates a similar pattern to those of Figure 5.2 and Figure 5.3. The highlighted regions mostly correspond to explicitly naked body parts. On the other hand, interpreting the classes *Neutral* and *Neutral Artwork* is not as easy as other classes since their class-discriminative features do not seem to be intuitive, as presented in Figure 5.4, and Figure 5.9.

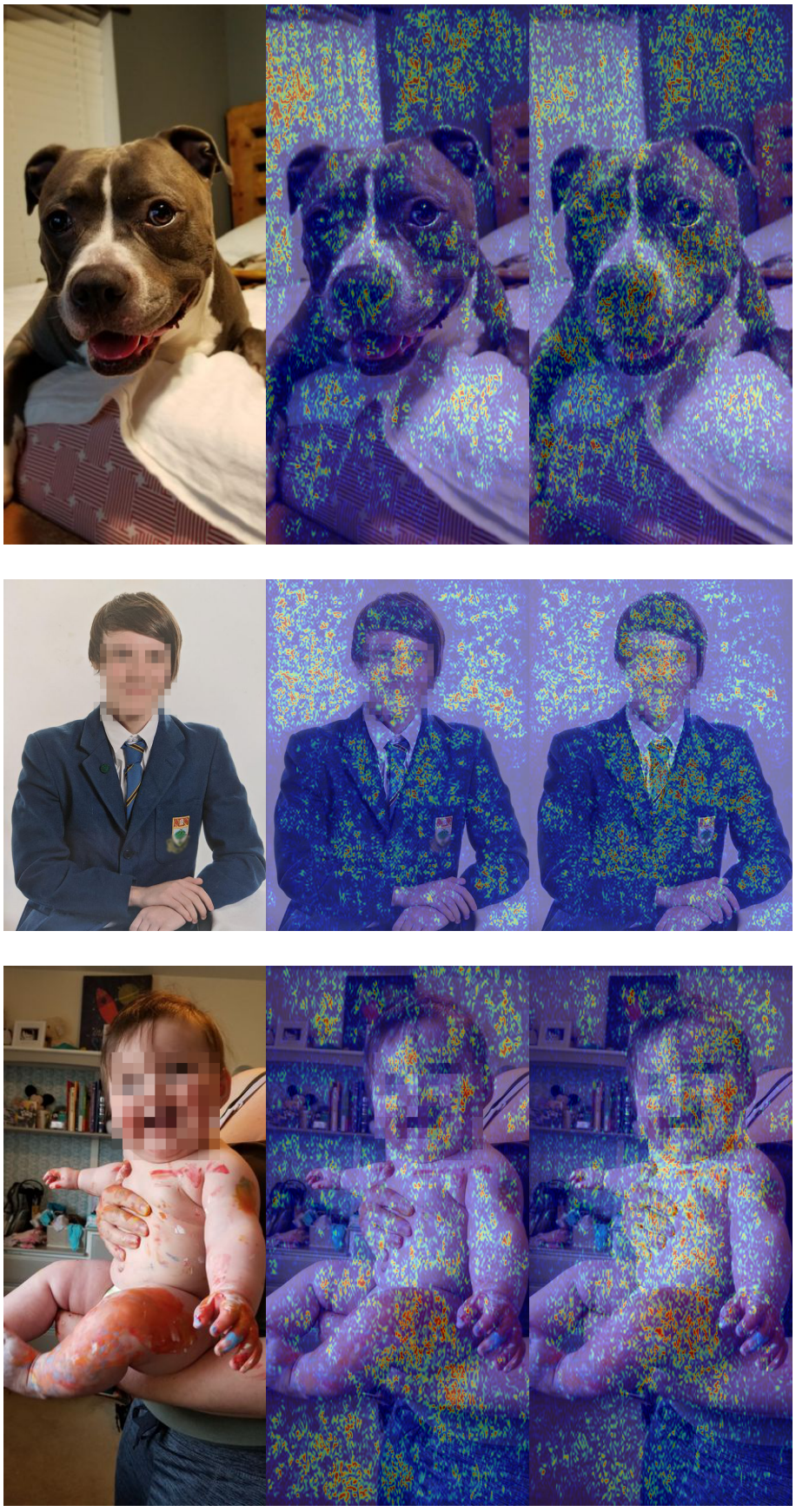


(a) (b) (c)

Figure 5.2: Network visualisation for *Explicit Nudity*. The first column depicts the original input image. The following columns are visualisation results respectively obtained using Guided Gradient-weighted Class Activation Map Saliency [4], Integrated Gradients Map [5].

(a) (b) (c)

Figure 5.3: Network visualisation for *Suggestive Nudity*. The first column depicts the original input image. The following columns are visualisation results respectively obtained using Guided Gradient-weighted Class Activation Map Saliency [4], Integrated Gradients Map [5].



(a) (b) (c)

Figure 5.4: Network visualisation for *Neutral*. The first column depicts the original input image. The following columns are visualisation results respectively obtained using Guided Gradient-weighted Class Activation Map Saliency [4], Integrated Gradients Map [5].

(a) (b) (c)

Figure 5.5: Network visualisation for *Sexually Explicit Artwork*. The first column depicts the original input image. The following columns are visualisation results respectively obtained using Guided Gradient-weighted Class Activation Map Saliency [4], Integrated Gradients Map [5].

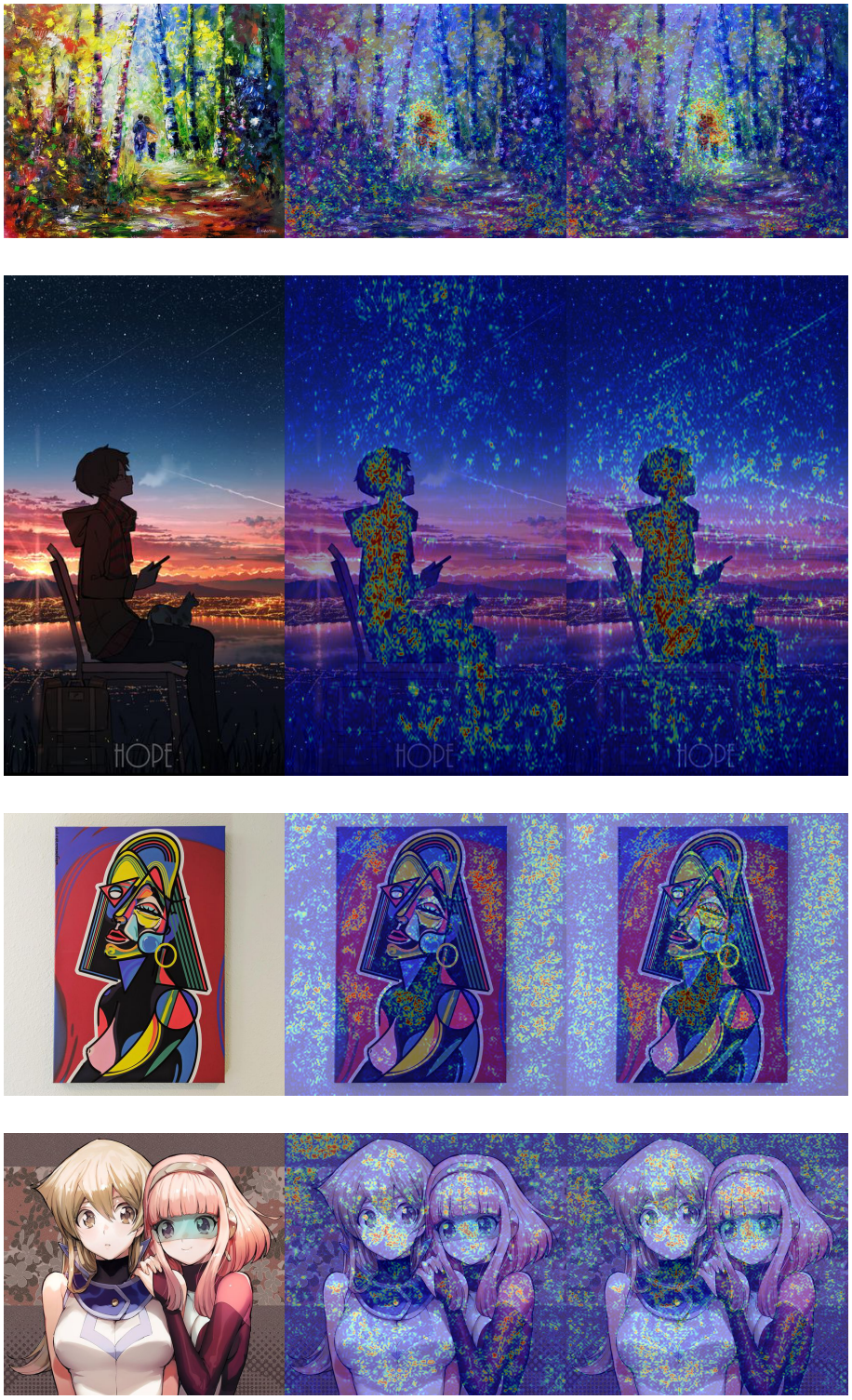
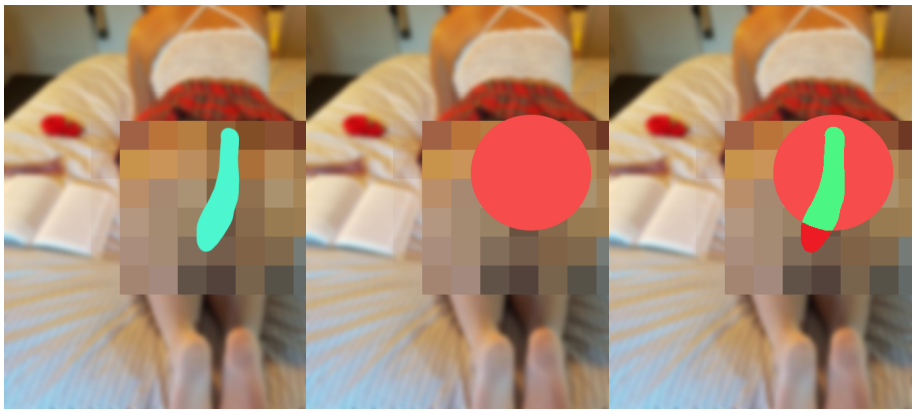


Figure 5.6: Network visualisation for *Neutral Artwork*. The first column depicts the original input image. The following columns are visualisation results respectively obtained using Guided Gradient-weighted Class Activation Map Saliency [4], Integrated Gradients Map [5].

5.2 Results for Automatic Obfuscation

If the input image content is classified as explicit nudity by the classification network, we can utilise the information from the gradient-weighted class activation map to create an obfuscation map. In this section, we will illustrate several selected examples from our testing set and present quantised results of this algorithm.

Evaluation of this method is not straightforward. Definition of how much of the area should be covered is not clear and essentially depends on the use-case. We have not trained our model specifically to obfuscate images but instead use the information from the classification network to generate obfuscation masks. Therefore, using metrics such as intersection over union is not a fair evaluation for the model. Our use case should not penalise the accuracy metric for the area it covers. Therefore, we created a manually annotated test set for the minimum region of the image that needs be covered. This region is described in Section 4.2.



(a) (b) (c)

Figure 5.7: Visualisation of quantised evaluation criteria
(a) Annotated test set for the minimum area that needs to be covered by the obfuscation mask
(b) Generated obfuscation mask
(c) Green area: Correctly covered area, Red area: Uncovered area

We measure the ratio of the minimum region that is intersected by the obfuscation mask. As an example, see Figure 5.7. Our intersection score is defined by the ratio of the green area in Figure 5.7.c divided by the blue area in Figure 5.7.a. Our test-set consists of 100 manually annotated images. The histogram of intersection ratio values obtained from the test-set is presented in Figure 5.8. Overall, we obtained an average intersection ratio of 0.68 in our test set. At least half of the images have an intersection ratio of at least 0.80. These results are summarised in Table 5.3.

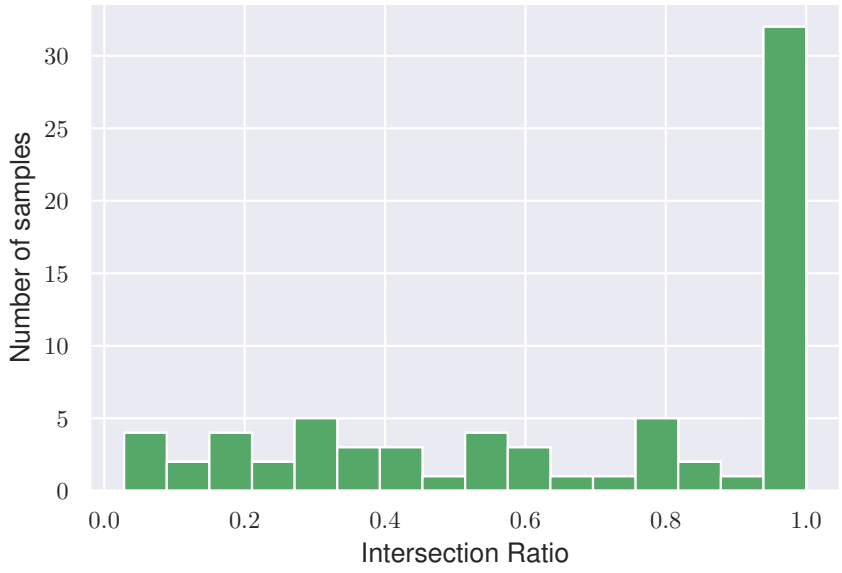


Figure 5.8: Summary of testing accuracies from the experiments

We present selected example outputs in Figure 5.9. The first column corresponds to the input images. The second column is the output obfuscation mask obtained, as described in Step 5 of Section 5.3. The last column is the final post-processed obfuscation mask, as described in Step 6 of Section 5.3

Table 5.3: The summary of quantised results of the obfuscation algorithm

Metric	**Value**
Mean Intersection Ratio	0.6818

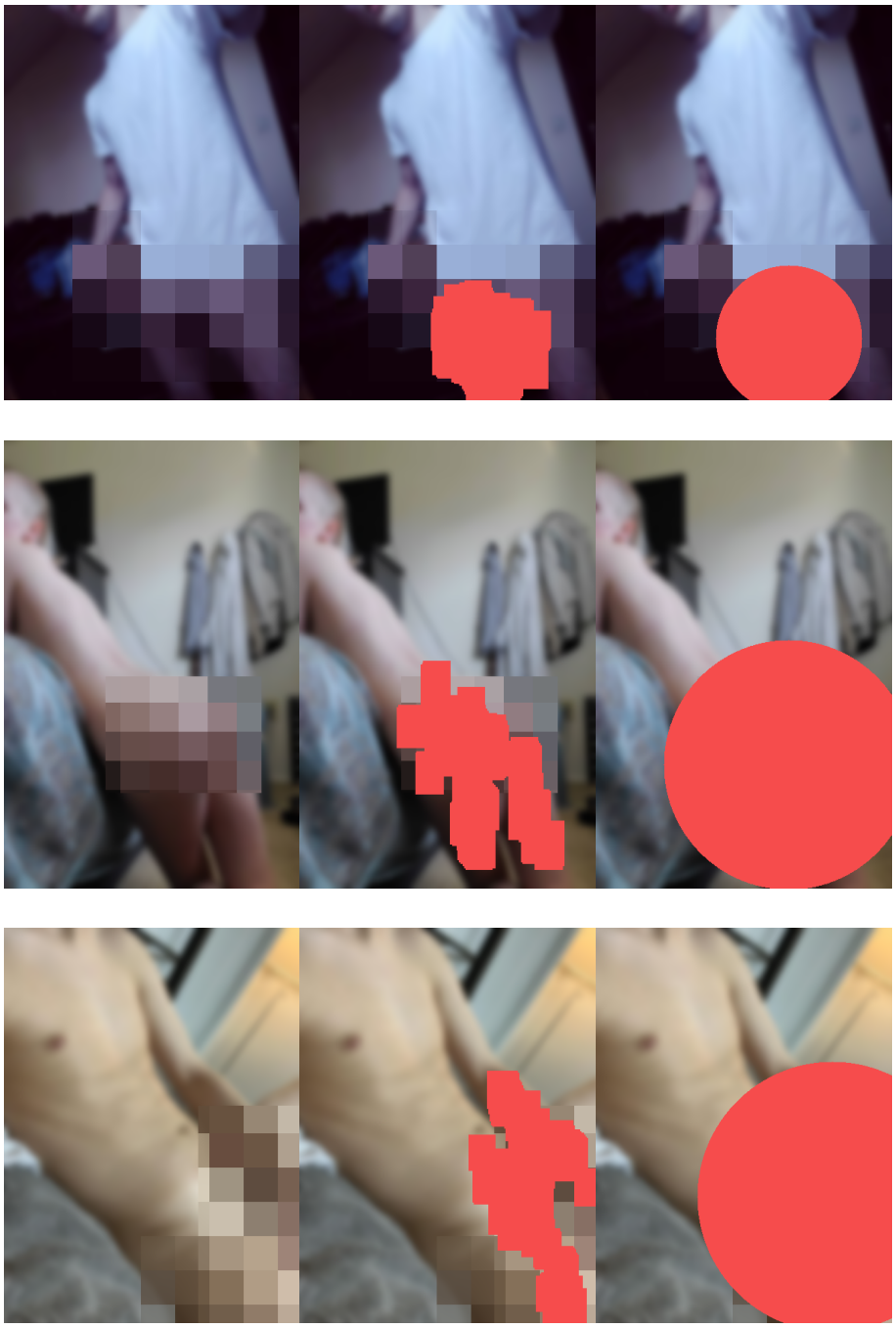


Figure 5.9: Selected example outputs of the obfuscation algorithm

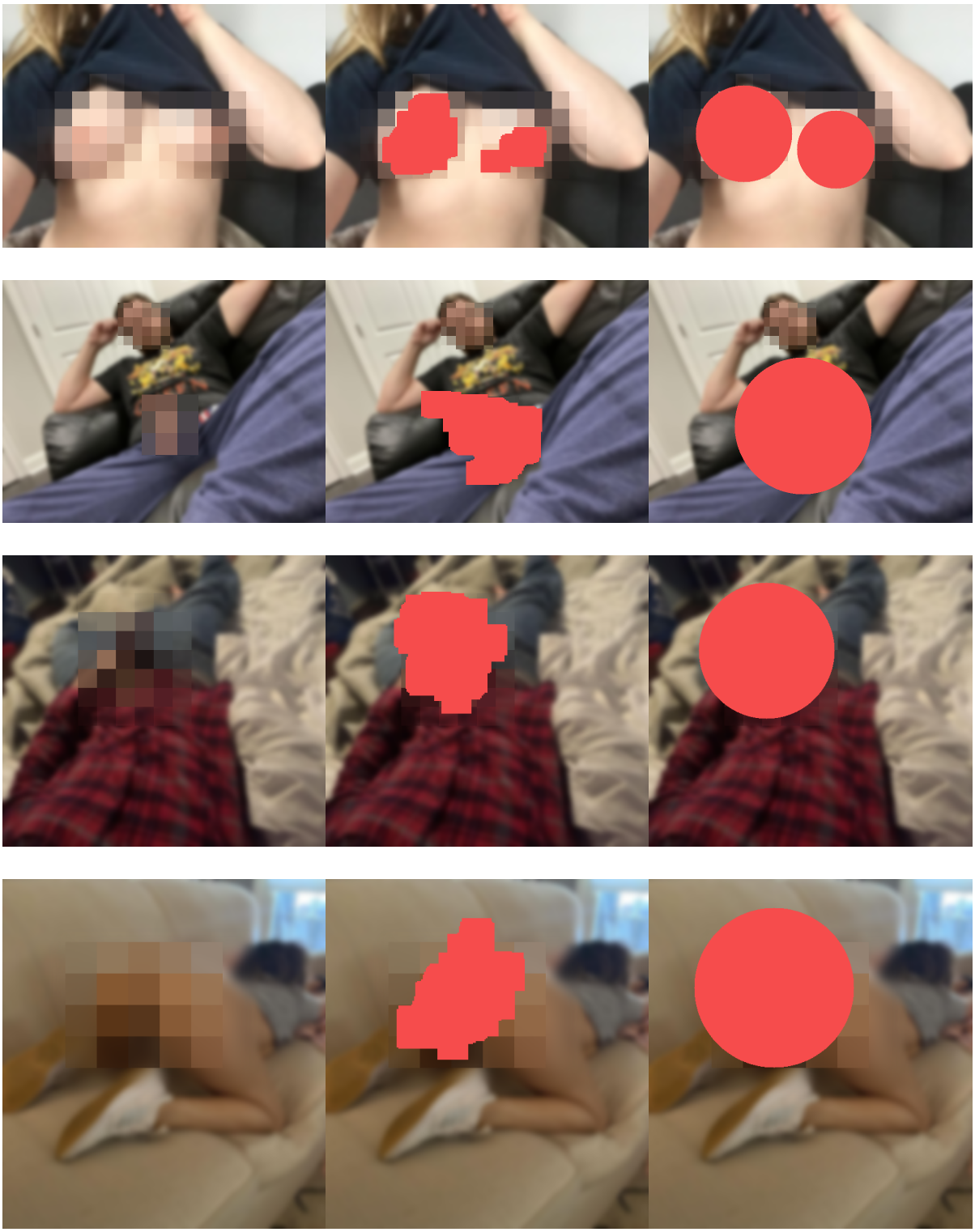


Figure 5.9: Selected example outputs of the obfuscation algorithm

6 Conclusions

6.1 Conclusion

With the internet increasingly getting more accessibility, social media platforms have grown into massive content repositories where millions of users produce and consume billions of contents. This dramatic change consequently forced content management to exploit more of technology and less of human resources on the grounds of a massive volume of content. Automating content moderation is scalable and feasible solution for social media platforms.

Over the scope of this thesis work, we propose a neural networks based automatic content moderation pipeline. Our pipeline consists of two essential parts: the first part that classifies input images into granular content classes and the second class that obfuscated the part of the image that might be inappropriate for the audience. This solution is cost-efficient in term of human labour. Our classification model is trained using scraped live data with label noise, and the obfuscation algorithm does not need additional training but instead use the information learnt by the classification network.

The classification network achieves a top-1 accuracy of 0.903 and a top-2 accuracy of 0.986. If the content is labelled with explicit nudity, the image is obfuscated. The obfuscation algorithm covers a minimum explicitly nude area of 0.68 on average.

6.2 Future Work

There are several improvements which could be made to this work that might result in performance improvements.

Firstly, we could make numerous improvements regarding dataset. The data can be manually labelled using in-house annotators or crowd-sourcing. This will reduce labelling noise significantly and might present improvements over the model performance. Another improve-

ment can be increasing the variety of the dataset by collecting data from additional sources for a longer period of time. We believe this will improve the generalisation ability of the model.

Secondly, we could replace our obfuscation algorithm based on traditional computer vision with weakly supervised object localisation techniques. Considering our specific use-case, this might not result in an improvement on mean intersection ratio metric; though this approach could improve the localisation. Some approaches that might be explored in this regard include .

Bibliography

[1] Carlos Virgilio González. Classification of motor imagery eeg signals with csp filtering through neural networks models. 10 2018.

[2] Alexander LeNail. Nn svg, 2020.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[4] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, Oct 2019.

[5] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks, 2017.

[6] Andrew Arsht and Daniel Etcovitch. The human cost of online content moderation, Mar 2018.

[7] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 44(1.2):206–226, 2000.

[8] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., USA, 1 edition, 1997.

[9] Vinod Nair and Geoffrey E. Hinton. *Rectified Linear Units Improve Restricted Boltzmann Machines*. ICML'10. Omnipress, Madison, WI, USA, 2010.

[10] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels, 2019.

[11] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision, 2015.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

[18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

[20] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017.

[21] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks?, 2014.

[22] Will Archer Arentz and Bjørn Olstad. Classifying offensive sites based on image content. *Comput. Vis. Image Underst.*, 94(1–3):295–310, April 2004.

[23] Michael J. Jones and James M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision*, 46(1):81–96, Jan 2002.

[24] W. Hu, O. Wu, Z. Chen, Z. Fu, and S. Maybank. Recognition of pornographic web pages by classifying texts and images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1019–1034, 2007.

[25] Henry A. Rowley, Yushi Jing, and Shumeet Baluja. Large scale image-based adult-content filtering. In *VISAPP*, 2006.

[26] Hong Zhu, Shuming Zhou, Jianying Wang, and Zhongke Yin. An algorithm of pornographic image detection. In *Proceedings of the Fourth International Conference on Image and Graphics*, ICIG '07, page 801–804, USA, 2007. IEEE Computer Society.

[27] P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recognition*, 40(3):1106 – 1122, 2007.

[28] QING-FANG ZHENG, WEI ZENG, WEI-QIANG WANG, and WEN GAO. Shape-based adult image detection. *International Journal of Image and Graphics*, 06(01):115–124, 2006.

[29] Margaret M Fleck, David A Forsyth, and Chris Bregler. Finding naked people. In *European conference on computer vision*, pages 593–602. Springer, 1996.

[30] A. P. B. Lopes, S. E. F. de Avila, A. N. A. Peixoto, R. S. Oliveira, and A. de A. Araújo. A bag-of-features approach based on hue-sift descriptor for nude detection. In *2009 17th European Signal Processing Conference*, pages 1552–1556, 2009.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[32] Jay Mahadeokar and Gerry Pesavento. Open sourcing a deep learning solution for detecting nsfw images. *Yahoo Engineering*, Sep 2016.

[33] Mohamed Moustafa. Applying deep learning to classify pornographic images and videos. 2015.

[34] Murilo Varges da Silva and Aparecido Nilceu Marana. Spatiotemporal cnns for pornography detection in videos. In Ruben Vera-Rodriguez, Julian Fierrez, and Aythami Morales, editors, *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 547–555, Cham, 2019. Springer International Publishing.

[35] Jônatas Wehrmann, Gabriel S. Simões, Rodrigo C. Barros, and Victor F. Cavalcante. Adult content detection in videos with convolutional and recurrent neural networks. *Neurocomputing*, 272:432 – 438, 2018.

[36] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[37] Zachary Chase Lipton. The mythos of model interpretability. *CoRR*, abs/1606.03490, 2016.

[38] Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. Analyzing the behavior of visual question answering models, 2016.

[39] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 340–353, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

Non-exclusive licence to reproduce thesis and make thesis public

I, Dogus Karabulut

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

"Neural Networks Based Automatic Content Moderation on Social Media"

supervised by Dr. Cagri Ozcinar and Prof. Gholamreza Anbarjafari.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Dogus Karabulut

20.05.2020